Semi-Supervised Thai Sentence Segmentation Using Local and Distant Word Representations

Mr. Chanatip Saetia

A Thesis Submitted in Partial Fulfillment of the Requirements
for the Degree of Master of Engineering in Computer Engineering
Department of Computer Engineering
FACULTY OF ENGINEERING
Chulalongkorn University
Academic Year 2020

การตัดประโยคภาษาไทยแบบกึ่งมีผู้สอนโดยใช้ตัวแทนของคำประเภทเฉพาะที่และไกล

นายชนาธิป แซ่เตีย

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิศวกรรมศาสตรมหาบัณฑิต
สาขาวิชาวิศวกรรมคอมพิวเตอร์ ภาควิชาวิศวกรรมคอมพิวเตอร์
คณะวิศวกรรมศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย
ปีการศึกษา 2563

Thesis Title	Semi-Supervised Thai Sentence Segmentation Using Local and Distant Word Representations
By	Mr. Chanatip Saetia
Field of Study	Computer Engineering
Thesis Advisor	Assistant Professor PEERAPON VATEEKUL, Ph.D.

Accepted by the FACULTY OF ENGINEERING, Chulalongkorn University in Partial Fulfillment of the Requirement for the Master of Engineering

.... Dean of the FACULTY OF ENGINEERING

()

THESIS COMMITTEE

.... Chairman

(Professor BOONSERM KIJSIRIKUL, Ph.D.)

.... Thesis Advisor

(Assistant Professor PEERAPON VATEEKUL, Ph.D.)

.... Examiner

(Ekapol Chuangsuwanich, Ph.D.)

.... External Examiner

(Thadpong Pongthawornkamol, Ph.D.)

ชนาธิป แซ่เตีย : การตัดประโยคภาษาไทยแบบกึ่งมีผู้สอนโดยใช้ตัวแทนของคำประเภทเฉพาะที่และไกล. (Semi-Supervised Thai Sentence Segmentation Using Local and Distant Word Representations) อ.ที่ปรึกษาหลัก : ผศ. ดร.พีรพล เวทีกูล

ประโยคคือหน่วยไวยากรณ์ที่มีขนาดเล็กที่สุด เพื่อที่สื่อใจความสำคัญครบถ้วนในประโยค ซึ่งช่วยในการแบ่งข้อความที่ขนาดยาวให้เป็นหน่วยที่เล็กลง อย่างไรก็ตามในภาษาไทย ไม่มีตัวแบ่งประโยคที่บ่งชี้ชัด เราจึงได้พัฒนาโมเดลการเรียนรู้เชิงลึกเพื่อการตัดประโยคจากข้อความ ซึ่งประกอบด้วยสามองค์ประกอบ อย่างแรกคือการใช้ตัวแทนข้อมูลคำข้างเคียง หรือตัวแทนข้อมูลแบบใกล้ในการจับกลุ่มคำที่อยู่ใกล้กับตัวแบ่งประโยค อย่างที่สองคือการสนใจคำที่เป็นอนุประโยคที่อยู่ด้วยตัวเองไม่ได้ โดยใช้ตัวแทนข้อมูลแบบไกลซึ่งได้จากกลไกจุดสนใจ อย่างสุดท้ายคือการใช้สองเทคนิคเพื่อใช้ประโยชน์จากข้อมูลที่ไม่มีการกำกับข้อมูล เนื่องจากข้อมูลที่มีการกำกับข้อมูลนั้นมีน้อย และยังยากและต้องการเวลาในการกำกับข้อมูล โดยเทคนิคแรกคือการสอนแบบหลายมุมมอง ซึ่งเป็นการเรียนรู้กึ่งมีผู้สอน และเทคนิคที่สองคือการใช้โมเดลภาษาแบบถูกสอนมาก่อนเพื่อพัฒนาตัวแทนของข้อมูล ในการทดลองของการตัดคำภาษาไทย โมเดลของเราสามารถลดความผิดพลาดสัมพัทธ์ลง 7.4% และ 18.5% เมื่อเปรียบเทียบกับโมเดลก่อนหน้า เมื่อเทียบบนชุดข้อมูล Orchid และ UGWC ตามลำดับ เรายังได้ทดสอบกับงานที่ใกล้เคียงกันบนภาษาอังกฤษ คือการทำนายเครื่องหมายวรรคตอนที่หายไป โดยโมเดลของเราสามารถลดความผิดพลาดสัมพัทธ์เมื่อเทียบกับโมเดลก่อนหน้าลง 7.6% จากศึกษาพบว่าการใช้ตัวแทนข้อมูลจากคำใกล้เคียงเป็นปัจจัยหลักในการพัฒนาขึ้นบนภาษาไทย ในขณะที่ในภาษาอังกฤษการเรียนรู้กึ่งมีผู้สอนเป็นปัจจัยหลักในการทำให้โมเดลดีขึ้น

สาขาวิชา	วิศวกรรมคอมพิวเตอร์	ลายมือชื่อนิสิต	....
ปีการศึกษา	2563	ลายมือชื่อ อ.ที่ปรึกษาหลัก	....

6170135321 : MAJOR COMPUTER ENGINEERING
KEYWORD:

Chanatip Saetia : Semi-Supervised Thai Sentence Segmentation Using Local and Distant Word Representations. Advisor: Asst. Prof. PEERAPON VATEEKUL, Ph.D.

A sentence is typically treated as the minimal syntactic unit used for extracting valuable information from a longer piece of text. However, in written Thai, there are no explicit sentence markers. We proposed a deep learning model for the task of sentence segmentation that includes three main contributions. First, we integrate n-gram embedding as a local representation to capture word groups near sentence boundaries. Second, to focus on the keywords of dependent clauses, we combine the model with a distant representation obtained from self-attention modules. Finally, due to the scarcity of labeled data, for which annotation is difficult and time-consuming, we also investigate and adapt two techniques, allowing us to utilize unlabeled data. The first one is Cross-View Training (CVT) as a semi-supervised learning technique, and the second one is a pre-trained language model (ELMo) to improve representation. In the Thai sentence segmentation experiments, our model reduced the relative error by 7.4% and 18.5% compared with the baseline models on the Orchid and UGWC datasets, respectively. We also applied our model to the task of punctuation restoration on the IWSLT English dataset. Our model outperformed the prior sequence tagging models, achieving a relative error reduction of 7.6%. Ablation studies revealed that utilizing n-gram representations was the main contributing factor for Thai, while the semi-supervised training helped the most for English.

Field of Study:	Computer Engineering	Student's Signature ....
Academic Year:	2020	Advisor's Signature ....

ACKNOWLEDGEMENTS

This master's dissertation can not be succeeded without supporting, guiding, and influencing from several groups of people. First, I would like to acknowledge my supervisor, Asst. Prof. Peerpon Vateekul. He does not only gives me the suggestion to accomplish my works but he also encourages me to improve my work gradually. He also accepts the mistaken point in this work and offers assistance to jump over an obstacle. Secondly, I would like to thanks Dr. Ekapol Chuangsuwanish, who presents an inspiring idea and shapes my work. Also, I would like to thanks my dissertation examiners: Prof. Boonserm Kijsirikul and Dr. Thadpong Pongtawornkamol.

I also appreciate joining the Data mind Labs in the Department of Computer Engineering, Faculty of Engineering, Chulalongkorn University. Amiable colleges in the labs always give the suggestion and assistance to keep my work going. This dissertation is a direct result of the discussion with others via a peer review system.

This work was funded by KLabs at Kasikorn Business Technology (KBTG), which provided facilities and data. The procedures that were conducted based on social data are visible to the public, and ethical issues that can arise from the use of such data were addressed. I would like to thank the linguists Sasiwimon Kalunsima, Nutcha Tirasaroj, Tantong Champaiboon, and Supawat Taerungruang for annotating the UGWC dataset used in this study. Moreover, I would like to give a special thanks to Dr. Tawunrat Chalothorn, who works as my supervisor in Klabs and allow working with this project.

Lastly, I would like to thanks my family and my girlfriend for their unconditional love and morale. With great understanding, they always listen to the difficulty of the work and give support until this dissertation is accomplished.

Chanatip Saetia

TABLE OF CONTENTS

LIST OF TABLES

Page

LIST OF FIGURES

1. INTRODUCTION

1.1. Motivation

Machine translation, automatic text summarization, dependency parsing, and semantic parsing are useful for processing, analyzing, and extracting meaningful information from text. These tasks require a basic unit that has a simple grammatical structure to reduce the tasks' complexity. For example, dependency parsing [1], which extracts a syntactic structure for language understanding, needs to consider every word pair in the text to assign a relation. Thus, the complexity of dependency parsing depends on the input text's sequence length. If the text is segmented into smaller parts, the task will be easier to perform. However, the basic unit should be not only as small as possible but also required to be complete in itself. For instance, if the basic unit is too short and does not contain sufficient information, many meaningful relations in dependency parsing will not be extracted inside the basic unit. Thus, the basic unit should be small yet contain complete meaning to make the mentioned tasks more efficient.

A sentence is raised as a basic unit because a sentence is always complete in itself. Moreover, a sentence can also be easily extracted from raw text because the sentence boundaries in English are easily identified by a period (".") [2]. Many prior works require a sentence to perform their tasks. For example, in machine translation, a sentence pair is required for supervised training [3-5]. Meanwhile, many automatic text summarization works treat a sentence as one item of information and select the important ones to be summarized [6-8]. Dependency parsing also requires a sentence as an input text to extract its syntactic structure that the machine understands [1, 9, 10].

However, there is no explicit end-of-sentence marker for identifying the sentence boundary in some written languages, such as Thai, Arabic, Khmer, and Lao [11]. Therefore, extracting sentences from raw text in these languages is not trivial. For example, "He wishes to buy 4 ingredients for cooking an omelet. Therefore, he goes shopping and buys an egg, milk, salt, and pepper." The text can be segmented with a period "." into two sentences "He wishes to buy 4 ingredients for cooking a fried egg." and "Therefore, he goes shopping and buys an egg, milk, salt, and pepper." Meanwhile, in Thai, the same text is "เขาต้องการที่จะซื้อส่วนประกอบ 4 อย่างสำหรับทำไข่เจียว ดังนั้นเขาจึงไปซื้อไข่ นม เกลือ และพริกไทย" Note that there is no punctuation or even a word to indicate where the text should be segmented. Although most Thai people usually use a space character as a sentence boundary, the illustrated text shows that only one out of six space characters is a sentence boundary. Therefore, there is no explicit marker for identifying sentence boundaries to segment the text, especially for the exemplified case.

Prior works on Thai sentence segmentation have adopted traditional machine learning models to predict where a sentence boundary is in the text. The authors of [12-14] proposed traditional models to determine whether a considered space is a sentence boundary based on the words and their part of speech (POS) near the space. Although a space is usually considered essential as a sentence boundary marker in Thai, approximately 23% of the end of sentences is not a space character in a news domain corpus [15]. Thus, Zhou N. et al. [15] proposed a conditional random field (CRF)-based model with n-gram features to predict which word is the sentence boundary. This work considered Thai sentence segmentation as a sequence tagging problem similar to named entities recognition and part-of-speech tagging. Each word in the text will be classified whether it is the end of a sentence or not, as shown in Figure 1. With a CRF module [16], the model extracts sentence-level tag information, where each prediction in a sequence considers the previous predicted tags instead of only the input words. Meanwhile, the n-gram [17], which is an input feature, is constructed from a combination of words around the considered position. This method achieves the state-of-the-art result for Thai sentence segmentation and achieves greater accuracy than other models by approximately 10% on the Orchid dataset [18].

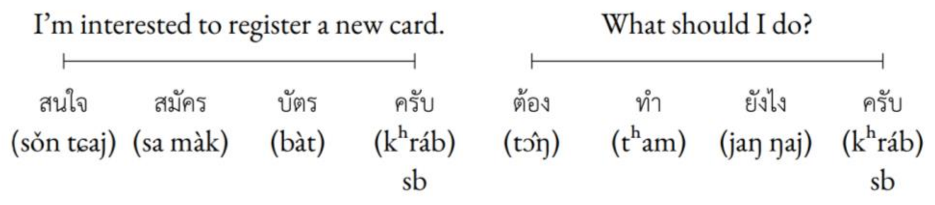


Figure 1. An example of a labeled paragraph. Here, sb represents a sentence boundary.

Several deep learning approaches have been applied in various tasks of natural language processing (NLP), including the long short-term memory [19], self-attention [20], and other models. To tackle the sequence tagging problem, Huang Z. et al. [21] proposed a deep learning model called Bi-LSTM-CRF, which integrates a CRF module to gain the benefit of both deep learning and traditional machine learning approaches. In their experiments, the Bi-LSTM-CRF model achieved an improved level of accuracy in many NLP sequence tagging tasks, such as named entity recognition, POS tagging and chunking. This model is also used as a base (backbone) model for many works that achieve promising accuracy in sequence tagging tasks. [22-26].

In this work, two models are chosen as baseline models. First, the Bi-LSTM-CRF model is adopted as our baseline and backbone model because many sequence tagging works also apply the model as a backbone model and yield respectable performance. Second, the CRF model, which achieved the best result on the Thai sentence segmentation task [27] is also treated as another baseline model to compare with prior works.

This work makes three contributions to improve Bi-LSTM-CRF for Thai sentence segmentation. These contributions apply the suitable deep learning modules carefully to tackle various problems of this task. Each contribution is described as follows.

First, we propose adding n-gram embedding to Bi-LSTM-CRF due to its success in [27]. To integrate n-gram features in Bi-LSTM-CRF, the feature is embedded into a dense embedding vector trained along with the model. With the n-gram embedding addition, it can extract a local representation from n-gram embedding, which helps in capturing word groups that exist near a sentence boundary. Although Jacovi A. et al. [28] reported that a convolutional neural network (CNN) can be used as an n-gram detector to capture local features, we chose n-gram embedding over a CNN due to its better accuracy, as will be shown in Section 5.4.1.

Second, we propose adding distant representation into the model via a self-attention mechanism [20], which can focus on the keywords of dependent clauses that are far from the considered word. Self-attention has been used in many recent state-of-the-art models, most notably the Transformer [20] and Bidirectional Encoder Representations from Transformers (BERT) [29]. BERT has outperformed Bi-LSTM on numerous tasks, including question answering and language inference. Therefore, we choose to use self-attention modules to extract distant representations along with local representations to improve model accuracy.

Third, we also apply two techniques to utilize unlabeled data: semi-supervised learning and a pre-training method, which are essential for low-resource languages such as Thai, for which annotation is costly and time-consuming. The first technique is semi-supervised learning [30]. Many semi-supervised learning approaches have been proposed in the computer vision [31, 32] and natural language processing [33-35] fields. Our choice for semi-supervised learning to enhance model representation is Cross-View Training (CVT) [33]. Clark K. et al. [33] claim that CVT can improve the representation layers of the model, which is our goal. However, CVT was not designed to be integrated with self-attention and CRF modules; consequently, we provide a modified version of CVT in this work.

Instead of using only CVT to improve the representation with unlabeled data, the pre-training method is also adopted in our model. There are many proposed pre-trained word representations in the field of NLP [29, 36-38]. In this task, to decide whether each word is a sentence boundary, the context of the given text is essential. Thus, contextualized word representations are chosen over Word2Vec [37] that embeds each word independently from the context. Currently, BERT [29] or other variant contextual models of BERT are usually a part of the state-of-the-art methods for many tasks [22, 39, 40]. However, BERT models require a large amount of GPU memory in the training process. Therefore, ELMo, which needs less GPU memory than BERT [41], is chosen in this work due to our resource limitations. Note that some variants of BERT models are also optimized for less memory usage but require additional techniques, such as knowledge distillation [41] and factorized embedding parameterization [42], which requires further investigation on the Thai corpus.

Based on the above contributions, we pursue the experiment on Thai sentence segmentation. The experiment is performed on two Thai sentence segmentation datasets, including Orchid and UGWC [43]. In the experiment, the proposed model is compared to four existing methods: POS-trigram [13], Winnow [12], Maximum entropy [14], and CRF [27]. Moreover, we also perform ablation studies, which add each proposed contribution sequentially to observe their individual effect on the performance. The ablation studies found that local representation (n-grams) yields the largest improvement on Thai sentence segmentation; thus, its effect is further analyzed using interpretation techniques.

1.2. Research objectives

This thesis aims to solve Thai sentence segmentation using a deep learning method. The main hypothesis is "*The accuracy in Thai sentence segmentation will be enhanced by applying various deep learning modules and training with a semi-supervised method along with a pre-trained language model.*" Moreover, we also perform an interpretation of n-gram features to reveal the understanding of the trained model. To sum up, the following two objectives are specified and will be addressed by the proposed contributions in this work:

- **To propose** a novel deep learning model for Thai sentence segmentation
- **To utilize** unlabeled data with semi-supervised techniques to improve the representation of a model along with a pre-trained language model

1.3. Contributions

The studies in this thesis will focus on four main contributions to improve the accuracy of the model in Thai sentence segmentation as following:

- We added n-gram features to Bi-LSTM-CRF as local representation to captures the word groups around boundaries.
- We applied self-attention modules to extract a distant representation that focuses on the keywords in the dependent clause.
- We presented a semi-supervised learning technique or CVT, which is specific for Thai sentence segmentation, to utilize unlabeled data.
- We adapted ELMo or deep contextual embedding as a pre-trained language model to improve the representation of the model.

1.4. Thesis outline

Chapter 2 presents an overview of background knowledge that is related to this thesis. A sentence in Thai is defined. Deep learning modules, which are used in this work, are reviewed. Furthermore, we describe how to apply a deep learning model with semi-supervised learning and elaborate on the detail of CVT. The pre-trained language model and ELMo are also included in this chapter. Finally, interpretation approaches of a deep learning model are explained.

In Chapter 3, we review the prior works of Thai sentence segmentation for a comparison with our model. Moreover, we also review the literature on a related task which is English punctuation restoration.

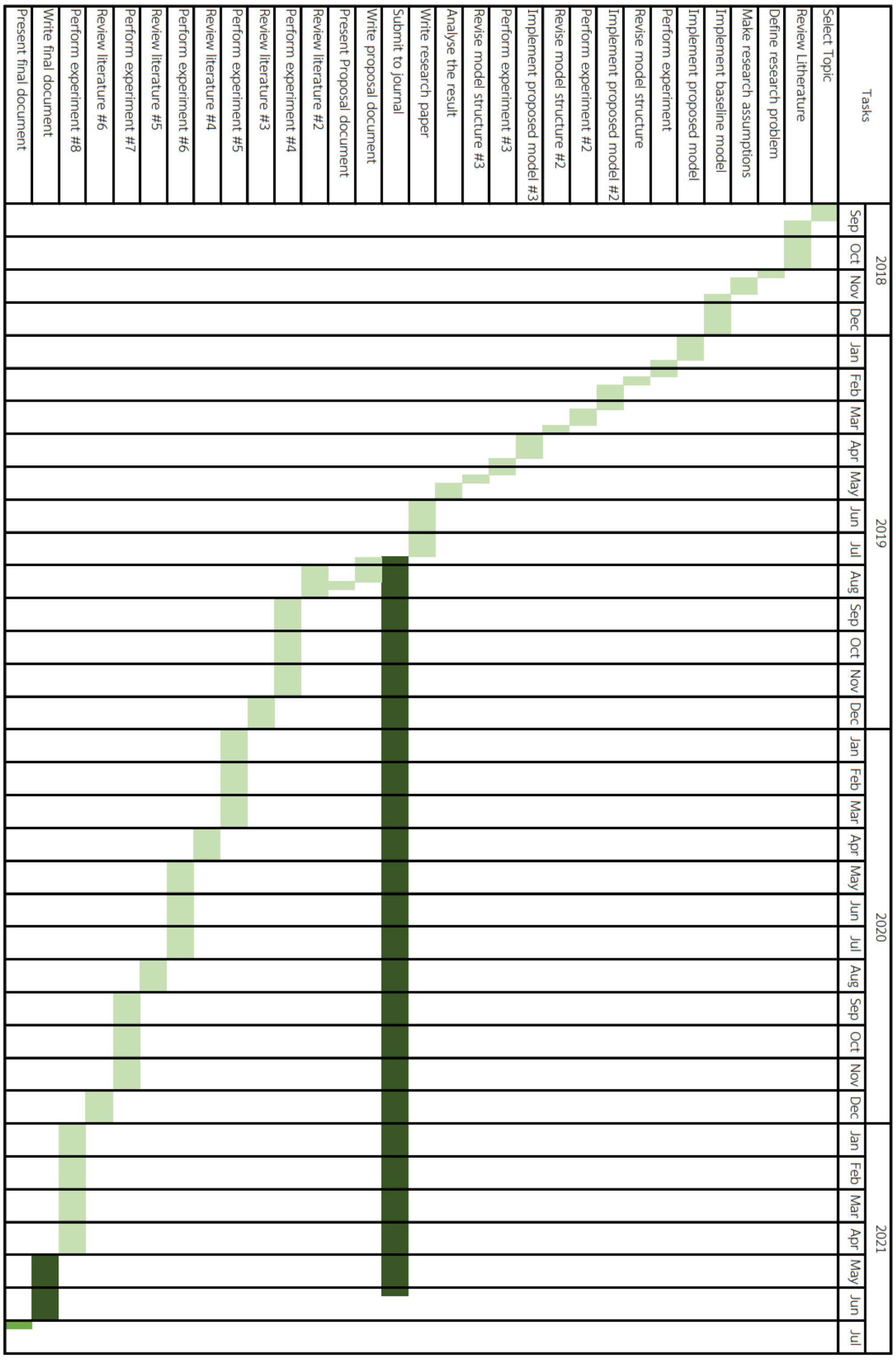
Chapter 4 presents our proposed model architecture, including local and distant features. We also provide how to train the model with a novel CVT and how to pre-trained the language model.

In Chapter 5, we describe the experimental setups and the results on both Thai sentence segmentation and English punctuation restoration. The data statistics and hyper-parameter in our experiments are described in the experimental setups section. Meanwhile, the results show a comparison between our model and the baseline and the improvement of each contribution. Moreover, we analyze how the proposed contribution improves the model. In addition, to understand the impact of n-gram features, we also perform the model interpretation to indicate the importance of local representation is in both tasks.

Finally, Chapter 6 concludes this thesis and shows the best performance for each task. Also, we summarize the ablation studies about each contribution.

1.5. Research schedule

In this section, Gantt chart is provided with the project's activities and the duration of each activity:



1.6. Publications

The work in this thesis primarily relates to the following peer-reviewed article:

1. Saetia, C., Chuangsuwanich, E., Chalothorn, T., & Vateekul, P. (2019). Semi-supervised Thai Sentence segmentation using local and distant word representations. *Engineering Journal*.

2. BACKGROUND KNOWLEDGE

In this chapter, the background knowledge related to this work is presented. There are four subsections included. The first section describes what a sentence is in Thai. Second, deep learning models related to the work is described. Third, we explain semi-supervised learning algorithms, including Cross-View Training. Finally, the last subsection presents various of pre-trained language models.

2.1. Sentence in Thai

A sentence is considered as a minimal syntactic unit for many tasks such as text summarization, question answering, and machine translation. To extract sentences from the input text, a machine exploits the explicit sentence marker like period "." in English for identifying sentence boundary.

However, there are several languages, such as Thai, Lao, and Myanmar, of which sentence marker is not explicit. In Thai, the text does not contain any marker which certainly identifies the sentence boundary; nevertheless, the Thai writer usually uses space as a vital element to separate apart of text into the sentence. In contrast, there are also three ways to use space in contexts [44]. The first is before and after an interjection. The second is before conjunctions. Before and after a numeric expression is the last one. Therefore, segmenting text into sentences is not simply performed by splitting with space.

As studied in [11], annotated sentence boundary is found in three cases with substantial agreement among people. First, the sentence boundary is found when the topic shift occurs. Second, the overt noun and pronoun, which is used for continuing the topic in a coordinate clause, is seen as the sentence boundary. In addition, the coordinate clause, which is identified from a zero subject, is not considered as a new sentence. The last case is that sentence boundary is identified where the discourse marker is found, such as "และต่อมา" (and then), "ตลอดระยะเวลาดังกล่าวนี้" (throughout this period), "ในสมัยนั้น" (in this period) and "ในระยะแรก" (in the first phrase), is found.

We also provides ten examples of sentence segmentation from various sources including Pantip's posts, books and dissentions. '|' is considered as a sentence boundary in the following text.

- ปัจจุบันพลังงานจากก๊าซธรรมชาติมีบทบาทมากขึ้น โดยเป็นแนวทางหนึ่งในการลดการใช้พลังงานจากน้ำมัน และเป็นพลังงานอีกทางเลือกหนึ่งที่ประเทศไทยเลือกใช้ | ซึ่งที่ผ่านมามีการใช้ประโยชน์จากก๊าซธรรมชาติไม่มากเท่าที่ควร | เนื่องจากมีความยุ่งยากในการขนส่งก๊าซธรรมชาติ โดยต้องลงทุนจำนวนมากในการก่อสร้างท่อส่งก๊าซ | ดังนั้นหากแหล่งสำรองก๊าซธรรมชาติตั้งอยู่ห่างจากแหล่งที่ต้องการใช้เป็นพลังงานแล้ว ก็จะไม่มีการนำมาใช้ประโยชน์จากแหล่งก๊าซธรรมชาตินั้นๆ | เนื่องจากต้องเสียค่าใช้จ่ายเป็นจำนวนมากในการก่อสร้างท่อส่งก๊าซ
- เจอบ่อยมากค่ะพวกที่พึ่งมาคุยกันได้แค่วันสองวันแล้วมาขอเป็นแฟนเลย | เคยเจอที่อาการหนักสุดคือ คุยวันแรกก็ขอจีบเลย | ละก็มาขอวิดีโอคอล | แต่เราไม่เอาเค้าก็เลยขอแค่โทรปกติ | แล้วก็เล่นเกมด้วยกันไปค่ะ
- สวดมนต์เย็นบนพระที่นั่ง บ่ายวันนี้มั้งคะ | เสด็จให้มาทูลถามเสด็จว่าจะเสด็จหรือไม่เสด็จ | ถ้าเสด็จจะเสด็จ เสด็จจะเสด็จด้วย
- ในปัจจุบันมีบริษัทผลิตละครโทรทัศน์ที่มีชื่อเสียงหลายค่ายที่ผลิตละครอย่างต่อเนื่องเพื่อแพร่ภาพออกอากาศทางโทรทัศน์ระบบฟรีทีวี ได้แก่ สถานีโทรทัศน์ไทยทีวีสีช่อง 3 สถานีโทรทัศน์กองทัพบกช่อง 5 สถานีโทรทัศสีกองทัพบกช่อง 7 และสถานีโทรทัศน์โมเดิร์นไนน์ทีวี (ช่อง 9) | โดยแต่ละค่ายได้ผลิตผลงานละครที่โด่งดังและประสบความสำเร็จมากมาย | บริษัทผลิตละครที่มีอยู่ในปัจจุบัน บ้างก่อตั้งมาแล้วหลายปี สั่งสมประสบการณ์มามากมาย บ้างก็เพิ่งก่อตั้งไม่นาน | โดยแต่ละค่ายย่อมมีกลยุทธ์และวิธีการดำเนินงานที่แตกต่างกันไป | ต่างทำหน้าที่ผลิตละครโทรทัศน์เพื่อตอบสนองความต้องการและความพึงพอใจของผู้ชม และดึงดูดผู้ชมให้สนใจชมละครของตน | ซึ่งในปัจจุบันมีบริษัทผลิตละครโทรทัศน์รายใหญ่ เป็นที่รู้จักของประชาชนอยู่หลายบริษัท
- การฉีดยาชาเฉพาะที่เป็นสิ่งที่ทันตแพทย์เกือบทั้งหมดทำก่อนการรักษาทางทันตกรรมในเด็ก | แต่การฉีดยาชาเป็นสิ่งที่ก่อให้เกิดความกลัวและกังวลมากที่สุด | เด็กมักกลัวเข็ม | กังวลต่อสิ่งแวดล้อมใหม่ขณะมาทำการรักษา | รวมไปถึงอาจมีประสบการณ์ที่ไม่ดีจากการฉีดวัคซีน เจาะเลือด หรืออาจได้รับคำบอกเล่าที่น่ากลัว | จึงทำให้มีพฤติกรรมต่อต้าน ขัดขวางการฉีดยาชา
- การเรียนกวดวิชาแม้จะก่อให้เกิดประโยชน์ แต่การเรียนกวดวิชาก็ได้ก่อให้เกิดผลกระทบในด้านลบพอสมควร | โดยเฉพาะการสร้างภาระค่าใช้จ่ายแก่ผู้ปกครองอย่างมาก | นอกจากการสร้างภาระค่าใช้จ่ายกับผู้ปกครองแล้ว การกวดวิชายังก่อให้เกิดผลกระทบต่อนักเรียน | เนื่องจากนักเรียนต้องใช้เวลาว่างไปกับการเรียนกวดวิชา | ก่อให้เกิดความห่างเหินระหว่างเด็กกับผู้ปกครอง | มีผลต่อความอบอุ่นในครอบครัว
- ป่าชายเลนมีความสำคัญทางด้านอนุรักษ์พื้นที่ชายฝั่งทะเล | โดยทำหน้าที่เป็นปราการตามธรรมชาติ | ป้องกันลมพายุ | ป้องกันชายฝั่งไม่ให้ถูกกัดเซาะจากกระแสคลื่น | ช่วยในการรักษาคุณภาพสิ่งแวดล้อม โดยช่วยดักกรองของเสีย และขยะบริเวณชายฝั่งก่อนลงสู่ทะเล | นอกจากนี้ป่าชายเลนยังเป็นแหล่งกักเก็บคาร์บอนที่มีความสำคัญอีกด้วย
- ความรุนแรงในสังคมและความรุนแรงในครอบครัวมีความคล้ายคลึงกันที่เหยื่อ คือผู้อ่อนแอและมีสถานภาพต่ำกว่า เช่น สตรี เด็ก ฯลฯ | แต่ผลกระทบของความรุนแรงในครอบครัวรุนแรงและยาวไกลกว่าความรุนแรงในสังคม | เพราะนอกจากจะเป็นสาเหตุสำคัญของปัญหาครอบครัวแตกแยกและนำมาซึ่งปัญหาสังคมอีกมากมายแล้ว | ความรุนแรงในครอบครัวยังเป็นมรดกถ่ายทอดไปลูกหลานต่อ ๆ ในอนาคตอย่างไม่รู้จบสิ้น

- ในปัจจุบันความสำคัญในการใช้สุนัขล่าสัตว์ และช่วยหาอาหารนั้นได้ลดความสำคัญลงไป | แต่มนุษย์จะเลี้ยงสุนัขเพื่อใช้เป็นเพื่อน และใช้ประโยชน์อย่างอื่น ที่นอกเหนือไปจากการเฝ้าบ้าน การป้องกันขโมย ใช้ต้อนฝูงสัตว์ หรือเลี้ยงเพื่อการค้า | ในบางครั้งสุนัขที่ได้รับการฝึกหัด สามารถจะใช้ในงานสะกดรอยติดตามจับตัวผู้ร้าย ดมกลิ่นระเบิด ตรวจค้นหาสารเสพติด เป็นต้น
- มลพิษทางอากาศภายในอาคาร บ้านเรือน เป็นปัญหาหนึ่งที่สำคัญต่อมนุษย์ | เนื่องจากมนุษย์ใช้เวลาส่วนใหญ่อยู่ในอาคาร บ้านเรือน หรือ สถานที่ทำงาน | พบว่ามนุษย์โดยเฉพาะคนในเมืองใหญ่ใช้เวลาประมาณ 89% ในอาคาร บ้านเรือน | จึงไม่ใช่เรื่องที่น่าประหลาดใจ ถ้าการได้รับมลพิษในอากาศของประชากรในเขตเมืองจะเกิดขึ้นภายในอาคารมากกว่าที่เกิดขึ้นขณะดำเนินกิจกรรมอยู่ภายนอกอาคาร **|** จากข้อเท็จจริงดังกล่าวทำให้องค์กรพิทักษ์สิ่งแวดล้อมของประเทศสหรัฐอเมริกากำหนดให้ความเสี่ยงทางสุขภาพของมนุษย์อันเนื่องมาจากคุณภาพอากาศภายในอาคารอยู่ใน 5 อันดับแรกของความเสี่ยงทางสุขภาพเนื่องจากสภาวะแวดล้อมด้านต่าง ๆ

2.2. Deep learning models for NLP

There are various deep learning techniques which are used in NLP and achieve improved results on numerous tasks, including LSTM, self-attention, and others. There are two techniques used in this work and described as follows.

2.2.1. Long-short term memory (LSTM) [19]

Long-short term memory (LSTM) is one type of recurrent neural network (RNN). RNN can capture the temporal pattern from an input sequence; meanwhile, the traditional neural network assumes that each time step of an input sequence is independent of each other. Hence, RNN, which utilizes input and history of the previous sequence, gives the promising accuracy over a traditional neural network in many NLP tasks, such as named entity recognition, POS tagging, and semantic role labeling. Although, in theory, vanilla RNN can capture arbitrary long-term dependency information from an input sequence, its practical training process faces the vanishing gradient problem, in which the model cannot learn from a long sequence of input. Therefore, LSTM is proposed to avoid this problem by allowing the gradient to flow unchanged.

LSTM is composed of a cell, input gate, output gate, and forget gate, as shown in Figure 2. The cell takes charge of remembering the values from the previous sequence; meanwhiles, the gates control the flow of in and out information. The calculation of each gate is defined in Equations 1 to 3. Each function takes the input representation of the current position x_t and the hidden state of the previous position h_{t-1} as an input where W_*, U_*, and b_* denote as a weight matrix and bias of linear transformations; meanwhile, σ_g and σ_c represent the sigmoid and hyperbolic tangent function.

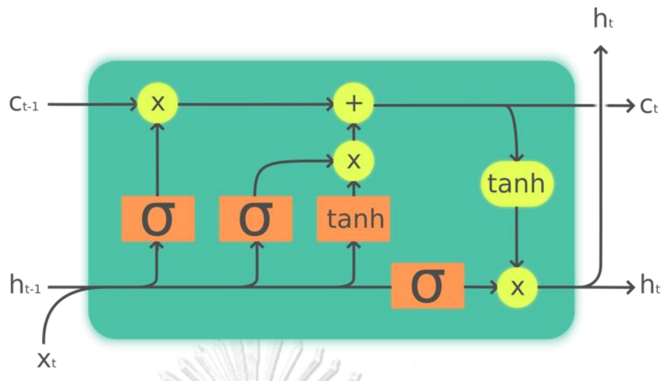


Figure 2. An architecture of LSTM [45].

$$f_t = \sigma_g\left(W_f x_t + U_f h_{t-1} + b_f\right) \tag{1}$$

$$i_t = \sigma_g(W_i x_t + U_i h_{t-1} + b_i) \tag{2}$$

$$\mathrm{o_t} = \sigma_\mathrm{g}(\mathrm{W_o x_t} + \mathrm{U_o h_{t-1}} + \mathrm{b_o}) \tag{3}$$

In each timestep, the model utilizes the output from these gates and generates two types of state. The first type is the cell state, which is used to calculate c_t, as shown in Equation 4. This cell state represents the memory from the previous sequence, where the forget gate determines whether the memory persists. Furthermore, by using forget gate combine with cell state without an input, LSTM can allow the gradient can flow over a long input sequence and get rid of the vanishing gradient problem. The second type is a hidden state, namely, the output of LSTM. The calculation of this hidden state h_t is shown in Equation 5, where the output gate and cell state go through element-wise multiplication to receive the hidden state.

$$c_t = f_t \oplus c_{t-1} + i_t \oplus \sigma_c(W_c x_t + U_c h_{t-1} + b_c) \tag{4}$$

$$h_t = o_t \oplus \sigma_c(c_t) \tag{5}$$

2.2.2. Self-attention [20]

Attention mechanism has become popular in many tasks of NLP, including machine translation, question answering, and others. Attention mechanism allows the model to focus selectively on each word regardless of the distance between words.

However, the attention mechanism relies on both input and output, which sometimes need the information from the previous sequence; therefore, the attention mechanism cannot be parallelized in the computation. To handle the mentioned problem, Self-attention mechanism is proposed. The only input sequence is considered in self-attention mechanism to learn the attention matrix; thus, the model can compute immediately without waiting to generate a previous output sequence and gain the ability of parallelization.

One of variant self-attention mechanism is Scaled-dot-product attention, of which inputs are keys K, query Q, and value V vectors and computed, as shown in Equation 6. The dot products of the key and query vectors are performed, and then scaled by dividing $\sqrt{d_k}$ where d_k is the number of dimensions of key and query vectors. After that, an attention matrix is acquired from applying the softmax function to the scaled matrix. Finally, the matrix product of the computed attention matrix and value vectors is calculated to be the output of Scaled-dot-product attention.

$$\text{ScaledDotProductAttention}(K, Q, V) = \text{Softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{6}$$

2.3. Semi-supervised learning

While deep neural networks achieve a satisfying accuracy over traditional methods, the model requires massive labeled data to generalize. Especially in NLP, an annotation of each task requires specialized knowledge from linguists; hence, receiving labeled data is costly and time-consuming. While suffering from the scarcity of labeled data, unlabeled data is utilized in the training process to improve the model.

There are many proposed semi-supervised learning approaches [30], which achieves successful results. Self-training [46, 47] is the simplest and earliest approach. As the name implied, the model, which trained with labeled data, is used to predict unlabeled data, and then the predicted confident data are used for training until there is no confident data.

However, training with self-training, the model cannot correct and magnify their error comes from wrong confident data. Therefore, multi-view training approaches [35, 48-50] are proposed to train different models with different views of data. In the first place, each model in multi-view training is learned separately, which take expensive computation. Ruder S. et al. [35] proposed multi-task tri-training to use with deep models for NLP tasks. Inspired by multi-task learning, this semi-supervised approach train three models which share the intermediate representation.

Recently, instead of multi-view training only with the full model, Cross-view training (CVT) [33] trains auxiliary views, which are generated from the restricted input, to match the primary view, which comes from the full view of input. Due to the lower computation and better accuracy of CVT, this method is selected as a semi-supervised learning algorithm to improve the representations of the proposed model. The detail of CVT is described in the following subsection.

2.3.1. Cross-view training (CVT) [33]

CVT is a semi-supervised learning technique whose goal is to improve the model representation using a combination of labeled and unlabeled data. During training, the model is trained alternately with one mini-batch of labeled data and B mini-batches of unlabeled data.

Labeled data are input into the model to calculate the standard supervised loss for each mini-batch and the model weights are updated regularly. Meanwhile, each mini-batch of unlabeled data is selected randomly from the pool of all unlabeled data; the model computes the loss for CVT from the mini-batch of unlabeled data. This CVT loss is used to train auxiliary prediction modules, which see restricted views of the input, to match the output of the primary prediction module, which is the full model that sees all the input. Meanwhile, the auxiliary prediction modules share the same intermediate representation with the primary prediction module. Hence, the intermediate representation of the model is improved through this process.

Similar to the previous work, we also apply CVT to a sequence tagging task. However, our model is composed of self-attention and CRF modules, which were not included in the sequence tagging model in [33]. The previous CVT was conducted on an LSTM using the concepts of forward and backward paths, which are not intuitively acquired by the self-attention model [20]. Moreover, the output used to calculate CVT loss was generated by the softmax function, which does not operate with CRF. Thus, in our study, both the primary and auxiliary prediction modules needed to be constructed differently from the original ones.

Similar to the previous work, we also apply CVT to a sequence tagging task. However, our model is composed of self-attention and CRF modules, which were not included in the sequence tagging model in [33]. The previous CVT was conducted on an LSTM using the concepts of

forward and backward paths, which are not intuitively acquired by the self-attention model [20]. attending all words of an input at the same time.

Moreover, the output used to calculate CVT loss was generated by the softmax function, which does not operate with CRF. Thus, in our study, it is necessary for both the primary and auxiliary prediction modules to be constructed differently from the original modules.

2.4. Pre-trained language model

Employing unlabeled data, many approaches in NLP tasks use the concept of pre-train to create representations from the massive unlabeled text. Pre-trained word vectors [37] are firstly proposed to generate dense word representations from tremendous data instead of sparse representations, such as one-hot vectors or TF-IDF. These pre-trained word vectors are included as the input representation to improve the performance of a model. However, utilizing pre-trained word vectors do not regard of context around the words; thus, a pre-trained language model, which is trained from enormous unlabeled data, is presented to acquire contextual word representations [51, 52].

The successful pretraining objectives are generally classified into two types, autoregressive (AR) and autoencoder (AE). AR are trained for estimating the probability distribution of the next word given the previous sequence; on the other word, AR encodes only uni-directional context. Meanwhile, AE intends to predict the original text from the corrupted text; therefore, AE can consider a bi-directional context from the corrupted text.

There are various pre-trained language models proposed to utilize enormous unlabeled data. Peter M. E. et al. [36] proposed ELMo, which exploits Bi-LSTM to extract the contextual information from the text. ELMo is learned using the concept of autoregressive where the word representation is generated from the forward and backward paths of Bi-LSTM.

From the success of Transformer, OpenAI GPT [38] utilizes Transformer instead of Bi-LSTM to be a pre-trained language model. OpenAI GPT is also trained with autoregressive like ELMo. Utilizing both left and right context jointly, BERT [29] uses the concept of the masked language model to train Transformer in an autoencoder way. The words in input text are randomly dropped and fed to the model, and then the rest of the input text tries to restore the dropped words.

In this work, we utilize the output of ELMO as a part of the input vectors of the model. The reason that we choose ELMO instead of BERT is using the transformer approach requires a too high amount of memory to fit in our GPU.

2.4.1. Embedding from Language Model (ELMo) [36]

Word representation is an embedding of the word which is changed from discrete representation into dense and continuous representation. Before ELMo is invented, the word embedding is pre-trained in a context-independent way. Therefore, Peter M. E. et al. proposed the new way to embed the sequence of words to be the input vectors for target tasks.

This representation exploits two techniques to improve the capability of word vectors. First, sequences of characters in word are used as part of an input to improve robustness in spell error and comprehend an affix of words. Second, a model exploits the Bidirectional language model concept (BiLM) to learn the context of a sentence. By applying these two concepts, the representation gain the understanding of a complex characteristic of words and usability of the word in various linguistic context.

The architecture of the model is based on L layers of Bi-LSTM. The input of the model x_t is composed of context-independent word embedding and character-based representation which produced by CNN. In Equation 7, Each step t is embedded as a single vector R_t where $\vec{h}_{t,j}^{LM}$ is forward path output, $\overleftarrow{h}_{t,j}^{LM}$ is backward path output of layer j. The vector R_t comprise forward and backward path output from every L layer. As a result, ELMo representation or R_t acquire the advantage of different information provided by each layer.

$$\mathrm{R}_t = \{x_k^{LM}, \vec{h}_{t,j}^{LM}, \overleftarrow{h}_{t,j}^{LM} \mid j = 1, \dots, L\} \tag{7}$$

In the pretraining process, the model is taught by the autoregressive concept or next-word prediction. In a forward path, each token x_k is predicted from $\{x_t | t = 1, \dots, k-1\}$ as shown in Equation 8. Meanwhile, in a backward path, each token x_k is predicted from $\{x_t | t = k+1, \dots, T\}$ as shown in Equation 9. The combination of both forward and backward paths is jointly formulated into the loss, as shown in Equation 10. Note that forward and backward path output is independently calculated, but its parameters of Bi-LSTM are sharing.

$$p(x_1, x_2, \dots, x_T) = \prod_{k=1}^{T} p(x_k | x_1, x_2, \dots, x_{k-1}) \tag{8}$$

$$p(x_1, x_2, \dots, x_T) = \prod_{k=1}^{T} p(x_k | x_{k+1}, x_{k+2}, \dots, x_T) \tag{9}$$

$$\sum_{k=1}^{T} \big(logp(x_k | x_1, x_2, \dots, x_{k-1}) + logp(x_k | x_{k+1}, x_{k+2}, \dots, x_T)\big) \tag{10}$$

2.5. Interpretation process

Although deep learning models make an advance in the NLP domain, there are always questions about how the models think when they operate their tasks. The prior works show that sometimes the model learns unwanted social biases from the training data [53] or understands superficial patterns to perform their tasks [54]. Therefore, the human tries to understand the models and gives feedback when they make a mistake.

To perform the interpretation at an instance level, many researchers provide various methods to explain the prediction in different aspects [53, 55-58]. There are also invented tools that ease to adapt to text classification [58, 59].

An intermediate layer interpretation is also studied in many fields. In computer vision, the activation of each convolutional layer is utilized to show how the features of the images are crafted [60]. Meanwhile, in NLP, the attention mechanism is used in a machine translation and gives the visualization to show how the input text related to the answer [4].

However, most methods are invented for specific models and difficult for users to implement for their models. Therefore, Allennlp Interpret [61] is presented to give an understanding of the effect of the input sequence in the prediction through the gradient. The methods in this toolkit are easy to apply and can be utilized with any deep model in NLP. Two groups of methods are implemented in this toolkit. The first group is gradient-based saliency maps [55, 62, 63]. The process of interpretation is inspired by a gradient technique [55]. The intuition of this method is to identify how the change in each feature affects the model. Simonyan K. et al. [55] shows that the score, which indicates the effect of the change in each feature, can be derived from a gradient of the feature. Therefore, the gradient of the feature can indicate the importance of the feature in the model. The second group is an adversarial attack which investigates the weakness of the model [56, 57].

In this work, we focus on a saliency maps technique in Allennlp interpret toolkit to find the importance of each n-gram and compare between Thai and English top most important features. The technique that we apply is simply calculated from the vanilla gradient of word embedding.

3. RELATED WORKS

In this chapter, three subsections are included. First, the prior works of Thai sentence segmentation are reviewed. The second section concerns English punctuation restoration due to the similarity between sentence segmentation and punctuation restoration. In the last section, Bi-LSTM-CRF, which is considered as the baseline model, is described.

3.1. Previous studies in Thai sentence segmentation

Due to the essential of space in sentence segmentation, previous works [12-14] from have focused on disambiguating whether a space functions as the sentence boundary. These works extract contextual features from words and POS around the space. Then, the obtained features around the corresponding space are input into traditional models to predict whether space is a sentence boundary. Moreover, to improve the model accuracy, Thai grammar rules [64, 65] are also integrated with the statistical models.

Although a space is usually considered essential as a sentence boundary marker, approximately 23% of the sentences end without a space character in one news domain corpus [27]. Hence, Zhou N. et al. [27] proposed a word sequence tagging CRF-based model in which all words can be considered as candidates for the sentence boundary. A space is considered as only one possible means of forming a sentence boundary. The CRF-based model [16], which is extracted from n-grams around the considered word, achieves a F1 score of 91.9%, which is approximately 10% higher than the F1 scores achieved by other models [12-14] on the Orchid dataset, as mentioned in [27]. Nararatwong R. et al. [66] extend this model using a POS-based word-splitting algorithm to increase identifiable POS tags, resulting in better model accuracy. Because the focus of this work is adjusting the POS as a postprocessing method, which is an input of the model instead of proposing a new sentence segmentation model, this work will not be considered in this paper.

In this work, we adopt the concept of word sequence tagging and compare it with two baselines: the CRF-based model with n-gram embedding, which is currently the state-of-the-art for Thai sentence segmentation, and the Bi-LSTM-CRF model, which is currently the deep learning state-of-the-art approach for sequence tagging.

3.2. Previous studies of English punctuation restoration

Most languages use a symbol that functions as a sentence boundary; however, a few do not use sentence markers including Thai, Lao, and Myanmar. Thus, few studies have investigated sentence segmentation in raw text. However, studies on sentence segmentation, which is sometimes called sentence boundary detection, are still found in the speech recognition field [67]. The typical input to the speech recognition model is simply a stream of words. If two sentences are spoken back to back, by default, a recognition engine will treat it as one sentence. Thus, sentence boundary detection is also considered a punctuation restoration task in speech recognition because when the model attempts to restore the period in the text, the sentence boundary position will also be defined.

Punctuation restoration not only provides a minimal syntactic structure for natural language processing, similar to sentence boundary detection but also dramatically improves the readability of transcripts. Therefore, punctuation restoration has been extensively studied. Many approaches have been proposed for punctuation restoration that uses different features, such as audio and textual features. Moreover, punctuation restoration is also considered to be a machine learning problem, similar to word sequence tagging and machine translation.

A combination of audio and textual features was utilized in [68-70] to predict and restore punctuation, including pitch, intensity, and pause duration, between words. We ignore these features in our experiment because our main task—Thai sentence segmentation— does not include audio features.

Focusing only on textual features, there are two main approaches, namely, word sequence tagging and machine translation. For the machine translation approach, punctuation is treated as just another type of token that needs to be recovered and included in the output. The methods in [71-73] restore punctuation by translating from unpunctuated text to punctuated text. However, our main task, sentence segmentation, is an upstream task in text processing, unlike punctuation restoration, which is considered a downstream task. Therefore, the task needs to operate rapidly; consequently, we focus only on the sequence tagging model, which is less complex than the machine translation model.

In addition to those machine translation tasks, both traditional approaches and deep learning approaches must solve a word sequence tagging problem. Of the traditional approaches, contextual features around the considered word were used to predict following punctuation in the n-gram [74] and CRF model approaches [75, 76]. Meanwhile, in the deep learning approaches, a deep convolutional neural network [77], T-LSTM (Textual-LSTM) [70] and a bidirectional LSTM model with an attention mechanism, called T-BRNN [78], have been adopted to predict a punctuation sequence from the word sequence. T-BRNN [78] was proposed to solve the task as a word-sequence tagging problem, and it is currently the best model that uses the word sequence tagging approach. Tilk O. et al. [78] also proposed a variant named T-BRNN-pre, which integrates pretrained word vectors to improve the accuracy. Meanwhile, Kim S. [79] proposed to insert a multi-head attention module between Bi-LSTM to improve the model accuracy called Deep Recurrent Neural Network with Layer-Wise Multi-head Attentions (DRNN-LWMA). The model is currently the best model that uses the word sequence tagging approach. Yi J. et al. [22] adopt the pre-training method to improve the accuracy of the model. This work selects the Bi-LSTM-CRF as a backbone model. Meanwhile, the input words are embedded by a pre-trained BERT before feeding to the backbone model. Because POS tags are helpful for this task, POS tags are always fed to the model in this task. However, POS tags are generated from the machine learning model which is usually error-prone. Thus, Yi J. et al. [22] also adopted adversarial transfer learning to imitate the effect of an error from predicted POS tags. As a result, their proposed model gains a 9.2% F1 score improvement compared to prior works.

To demonstrate that our model is generalizable to other languages, we compare it with other punctuation restoration models, including T-LSTM, T-BRNN, T-BRNN-pre, DRNN-LWMA, and DRNN-LWMA-pre. These models adopt a word sequence tagging approach and do not utilize any prosodic or audio features.

3.3. Bi-directional LSTM with CRF (Bi-LSTM-CRF) [21]

From the success of bidirectional LSTM in NLP, Bi-LSTM-CRF is proposed as a sequence tagging model for NLP tasks, such as Name entity recognition, POS tagging, and chunking. The model gain benefit of both deep learning and traditional approaches. By using Bi-LSTM, the model is capable of efficiently utilizing both past and future input features. Meanwhile, adapting CRF, the model can use sentence-level tag information for the decoding process.

The architecture of Bi-LSTM-CRF is illustrated in Figure 3. The model is composed of two main modules. The first module is bi-directional LSTM, which extract the context vectors from the forward and backward paths. The second module is CRF, which used the output of bi-directional LSTM to calculate emission and transmission scores and then use the Viterbi algorithm to decode the best sequence tag for an input.

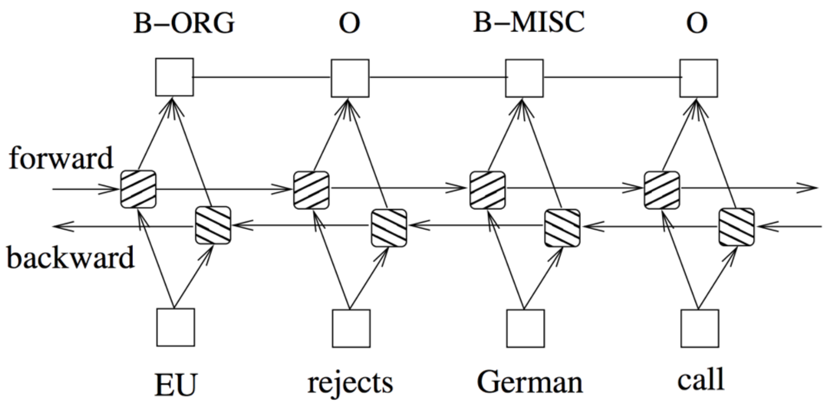


Figure 3. An architecture of Bi-LSTM-CRF [21].

4. METHODOLOGY

In this chapter, we describe our methodology in two subsections. The first subsection specifies the model architecture and the details of each module. Meanwhile, the second subsection expounds on how the model is trained with unlabeled data through pre-trained concept and the modified CVT.

4.1. Model architecture

In this work, the model predicts the tags $\vec{y} = [y_1, y_2, \dots, y_N], \forall y \in Y$ for the tokens in a word sequence $\vec{x} = [x_1, x_2, \dots, x_N]$ where N is the sequence size and x_t, y_t denote the token and its tag at timestep t, respectively. The word sequence is fed into the model at the same time to provide sequential information of the input text to the model. Each token x_t consists of a word, its POS, and its type. There are five defined word types: English, Thai, punctuation, digits, and spaces.

The tag set Y is populated based on the considered task. In Thai sentence segmentation, the assigned tags are sb and nsb; sb denotes that the corresponding word is a sentence boundary considered as the beginning of a sentence, while and nsb denotes that the word is not a sentence boundary. Meanwhile, there are four tags in the punctuation restoration task. Words not followed by any punctuation are tagged with O. Words that are followed by a period ". ", comma "," or question mark "?" are tagged to $period$, $comma$, and $question$, respectively.

Our model architecture is based on Bi-LSTM-CRF, as shown in Figure 4. The model is divided into three modules. The first, low-level module, consists of two separate structures: local and distant structures. This module is designed to utilize an unlabeled dataset via cross-view training, which will be described in Section 4.2.2. The module gives the two separate outputs from two structures, which are only concatenated to feed to the next module. Meanwhile, the second, high-level module, contains a sequence of stacked Bi-LSTM and self-attention layers, which helps the model learn the context from the whole word sequence. The final module, the prediction module, is responsible for predicting the tags $\vec{y}$. Each module is described more completely in the next three subsections.

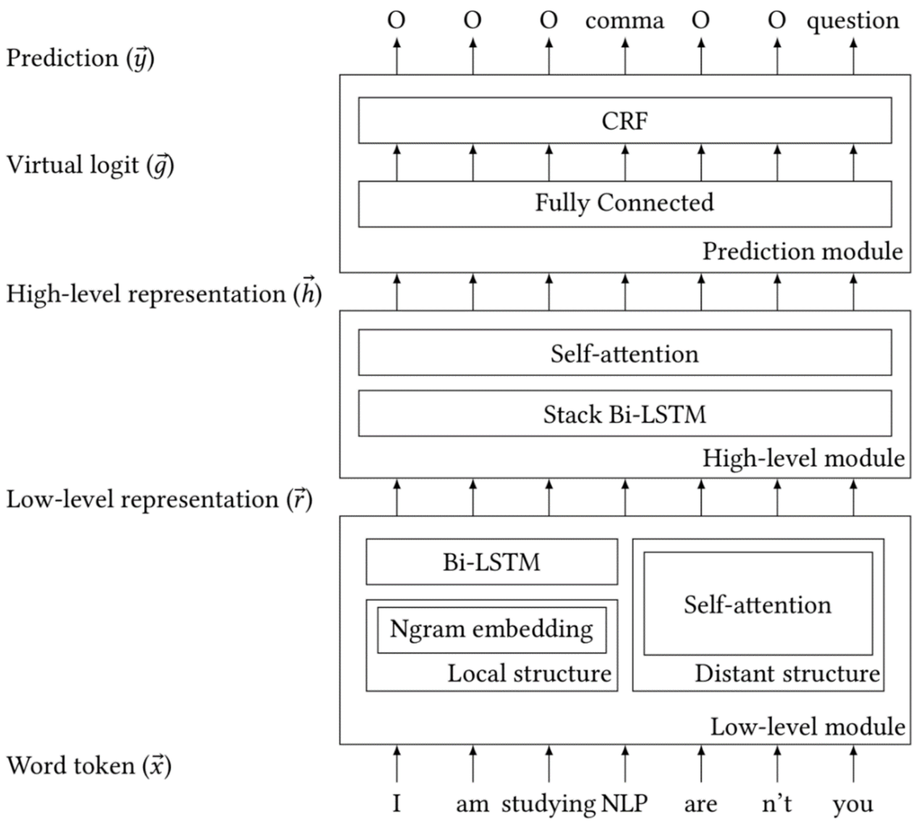


Figure 4. Model architecture that integrates local and distant representation. This model is composed of three main modules: a low-level module, a high-level module, and a prediction module. In the low-level module, two structures (local and distant) are responsible for extracting different features.

4.1.1. Low-level module

A sequence of word tokens is input into the low-level module. The input tokens pass through two structures. The first structure generates a sequence of local representation vectors $R_{local} = [\vec{r}_{1,\,local}, \vec{r}_{2,\,local}, \dots, \vec{r}_{N,\,local}]$, which is embedding vectors of n-gram features. After obtaining a sequence of representation vectors, the local representation vectors are fed to the Bi-LSTM to obtain the recurrent representation vectors. $R_{recurrent} = [\vec{r}_{1,\,recurrent}, \vec{r}_{2,\,recurrent}, \dots, \vec{r}_{N,\,recurrent}]$, as shown in Equation 11:

$$\mathrm{R_{recurrent}} = \mathrm{BiLSTM}(\mathrm{R_{local}}) \tag{11}$$

The second structure generates low-level distant representation vectors $R_{distant} = [\vec{r}_{1,\,distant}, \vec{r}_{2,\,distant}, \dots, \vec{r}_{N,\,distant}]$, which produced by Self-attention. Then, the recurrent and distant representation vectors are concatenated to form the low-level representation vector $\mathrm{R} =$

$[\vec{r}_1, \vec{r}_2, \dots, r_N]$, as shown in Equation 12, where $\oplus$ represents a concatenation between two vectors:

$$\vec{\mathrm{r}}_{\mathrm{t}} = \vec{\mathrm{r}}_{\mathrm{t,recurrent}} \oplus \vec{\mathrm{r}}_{\mathrm{t,distant}} \tag{12}$$

4.1.1.1. Local structure

This structure is shown as the left submodule of the low-level module in Figure 4. It extracts the local representation vectors R_{local}. Its input tokens are used to create n-gram tokens, which are unigrams x_a, bigrams (x_a, x_b), and trigrams (x_a, x_b, x_c). Each n-gram token is represented as an embedding vector, which is classified as a unigram embedding vector $\vec{e}_{uni}$, a bigram embedding vector $\vec{e}_{bi}$ or a trigram embedding vector $\vec{e}_{tri}$. Each vector $\vec{e}_{gram}$ is mapped from a token by gram embedding $Embedding_{gram}(x)$, which is a concatenated vector of the word embedding $Word_{gram}(x)$, POS embedding $POS_{gram}(x)$ and type embedding $Type_{gram}(x)$, as shown in Equation 9:

$$\mathrm{Embedding}_{\mathrm{gram}}(\mathrm{x}) = \mathrm{W}_{\mathrm{gram}}(\mathrm{x}) \oplus \mathrm{POS}_{\mathrm{gram}}(\mathrm{x}) \oplus \mathrm{Type}_{\mathrm{gram}}(\mathrm{x}) \tag{13}$$

Unigram embedding vector $\mathbf{Embedding}_{uni}(x)$ is included along with contextual pretrained vector $ELMo_{\mathrm{uni}}(\mathrm{x})$ from ELMo, as shown in Equation 14:

$$\mathrm{Embedding}_{\mathrm{uni}}(\mathrm{x}) = \mathrm{W}_{\mathrm{uni}}(\mathrm{x}) \oplus \mathrm{POS}_{\mathrm{uni}}(\mathrm{x}) \oplus \mathrm{Type}_{\mathrm{uni}}(\mathrm{x}) \oplus ELMo_{\mathrm{uni}}(\mathrm{x}) \tag{14}$$

Each n-gram token at timestep t is generated by the previous, present and next token (x_{t-1}, x_t, x_{t+1}) and embedded into vectors as shown in Equations 15-17 for unigram, bigram, and trigram consecutively.

$$\vec{e}_{t,uni} = \mathrm{Embedding}_{uni}(x_t) \tag{15}$$

$$\vec{e}_{t,bi} = \mathrm{Embedding}_{bi}(x_{t-1}, x_t) \tag{16}$$

$$\vec{e}_{t,tri} = \mathrm{Embedding}_{tri}(x_{t-1}, x_t, x_{t+1}) \tag{17}$$

At each timestep t, a local representation vector $\vec{r}_{t,local}$ is combined from the n-gram embedding vectors generated from the context around x_t. A combination of embedding vectors, which is used to construct a local representation vector, is shown in Equation 18. A combination consists of the unigram, bigram, and trigram embedding vectors at timesteps t-1, t and t+1 and it is a concatenation of all the embedding vectors:

$$\vec{r}_{t,local} = \vec{e}_{t-1,uni} \oplus \vec{e}_{t,uni} \oplus \vec{e}_{t+1,uni} \oplus \vec{e}_{t-1,bi} \oplus \vec{e}_{t,bi} \oplus \vec{e}_{t+1,bi} \oplus \vec{e}_{t-1,tri} \oplus \vec{e}_{t,tri} \oplus \vec{e}_{t+1,tri} \tag{18}$$

4.1.1.2. Distant structure

The distant structure, which is a self-attention module, is shown in Figure 4 on the right side of the low-level module. The structure extracts low-level distant representation vectors $R_{distant}$ from a sequence of unigram embedding vectors E_{uni}, as shown in Equation 19. In this case, the self-attention module is a scaled dot-product attention, where key, query, and value vectors are the linear projections of the unigram embedding vectors shown in Figure 5. The linear transformations for key, query, and value are learned separately and updated in the model through backpropagation. The output vector, which is the scaled dot-product attention at each timestep, is concatenated with the input vector $\vec{e}_{t,uni}$ and projected by a linear transformation. That projected vector is the output vector of a self-attention module, which is a low-level distant representation vector.

$$R_{distant} = \text{SelfAttention}(E_{uni}) \tag{19}$$

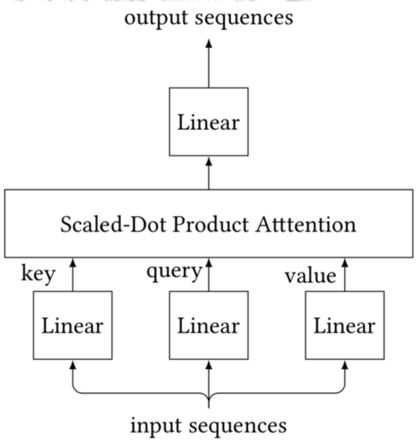


Figure 5. The architecture of a self-attention module.
This module mainly contains Scaled-Dot Product Attention, which requires three inputs: Key, Query, and Value. Those inputs are generated from the same input sequence but projected by different linear transformations.

4.1.2. High-level module

The low-level representation vectors R are used as the input for this module, which outputs the high-level representation vectors H whose calculation is shown in Equation 20. The high-level module, as shown in Figure 4, is composed of a stacked bidirectional LSTM and a self-attention module. A stacked bidirectional LSTM contains K layers of bidirectional LSTMs in which the output from the previous bidirectional LSTM layer is the input of the next bidirectional LSTM layer. The self-attention part of this structure is the same as that in the low-level distant structure. The self-attention module helps to generate the high-level distant representation vectors that are output by the high-level module.

$$H = \text{SelfAttention}\big(\text{StackBiLSTM}(R)\big) \tag{20}$$

4.1.3. Prediction module

The prediction module is the last module. It includes two layers: a fully connected layer and a CRF layer. In the fully connected layer, the output vectors from the high-level module are projected by a linear transformation as shown in Equation 21. The purpose of this layer is to create the virtual logit vectors $G = [\vec{g}_1, \vec{g}_2, \dots, \vec{g}_N]$, which represent the probability distribution for CVT, as discussed in Section 4.2.2.2. Therefore, the number of dimensions of logits equals the number of possible tags in each task:

$$\vec{g}_t = \text{NN}\big(\vec{h}_t\big) \tag{21}$$

The CRF layer is responsible for predicting the tag y_t of a token at each timestep, as shown in Equation 22. The layer receives a sequence of virtual logit vectors (G) as input and then decodes them to a sequence of tags $\vec{y}$ using the Viterbi algorithm.

$$\vec{y} = \text{CRF}(G) \tag{22}$$

4.2. Training process

To train the sentence segmentation model, the process is split into two steps. First, the language model (ELMo) is trained with unlabeled data. The pre-trained language model is treated as a part of input vectors for the model. The second step is to train the model with CVT for the sentence segmentation problem. Each mentioned step is described in the following subsections.

4.2.1. Pre-trained language model

In this process, the pre-trained language models come from different sources depending on the language. In UGWC, we pre-train ELMo with original implementation on our unlabeled dataset. Meanwhile, in English punctuation restoration, the original ELMo of English is applied in this case.

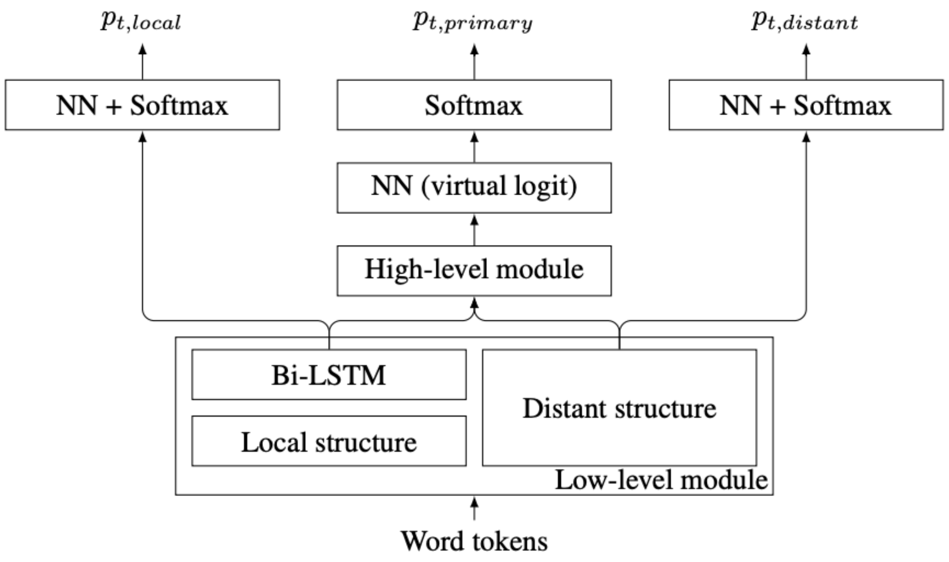


Figure 6. Unlabelled data utilization.
two auxiliary predictions ($\vec{p}_{t,local}$ and $\vec{p}_{t,distant}$) are obtained from the local and distant structures in the low-level module. The primary prediction $\vec{p}_{t,primary}$ is obtained from the virtual logit vector $\vec{g}_t$.

4.2.2. Cross-view training (CVT)

In this step, CVT is adopted to train the sentence segmentation with labeled and unlabelled data simultaneously in a semi-supervised learning way. The weights of the model are initialized normally, except for the low-level module, which is initialized from the pre-trained language model, as described in Section 4.2.1.

As discussed in Section 2.3.1, CVT requires primary and auxiliary prediction modules for training with unlabeled data to improve the representation. Thus, we construct both types of prediction modules for our model. The flow of unlabeled data, which is processed to obtain a prediction by each module, is shown in Figure 6. The output of each prediction module is transformed into the probability distribution of each class by the softmax function and then used to calculate $Loss_{CVT}$, as shown in Equation 23.

$$Loss_{CVT} = \frac{1}{|D|}\sum_{t \in D} D_{\mathrm{KL}}(\vec{p}_{t,primary}, \vec{p}_{t,local}) + D_{\mathrm{KL}}(\vec{p}_{t,primary}, \vec{p}_{t,distant}) \qquad (23)$$

The $Loss_{CVT}$ value is based on the Kullback–Leibler divergence (KL divergence) between the probability distribution of the primary $\vec{p}_{t,primary}$ output and those of two auxiliary modules,

$\vec{p}_{t,local}$ and $\vec{p}_{t,distant}$, where $t \in [1, \dots, N]$. The KL divergence at each timestep is averaged when the timesteps are dropped timesteps D.

The details of the primary and auxiliary prediction modules, which are used in the $Loss_{CVT}$ calculation, are described in the following subsections.

4.2.2.1. Primary prediction module

In [33], the output of the primary prediction module is acquired from the last layer and used to predict tags. However, our model uses a CRF layer to decode the tags instead of the softmax function. Thus, in semi-supervised learning, the probability distribution of the primary prediction module should be acquired from the CRF layer. However, the Viterbi algorithm, which is used for decoding, gives only the best combination for the prediction, but does not provide the probability distribution. Normally, the distribution from the CRF is calculated by a forward-backward algorithm [80] which is time consuming. To reduce the training time, the probability distribution of the primary prediction module $\vec{p}_{t,primary}$ is obtained from the output of the softmax function, whose input is a virtual logit vector $\vec{g}_t$, as shown in Equation 24.

$$\vec{p}_{t,primary} = \mathrm{Softmax}(\vec{g}_t) \tag{24}$$

4.2.2.2. Auxiliary prediction module

Two auxiliary views are included to improve the model. The first view is generated from a recurrent representation vector $r_{t,recurrent}$ to acquire the local probability distribution $\vec{p}_{t,local}$, where $t \in [1, \dots, N]$. The second view is generated from the low-level distant representation vectors $r_{t,distant}$ to acquire the probability distribution of a distant structure in the low-level module $\vec{p}_{t,distant}$ where $t \in [1, \dots, N]$. By generating the views from these representation vectors separately, the local and distant structures in the low-level module can improve equally.

Although both representation vectors are used separately to create auxiliary views, the input of each structure is still not restricted, unlike [6], where the input is restricted to only previous or future tokens. Because BERT, which is trained by the masked language model, outperforms OpenAI GPT, which uses an autoregressive approach for training as reported in [29], we adopt the concept of the masked language model [81] to obtain both auxiliary views. This approach allows the representation to fuse the left and the right context, which results in a better representation. By using the masked language model, some tokens at each timestep are randomly dropped and denoted as removed tokens <REMOVED>; then, the remaining tokens are used to obtain auxiliary predictions in the dropped timesteps $D = [\, d \in N \mid where\ d\ is\ a\ dropped\ timestep \,]$, as shown in Figure 7. The details of both auxiliary prediction modules are described below.

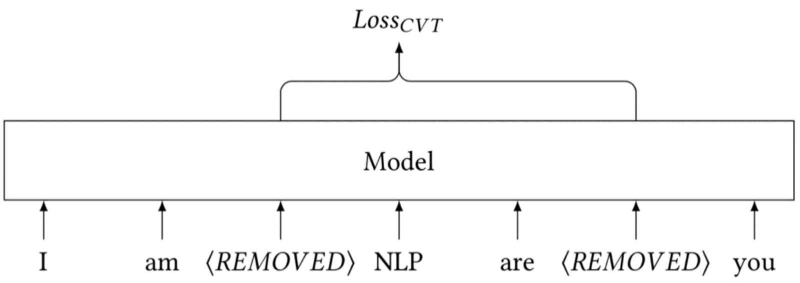


Figure 7. An example of performing a masked language model. Words are dropped (denoted as <REMOVED>) randomly. Those positions are used to calculate $Loss_{CVT}$ and update the auxiliary prediction modules to improve the model.

4.2.2.3. Local auxiliary module

For recurrent representation vectors, if one of the tokens is dropped, the related n-gram tokens that include the dropped tokens will also be dropped. For example, if (x_t) is dropped, (x_{t-1}, x_t) and (x_t, x_{t+1}) will also be dropped as removed tokens in the case of a bigram. The remaining n-gram tokens are then used to obtain the recurrent representation vectors at the dropped timesteps. Then, the vectors are provided as an input to the softmax function to obtain the probability distribution of the first auxiliary prediction module, as shown in Equation 25.

$$\vec{p}_{d,local} = \text{Softmax}\left(\text{NN}(\vec{r}_{d,recurrent})\right) \quad (25)$$

4.2.2.4. Distant auxiliary module

In the other auxiliary prediction module, a sequence of the low-level distant representation vectors is generated and some tokens are dropped. This sequence of vectors is also input into the Softmax function, just as in the first auxiliary prediction module, and the output is another probability distribution, which is the second auxiliary prediction, as shown in Equation 26.

$$\vec{p}_{d,distant} = \text{Softmax}\left(\text{NN}(\vec{r}_{d,distant})\right) \quad (26)$$

4.3. Interpretation process

In this section, we discuss our method to interpret our model decision process by identifying important n-gram features. These selected features can be compared to the human annotations or the label of the dataset in order to further analyze the model. The process is inspired by a vanilla gradient technique, which is proposed in [55].

Based on Equation 27, nine types of features, which are unigram, bigram, and trigram at timesteps $t-1$, t and $t+1$, are used as input features for our model. Thus, the gradient of these nine types of n-gram features are calculated. The gradient $\nabla\vec{e}_{t,uni}(\mathrm{x})$ is the derivative of the output of the model y with respect to $\vec{e}_{t,gram}(x)$ where $gram \in \{uni, bi, tri\}$ is the token type and $pos \in \{-1, 0, +1\}$ is the relative position from the current timestep, as shown in Equation 27.

$$\nabla\vec{e}_{t,gram}(\mathrm{x}) = \frac{\partial y}{\partial\vec{e}_{t,gram}(x)} \tag{27}$$

After that, the score of each feature $u_{pos,gram}$ is calculated, as shown in Equation 28. The score is the summation of all the gradients of tokens $x_{t+pos,gram}$ that are $u_{pos,gram}$ over all timesteps $t \in \{1, \dots, N\}$ in all documents D. This score is used to measure the importance of each feature. When the summed gradient of the feature is high, that feature is important in contributing to the model's predictions. On the other hand, if the summed gradient of a feature is low, that feature is not necessary for the model's prediction.

$$Score\left(u_{pos,gram}\right) = \sum^{D}\sum_{t=1}^{N}\nabla\vec{e}_{t+\mathrm{pos},gram}(x) \tag{28}$$

$$where\ x_{t+pos,gram} = u_{pos,gram}$$

To compare the model to humans, two lists of tokens are created from the computed score and labels. Then, the intersection of both lists is used to compare the similarity. The first list $\mathbf{T}_{pos,gram}^{(model)}$ includes the top 500 features which have the highest computed score $Score\left(u_{pos,gram}\right)$ from the gradient. The second list $\mathbf{T}_{pos,gram}^{(label)}$ contains of top 500 features with the highest frequency as a sentence boundary in the training set. After that, the features in the intersection between two lists are counted, as shown in Equation 29, where $\|\mathrm{A}\|$ is a number of instances in a set $\mathbf{A}$. The counted number $\mathrm{Count}(pos, gram)$ will be used for further analysis in Section 5.4.2.3.

$$\text{Count}(pos, gram) = \left\| \mathbf{T}_{pos,gram}^{(model)} \cap \mathbf{T}_{pos,gram}^{(label)} \right\| \tag{29}$$

5. EXPERIMENTS AND RESULTS

In this section is described conducted experiments. We have already explored the effect of local and distant representations. Moreover, the impact of CVT has already been investigated. However, there is no experiment on the pre-training process.

5.1. Dataset

Three datasets are used in the experiments as described in the following subsections. We use two datasets for Thai sentence segmentation, and the third dataset is used for English punctuation restoration. The statistics of the preprocessed data are shown in Table 1, including the number of sequences and the number of vocabulary words in each dataset.

Table 1. The number of passages and vocabulary words in each dataset.

Dataset	Statistics	# passage	# words	# vocab
Orchid (Thai)	Label data	3,427	685,319	17,047
UGWC (Thai)	Label data	48,374	1,242,118	46,463
	Unlabeled data	96,777	40,431,319	81.932
	Labeled + Unlabeled	145,151	41,673,437	109,415
IWSLT (English)	Label data	12,803	2,547,797	47,532
	Unlabeled data	8,449	1,678,167	35,931
	Labeled + Unlabeled	21,252	4,225,964	56,762

The labeled and unlabeled data are separately counted and shown in the rows.
Note: There are no unlabeled data in the Orchid dataset due to the lack of the same word segmentation and POS tag set.

We also calculate the average number of words per passage in the unlabeled data that do not appear in the labeled data, as shown in Table 2.

Table 2. Average number of words per passage that exist in the unlabeled data but not in the labeled data.

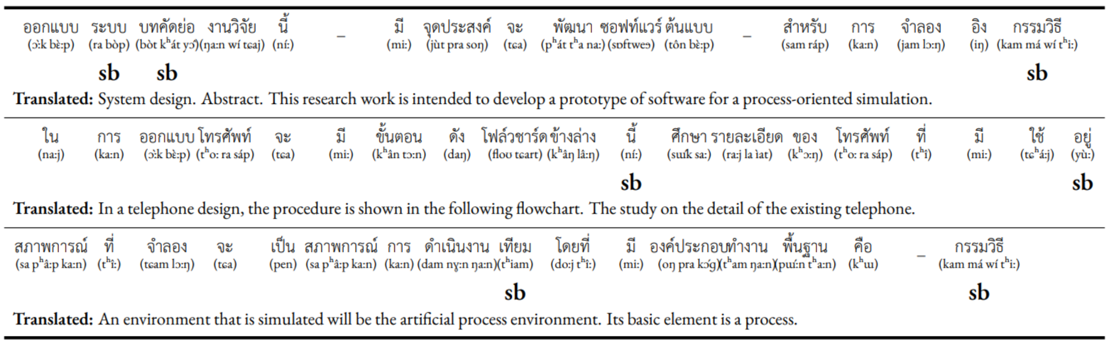
Dataset	# words
UGWC	0.650
IWSLT	1.092

5.1.1. Thai sentence segmentation

In this subsection, two Thai sentence segmentation datasets are described. The first dataset is Orchid and the second one is UGWC.

5.1.1.1. Orchid [18]

This dataset is a Thai part-of-speech-tagged dataset containing 10,864 sentences. In the corpus, text was separated into paragraphs, sentences, and words hierarchically by linguists. Each word was also manually assigned a POS by linguists. These data include no unlabeled data with the same word segmentation and POS tag set. Hence, we do not execute semi-supervised learning or pretraining process on this dataset. Sample paragraphs in this dataset is shown in Figure 8.



ออกแบบ	ระบบ	บทคัดย่อ	งานวิจัย	นี้	_	มี	จุดประสงค์	จะ	พัฒนา	ซอฟท์แวร์	ต้นแบบ	_	สำหรับ	การ	จำลอง	อิง	กรรมวิธี
(ɔ̀:k bɛ̀:p)	(ra bòp)	(bòt k^{h}át yɔ̂)	(ŋa:n wí tɕaj)	(ní:)		(mi:)	(jùt pra soŋ)	(tɕa)	(p^{h}át t^{h}a na:)	(sɒftweə)	(tôn bɛ̀:p)		(sam ráp)	(ka:n)	(jam lɔ:ŋ)	(iŋ)	(kam má wí t^{h}i:)
	sb	**sb**															**sb**

Translated: System design. Abstract. This research work is intended to develop a prototype of software for a process-oriented simulation.

ใน	การ	ออกแบบ	โทรศัพท์	จะ	มี	ขั้นตอน	ดัง	โฟล์วชาร์ต	ข้างล่าง	นี้	ศึกษา	รายละเอียด	ของ	โทรศัพท์	ที่	มี	ใช้	อยู่
(na:j)	(ka:n)	(ɔ̀:k bɛ̀:p)	(t^{h}o: ra sáp)	(tɕa)	(mi:)	(k^{h}ân tɔ:n)	(daŋ)	(floʊ tɕart)	(k^{h}âŋ lâ:ŋ)	(ní:)	(sɯ̀k sa:)	(ra:j la ìat)	(k^{h}ɔ:ŋ)	(t^{h}o: ra sáp)	(t^{h}î)	(mi:)	(tɕ^{h}á:j)	(yù:)
										sb								**sb**

Translated: In a telephone design, the procedure is shown in the following flowchart. The study on the detail of the existing telephone.

สภาพการณ์	ที่	จำลอง	จะ	เป็น	สภาพการณ์	การ	ดำเนินงาน	เทียม	โดยที่	มี	องค์ประกอบ	ทำงาน	พื้นฐาน	คือ	_	กรรมวิธี
(sa p^{h}â:p ka:n)	(t^{h}î:)	(tɕam lɔ:ŋ)	(tɕa)	(pen)	(sa p^{h}â:p ka:n)	(ka:n)	(dam nɣ:n ŋa:n)	(t^{h}iam)	(do:j t^{h}î:)	(mi:)	(oŋ pra kɔ̀g)	(t^{h}am ŋa:n)	(pɯ́:n t^{h}a:n)	(k^{h}ɯ)		(kam má wí t^{h}i:)
								sb								**sb**

Translated: An environment that is simulated will be the artificial process environment. Its basic element is a process.

Figure 8. Labeled paragraphs in Orchid dataset. Here, sb represents a sentence boundary.

The text in this dataset is collected from publications in the NECTEC annual conference. Therefore, the text is structured as formal document. The documents usually contain similar word phrases or function words [82] to convey the lexical meaning such as "For example" or "Introduction". Moreover, the named entities like the name of organization are often found in the text. As a result, the text sometimes contains duplicate word phrases or content words [82].

Our data preprocessing on the ORCHID corpus was similar to that in [27]: all the comments are removed, and the data are partitioned into 10 parts containing equal numbers of sentences to support 10-fold cross-validation. Each training set is split into one part used for validation and the rest is used for model training. Subsequently, all the words in each dataset are concatenated and then separated into sequences with 200 words per instance. Each sequence always begins with the first word of a sentence. If a sequence ends with an unfinished sentence, the next sequence starts with that complete sentence.

5.1.1.2. UGWC (User-Generated Web Content) [43]

This Thai dataset includes many types of labeled data useful in sentence segmentation tasks. The raw text was generated by users having conversations in the financial domain and was acquired mainly by crawling social sites. The labeled data for sentence segmentation were manually annotated by linguists using the definitions in [43]. Sample paragraph in this dataset is shown in Figure 9.

เมื่อ	กี้	สาย	ตัด	ไป	_	อยาก	ทราบ	ว่า	ได้	รับ	เรื่อง	ไว้	แล้ว	ยัง	ค่ะ
(mɯâ)	(kî:)	(sa:j)	(tàt)	(paj)		(jà:k)	(sâ:p)	(wâ:)	(dâj)	(ráb)	(rɯâŋ)	(wá:j)	(lɛ́:w)	(jaŋ)	(k^{h}â)
				sb	**sb**										**sb**

Translated: The connection is lost. I want to know if you get the request or not.

เปิดบัญชี	ครั้ง	แรก	_	ใช้	เงิน	เท่าไหร่	ค่ะ	_	ถ้า	เป็น	บัตร	_	debit	_	ธรรมดา
(pɤ̀:t ban tɕ^{h}i:)	(k^{h}táŋ)	(rɛ̂:k)		(tɕ^{h}áj)	(ŋɤn)	(tɕ^{h}āw ràj)	(k^{h}â)		(t^{h}â:)	(pen)	(bàt)				(t^{h}am ma da:)
		sb	**sb**												**sb**

Translated: Open an account for the first time. How much fee for the registration if I want a normal debit card?

ต้องการ	จะ	เปลี่ยน	เบอร์	ต้อง	ทำ	ยังไง	บ้าง	ครับ
(tɔ̂ŋ ka:n)	(tɕà)	(plìan)	(bɛ:)	(tɔ̂ŋ)	(t^{h}am)	(jaŋ ŋa:j)	(bâ:ŋ)	(k^{h}ráp)
			sb					**sb**

Translated: I want to change the number. What should I do?

Figure 9. Labeled paragraphs in UGWC dataset. Here, sb represents a sentence boundary

The text in this dataset is crawled from social media, which includes both conversations and news which are related to the financial domain. The text that is collected from conversations are normally formal, so the text always ends up with final particles “ครับ” and “คะ”. While, if the text is informal, the emoticon is often found in the text. In addition, the text from news contains a number of conjunctions to create the influence article.

At the time of this study, the dataset was extended from that in [43]; the data were collected from January 2017 to December 2017. The labeled dataset includes 48,374 passages. To support semi-supervised learning, the first 3 months of data (96,777 passages) are unlabeled.

Because the data stem from social media, some text exists that cannot be considered as part of any sentence, such as product links, symbols unrelated to sentences, and extra space between sentences. These portions were not originally annotated as sentences by the linguists. However, in this work, we treat these portions as individual sentences and tag the first word of each fraction as the sentence boundary.

For evaluation purposes, the collection of passages in this dataset is based on 5-fold cross-validation, similar to the previous work [43]. The passages are treated as input sequences for the model. For each passage, word segmentation and POS tagging are processed by the custom models from this dataset.

5.1.2. English Punctuation Restoration

For an English punctuation restoration experiment, a common dataset called IWSLT is performed. The detail of dataset is described in the following subsection.

5.1.2.1. IWSLT [83]

We adopted this English-language dataset to enable comparisons with models intended for other languages. The dataset is composed of TED talk transcripts. To compare our model with those of previous works, we selected the training dataset for the machine translation track in IWSLT2012 and separated it into training and validation sets containing 2.1 million and 295 thousand words, respectively. The testing dataset is the IWSLT2011 reference set, which contains 13 thousand words. To acquire unlabeled data for semi-supervised learning, we adopted the IWSLT2016 machine translation track training data; duplicate talks that also appear in IWSLT2012 are discarded.

The data preprocessing follows the process in [78]. Each sequence is generated from 200 words, of which beginning is always the first word in a sentence. If a sentence is cut at the end of a sequence, that sentence is copied in full to the beginning of the next sequence.

To use our model, the POS of each word is required. However, the IWSLT dataset contains only the raw text of transcripts and does not include POS tags. Thus, we implement POS tagging using a special library [84] to predict the POS of each word.

5.2. Hyperparameter settings

Before mapping each token included in the unigram, bigram, and trigram to the embedding vector, we limit the minimum frequency of occurring words that are not marked as an unknown token. There are 2 parameters set for the unigram C_{word} and the remaining C_{ngram}, respectively. We found that model accuracy is highly sensitive to these parameters. Therefore, we use a grid search technique to find the best value for both parameters for the model.

We apply two optimizers used in this work: Adagard [85] and Adam [86], whose learning rates are set to 0.02 and 0.001 for the Thai and English datasets, respectively. To generalize the model, we also integrate L2 regularization with an alpha of 0.01 to the loss function for model updating. Moreover, dropout is applied to the local representation vectors, recurrent representation vectors, between all bidirectional LSTMs and enclosed by the self-attention mechanism in the high-level module.

During training, both the supervised and semi-supervised models are trained until the validation metrics stop improving; the metrics are (1) sentence boundary F1 score and (2) overall F1 score for Thai sentence segmentation and English punctuation restoration, respectively.

CVT has three main parameters that impact model accuracy. The first is the drop rate of the masked language model, which determines the number of tokens that are dropped and used for learning auxiliary prediction modules as described in Section 3.2. The second is the number of unlabeled mini-batches B used for training between supervised mini-batches. Third, rather than using the same dropout rate for the local representation vectors, a new dropout rate is assigned.

Meanwhile, ELMo is pre-trained with different learning rate from the model. The parameters of ELMo, such as the layers of Bi-LSTM and hidden nodes are set as same as original. The pre-training of ELMo is processed three epochs with batch size equals 32.

The hyperparameter values were determined through a grid search to find their optimal values on the different datasets. All the hyperparameters for each dataset are shown in Table 3. The optimal values from the grid search depend on the task. For Thai sentence segmentation, the hyperparameters are tuned to obtain the highest sentence boundary F1 score, while the overall F1 score is used to tune the parameters for English punctuation restoration.

Table 3. Model hyperparameters for each dataset.

Parameters	Orchid	UGWC	IWSLT
C_{word}	2	2	2
C_{ngram}	2	2	13
Optimizer	AdaGrad	AdaGrad	Adam
Learning Rate	0.02	0.02	0.001
Batch size	16	16	16
Early stopping patience	5	5	5
Unigram embedding size (Text)	64	64	300
Unigram embedding size (POS and Type)	32	32	300
Bigram & Trigram embedding size (Text)	16	16	10

Parameters	Orchid	UGWC	IWSLT
Bigram & Trigram embedding size (POS and Type)	8	8	10
LSTM hidden size	25	25	256
Number of LSTM layers in High-level module (K)	2	2	4
Self-attention output size	50	50	256
A number of Low-level Self-attention layers	1	1	1
A number of High-level Self-attention layers	1	1	1
Low-level Self-attention projection size	64	64	32
High-level Self-attention projection size	25	25	128
Local embedding dropout	0.30	0.30	0.30
Dropout between layers	0.15	0.15	0.15
Dropped rate of masked language model	-	0.30	0.30
Number of unlabeled mini-batch B	-	1	2
Dropout of the unlabeled input	-	0.50	0.30
Hidden size of LSTM in ELMo	-	4096	4096
Number of layers of LSTM in ELMo	-	2	2

5.3. Evaluation

During the evaluation, each task is assessed using different metrics based on previous works. For Thai sentence segmentation, three metrics are used in the evaluation: sentence boundary F1 score, non-sentence boundary F1 score, and space correct [27]. In this work, we mainly focus on the performance of sentence boundary prediction and not non-sentence boundary prediction or space prediction. Therefore, we make comparisons with other models regarding only their sentence boundary F1 scores. The equation for the sentence boundary F1 score metric is shown in Equation 32 where $\#(A)$ is the number of A. In calculating the F1 score, the positive class is defined as the sentence boundary, and the negative class is defined as the non-sentence boundary.

$$precision_{sb} = \frac{\#\text{Collectly predicted sentence boundaries}}{\#\text{All predicted sentence boundaries}}$$

$$recall_{sb} = \frac{\#\text{Collectly predicted sentence boundaries}}{\#\text{All expected sentence boundaries}}$$

$$F_{1,sb} = \frac{2 \times precision_{sb} \times recall_{sb}}{precision_{sb} + recall_{sb}} \tag{32}$$

For English punctuation, the evaluation is measured on each type of punctuation and overall F1 score. For the punctuation restoration task, we care only about the performance of the samples belonging to the classes that are tagged to words followed by punctuation; therefore class O, which represents words not immediately followed by punctuation, is ignored in the evaluation. Consequently, the overall F1 score does not include O as the positive class in Equation 33.

$$precision_{overall} = \frac{\text{\#Collectly predicted punctuation marks}}{\text{\#All predicted punctuation marks}}$$

$$recall_{overall} = \frac{\text{\#Collectly predicted punctuation marks}}{\text{\#All expected punctuation marks}}$$

$$F_{1,overall} = \frac{2 \times precision_{overall} \times recall_{overall}}{precision_{overall} + recall^{overall}} \tag{33}$$

To compare the performance of each punctuation restoration model in a manner similar to sentence segmentation, the binary F1 score is calculated to measure model accuracy, as shown in Equation 34. The calculation of this metric is the same as that used in [77]. The metric considers only where the punctuation position is and ignores the type of restored punctuation. Therefore, this measure is similar to the metric sentence boundary F1, which only considers the position of the missing punctuation.

$$precision_{2-class} = \frac{\text{\#Collectly predicted punctuation positions}}{\text{\#All predicted punctuation positions}}$$

$$recall_{2-class} = \frac{\text{\#Collectly predicted punctuation positions}}{\text{\#All expected punctuation positions}}$$

$$F_{1,2-class} = \frac{2 \times precision_{2-class} \times recall_{2-class}}{precision_{2-class} + recall_{2-class}} \tag{34}$$

5.4. Results and Discussions

We report and discuss the results of our two tasks in five subsections. Comparison of CNN and n-gram models for local representation are discussed in the first subsection. The second and third subsections include the effect of local representation and distant representation, respectively. The impact of CVT is explained in the fourth subsection. The last subsection presents a comparison of our model and all the baselines. Moreover, we also conduct paired t-tests to investigate the significance of the improvement of each contribution.

5.4.1. Comparison of CNN and n-gram models for local representation

Jacovi A. et al. [28] proposed that a CNN can be used as an n-gram detector to capture local text features. Therefore, we also performed an experiment to compare a CNN and n-gram embedded as local structures. The results in Table 4 show that the model using the embedded n-gram yields greater improvement than the one using an embedded CNN on the Orchid and UGWC datasets.

Table 4. Comparison between CNN and n-gram embedding for local representation extraction.

Model	F1 score (%)	
	ORCHID	*UGWC*
Bi-LSTM-CRF	90.9	87.6
Bi-LSTM-CRF + n-gram	92.4	88.7
Bi-LSTM-CRF + CNN	91.2	87.9

5.4.2. Effect of local representation

To find the effect of local representation, we compare a standard Bi-LSTM-CRF model using our full implementation to the model that includes n-gram embedding to extract local representation. In Table 5 and Table 6, the standard Bi-LSTM-CRF model is represented as Bi-LSTM-CRF (row (e)), while the models with local features are represented as $+ \, local$ (row (f)).

Table 5. The result of Thai sentence segmentation for each model.
For the Orchid dataset, we report the average of each metric on 10-fold cross-validation. Meanwhile, average metrics from 5-fold cross-validation are shown for the UGWC dataset.

Model	Orchid			UGWC		
	precision	*recall*	*F1*	*precision*	*recall*	*F1*
(a) POS-trigram [13]	74.4	79.8	77.0	-	-	-
(b) Winnow [12]	92.7	77.3	84.3	-	-	-
(c) ME [14]	86.2	83.5	84.8	-	-	-
(d) CRF (Thai baseline) [27]	94.7	89.3	91.9	87.4	82.7	85.0
(e) Bi-LSTM-CRF [21]	92.1	89.7	90.9	87.8	87.4	87.6
Our Improvement						
(f) + local	93.1	91.7	92.4	88.4	89.0	88.7
(g) + local + distant	**93.5**	**91.5**	**92.5**	88.8	88.8	88.8
(h) + local + distant + CVT	-	-	-	**88.9**	89.0	88.9
(i) + local + distant + CVT + ELMo	-	-	-	88.8	**91.0**	**89.9**

Note: The CVT model is not tested on the Orchid dataset because of the lack of unlabeled data.

5.4.2.1. Thai sentence segmentation

The results in Table 5 show that using n-gram to obtain the local representation improves the F1 score of the model from 90.9% (row (e)) to 92.4% (row (f)) on the Orchid dataset and from 87.6% (row (e)) to 88.7% (row (f)) on the UGWC dataset. These results occur because many word groups exist that can be used to signal the beginning and end of a sentence in Thai. Words always found near sentence boundaries can be categorized into two groups. The first group consists of final particles, e.g., "นะ|คะ" (na | kha), "นะ|ครับ" (na | khrạb), "เลย|ครับ" (ley | khrạb), "แล้ว|ครับ" (læ̂w | khrạb), and others. These word groups are usually used at the ends of sentences to indicate the formality level. For instance, the model with local representation can detect the sentence boundary at "ครับ" (khrạb) that is followed by "แล้ว" (læ̂w), as shown in Figure 10, while the model without local representation cannot detect the word as a sentence boundary. The second group consists of conjunctions that are always used at the beginnings of sentences, e.g., "จาก|นั้น (after that)", "ไม่|งั้น (otherwise)" and others. The model that uses n-gram to capture word group information is better able to detect word groups near sentence boundaries. Thus, this model can identify these sentence boundaries easily in the Thai language.

	มี (mī)	นาน (nān)	แล้ว (lǣ̂w)	ครับ (khrạb)	เริ่มต้น (reìm t̂n)	ที่ (thī̀)	1 (h̄nụ̀ng)	ล้าน (l̂ān)
Bi-LSTM-CRF								sb
+ local				sb				sb

Figure 10. An example of sentence boundary prediction.
The outputs are predicted by a normal Bi-LSTM-CRF and by the model with local representation ($+\ local$). Here, "sb" indicates that the word is predicted as the sentence boundary.

5.4.2.2. English punctuation restoration

In contrast, for the English dataset, local representation using n-gram drops the overall F1 score of punctuation restoration from 64.4% (row (g)) to 63.6% (row (h)), as shown in Table 6. However, the binary F1 score increases slightly from 81.4% (row (g)) to 81.8% (row (h)) when compared to the Bi-LSTM-CRF model, which does not integrate n-gram embedding. Common phrases such as “In spite of”, “Even though” and “Due to the fact” might provide strong cues for punctuation; however, such phrases can be found at both the beginnings and in the middle of sentences. Because such phrases can be used in both positions, they may follow commas when they are in the middle of the sentence or periods when they are at the beginning of a sentence.

However, they still follow either a period or a comma; consequently, such phrases can still help identify whether the punctuation should be restored, which increases the binary F1 score, which considers only the positions of missing punctuation. Moreover, English does not use the concept of a final particle usually found at the end of the sentence—similar to the Thai word group mentioned earlier—including "นะ|คะ"(na | kha), "นะ|ครับ"(na | khrạb), "เลย|ครับ" (ley | khrạb), "แล้ว|ครับ" (læ̂w | khrạb) and others. Therefore, the word groups captured by n-gram can only help to identify where punctuation should be restored but they do not help the model determine the type of punctuation that should be restored.

Table 6. The result of English punctuation restoration by each model. Each model is evaluated by the F1 score of each class of punctuation, the overall F1 score and the binary F1 score.

Model	Comma	Period	Question mark	Overall	Binary
(a) T-LSTM [70]	45.1	56.6	49.4	50.8	-
(b) T-BRNN [78]	53.1	71.9	62.8	63.1	-
(c) T-BRNN-pre [78]	54.8	72.9	66.7	64.4	-
(d) DRNN-LWMA	59.3	74.7	**73.3**	67.2	-
(e) DRNN-LWMA-pre	61.9	75.5	69.6	68.6	-
(f) CRF (Thai baseline) [27]	44.9	60.8	26.7	52.7	-
(g) Bi-LSTM-CRF [21]	55.3	73.1	63.5	64.4	81.4
Our Improvement					
(h) + local	55.0	72.3	63.3	63.6	81.8
(i) + local + distant	56.7	72.9	59.2	64.5	81.7
(j) + local + distant + CVT	56.8	73.7	61.2	65.4	82.8
(k) + local + distant + CVT + ELMO	**64.0**	**78.6**	66.7	**71.0**	**86.9**

Note: F1 score for the "O" class, which indicates that there is no punctuation following the word, is not calculated because we care only the performance of each punctuation class.

5.4.2.3. Interpretation of n-gram feature

In this section, our model is interpreted and analyzed via the method presented in Section 2.5. $\text{Count}(pos, gram)$, which refer how similarity between the model and human, is calculated over the training set of each corpus, as shown in Table 7. The result of the interpretation shows the focused n-gram features are rarely found around the punctuation. Compared with the Thai sentence segmentation model, the focused n-gram features are usually located at the sentence boundary.

In UGWC section of Table 7, the result indicates that the model mostly focuses at the end of the sentence more than the beginning of the sentence due to the lower number of focused features at the relative position $pos = +1$. In the intersection list from UGWC at the relative position $pos = -1,0$, the features found near sentence boundaries are usually final particles, e.g., “นะ|คะ” (**ná I khá**), “นะ|ครับ” (**ná I khráp**), “เลย|ครับ” (**lə:y I khráp**), “แล้ว|ครับ” (**lǽw I khráp**), and others. These features are usually used at the ends of sentences to indicate the formality level. Meanwhile, the intersection list at the relative position $pos = +1$ are composed of greeting words and "Thank you" phrases that are always at the beginning of sentence, e.g., “สวัสดี (Hello)”, “ขอบคุณ (Thank you)” and others.

In contrast, in Orchid, the model focuses more at the beginning of sentence. The features can be categorized into two groups. First, the content words [82], such as topics (“บทคัดย่อ (abstract)”, “บทนำ (introduction)”), are focused due to their high frequency caused by the similar structure between documents. The second group is the function words [82], such as “เป็นดังนี้ (as follows)”, “ข้างล่างนี้ (as shown below)” and “ได้ดังนี้ (as follows)”, which are simply classified as the end of paragraphs or sentences.

As a result, the models of both Thai datasets focus on different types of features due to the different writing styles. Despite these findings, our model is able to learn and utilize these n-grams features, which are also used by humans to decide where the sentence boundary is. Therefore, using local representation to capture n-grams features attains more improvement over other contributions in Thai sentence segmentation.

Meanwhile, in English punctuation restoration, the model hardly learns n-gram features around the punctuation. As a result, the n-gram features degrades the performance in English punctuation restoration due to the overfitting problem.

Table 7. The number of features in the intersection of two lists that is created from the interpretation score and labels.

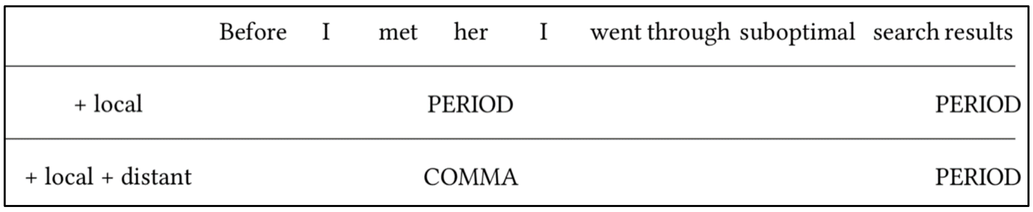
Type of n-gram (gram)	Relative position (pos)								
	Orchid			UGWC			IWSLT		
	-1	0	+1	-1	0	+1	-1	0	+1
Uni-gram	51	64	117	203	171	152	0	10	1
Bi-gram	6	22	54	162	165	3	0	10	0
Tri-gram	45	98	56	206	75	28	3	5	4

5.4.3. Effect of distant representation

The effect of this contribution can be found by comparing the model that integrates the distant representation and the model that does not. The model with distant features integrated is represented as $+\, local\ +\ distant$ (row (g) and row (i) in Table 5 and Table 6 respectively). In this case, the distant representation is composed of the self-attention modules in both the low- and high-level modules, as shown in Figure 4.

From the combination of local and distant representation, the results in Table 5 and Table 6 show that the distant feature improves the accuracy of the model on all datasets compared to the model with no distant representation. The F1 scores of the sentence segmentation models improved slightly, from 92.4% and 88.7% (row (f)) to 92.5% and 88.8% (row (g)) on the Orchid and UGWC datasets, respectively. For the IWSLT dataset, the distant feature can recover the overall F1 score of punctuation restoration, which is degraded by the n-gram embedding; it improves from 63.6% (row (h)) to 64.5% (row (i)). The reason is that the self-attention modules focus selectively on certain parts of the passage. Thus, the model focuses on the initial words of the dependent clauses, which helps in classifying which type of punctuation should be restored. An example is shown in Figure 11: the model with distant representation classifies the punctuation after “her” as a “COMMA” because “Before” is the word that indicates the dependent clause. Meanwhile, the model without distant representation predicts the punctuation as a “PERIOD” because there is no self-attention module; therefore, it does not focus on the word “Before”.

This also illustrates that the model can be improved by adding the self-attention modules to Bi-LSTM layers. In conclusion, the results have shown that each of the proposed modules have a positive effect on the overall performance.



	Before	I	met	her	I	went through	suboptimal	search results
+ local				PERIOD				PERIOD
+ local + distant				COMMA				PERIOD

Figure 11. An example of punctuation prediction.
The outputs are predicted by the model with distant representation ($+\, local\ +\ distant$) and without distant representation ($+\, local$). The "COMMA" indicates that the word is followed by a comma (,) and "PERIOD" indicates that a period (.) is restored at that position.

5.4.4. Effect of CVT

To identify the improvement from CVT, we compared the models that use different training processes: standard supervised training ($+local + distant$) and CVT ($+local + distant + CVT$). The model trained with CVT improves the accuracy in terms of the F1 score on both Thai and English datasets, as shown in Table 5 and Table 6.

5.4.4.1. Thai sentence segmentation

This experiment was conducted only on the UGWC dataset because no unlabeled data are available in the Orchid dataset, as mentioned in Section 5.1.1.1. The model improves the F1 score slightly, from 88.8% (row (g)) to 88.9% (row (h)) on the UGWC dataset. Since the labeled and unlabeled data in the UGWC dataset were drawn from the same source and domain, it provides little additional knowledge that can be learned by the model. The average number of new words found in the unlabeled data is only 0.650 per passage, as shown in Table 5. In other words, the unlabeled passages barely contain any new words.

5.4.4.2. English punctuation restoration

CVT also improved the model on the IWSLT dataset, from an overall F1 score of 64.5% (row (g)) to 65.3% (row (h)) and from a 2-class F1 score of 81.7% to 82.7%. Because both the labeled and unlabeled data were collected from TED talks, the number of vocabulary words grows substantially more than in the UGWC dataset because the talks cover various topics. In this dataset, an average of 1.225 new words are found in each new unlabeled data passage, as shown in Table 6. Thus, the improvement on the IWSLT dataset is more noticeable than in the UGWC dataset.

5.4.5. Effect of ELMo

By applying ELMo or contextual representation to the model, an accuracy in both tasks increase noticeably, from 88.8% to 89.9% in Thai sentence segmentation and from 65.4% to 71.0% in English punctuation restoration. From the results, the gap of improvement in Thai is lower than in English because the English ELMo is trained from one billion words which is ten times higher than the data which we used in Thai. Therefore, Thai ELMo that is given more data might elevate the accuracy of Thai sentence segmentation.

5.4.6. Comparison with baseline models

5.4.6.1. Thai sentence segmentation

For the Thai sentence segmentation task, our model is superior to all the baselines on both Thai sentence segmentation datasets, as shown in Table 5. On the Orchid dataset, the supervised model that includes both local and distant representation was adopted for comparison to the baseline model. Our model improves the F1 score achieved by CRF-ngram, which is the state-of-the-art model for Thai sentence segmentation in Orchid, from 91.9% (row (d)) to 92.5% (row (h)). Meanwhile, in the UGWC dataset, our CVT model with ELMo (row (h)) achieves an F1 score of 89.5%, which is higher than the F1 score of both the baselines (CRF-ngram and Bi-LSTM-CRF (rows d and e, respectively)). Thus, our model is now the state-of-the-art model for Thai sentence segmentation on both the Orchid and UGWC datasets.

To prove the significance of the model improvements, we compared the cross-validation results using paired t-tests to obtain the p-values, which are shown in Table 8 for the Orchid dataset and Table 9 for the UGWC dataset.

Table 8. The improvement of each contribution on the Orchid dataset results shown as p-values from paired t-tests

Model	CRF	Bi-LSTM-CRF	+ local
+ local	+0.47% ± 0.42% (0.009)	+1.53% ± 0.39% (<0.001)	-
+ local + distant	+0.54% ± 0.36% (0.002)	+1.60% ± 0.39% (<0.001)	+0.07% ± 0.22% (0.370)

Note: The number in the table reflects the percentage of improvement from the columns compared with the rows. The number in parentheses is the p-value computed from a paired t-test.

Table 9. The improvement of each contribution on the UGWC dataset results shown as p-values from paired t-tests

Model	CRF	Bi-LSTM-CRF	+ local	+ local + distant	+ local + distant + CVT
+ local	+3.66% ± 0.16% (<0.001)	+1.09% ± 0.14% (<0.001)			
+ local + distant	+3.77% ± 0.20% (<0.001)	+1.20% ± 0.20% (<0.001)	+0.11% ± 0.14% (0.182)		
+ local + distant + CVT	+3.92% ± 0.20% (<0.001)	+1.34% ± 0.14% (<0.001)	+0.26% ± 0.06% (0.001)	+0.15% ± 0.12% (0.065)	
+ local + distant + CVT + ELMo	+4.87% ± 0.25% (<0.001)	+2.29% ± 0.16% (<0.001)	+1.21% ± 0.10% (<0.001)	+1.10% ± 0.14% (<0.001)	+0.95% ± 0.05% (<0.001)

Note: The number in the table reflects the percentage of improvement from the columns compared with the rows. The number in parentheses is the p-value computed from a paired t-test.

5.4.6.2. English punctuation restoration

Our model outperforms all the sequence tagging models. DRNN-LWMA-pre (row (e)) is the current state-of-the-art model, as shown in Table 6. The CVT model with ELMo improves the overall F1 score from 68.6% of DRNN-LWMA-pre to 71.0% (row (k)).

5.4.7. Discussion

We have shown that incorporating local and global information with CVT can be used to improve the Thai sentence segmentation and English punctuation restoration tasks. However, we would like to note that our proposed method assumes no idiosyncrasies specific to the Thai language. They might be able to improve other languages or tasks as well. For example, one might consider the tasks of Elementary Discourse Unit (EDU) and clause segmentation which can help downstream tasks such as text summarization and machine translation by providing the minimal syntactic units. Also, the experiment results show that each contribution yields different results for each language. Thus, the results of features or methods are still essential to be discovered for a particular language, even though the trend of current NLP research is trying to find the best generic method for every language.

Moreover, due to the scarcity of labeled data in Thai, more amount of data with sentence boundaries is still needed. However, only a large number of instances are insufficient to build a generic model that can be applied to any written styles and domains; the dataset should be built from various data sources too. Also, all available sentence segmentation datasets, such as ORCHID, UGWC, and the recently released LST20 [87], should be integrated. However, the annotation criteria of the datasets are different making the task non-trivial. One possible venue for exploration is to use multi-criteria learning to utilize the shared information in all datasets. This method was successfully applied to Chinese word segmentation, which has a similar problem [88, 89].

6. CONCLUSIONS

In this paper, we propose a novel deep learning model for Thai sentence segmentation. This study makes three main contributions. The first contribution is to integrate a local representation based on n-gram embedding into our deep model. This approach helps to capture word groups near sentence boundaries, allowing the model to identify boundaries more accurately. Second, we integrate a distant representation obtained from self-attention modules to capture sentence contextual information. This approach allows the model to focus on the initial words of dependent clauses (i.e., "Before", "If", and "Although"). The last contribution is an adaptation of CVT, which allows the model to utilize unlabeled data to produce effective local and distant representations.

The experiment was conducted on two Thai datasets, Orchid and UGWC, and one English punctuation restoration dataset, IWSLT. On the Thai sentence segmentation task, our model achieves F1 scores of 92.5% and 89.9% on the Orchid and UGWC datasets, constituting a relative error reduction of 7.4% and 18.5%, respectively. On the English punctuation task, the binary F1 score reached 86.9% when considering only two punctuation classes (making the task similar to sentence segmentation in Thai). Based on our contributions, the local representation has the highest impact on the Thai corpus. From the interpretation process, the local representation revealed that it captures phrases that frequently occurred near the sentence boundary, which is usually the approach used by humans to recognize the boundary. Meanwhile, the distant representation and CVT result in strong improvements on the English dataset. Also, ELMo is suitable for both Thai and English corpus.

APPENDIX

A. Performance on a public UGWC dataset

In this section, the model is evaluated on a public dataset, a subset of UGWC; the dataset consists of 15,000 passages, 28,514 sentences, and 183,193 words. Due to the lack of public unlabeled data, the semi-supervised technique does not be applied. Thus, only the model with local and distant representation is evaluated by 5-fold validation. The performance is shown in Table 10.

Table 10. The result of public UGWC and LST20 by the proposed model.

Algorithm	Public UGWC			LST20		
	Precision	Recall	F1 score	Precision	Recall	F1 score
Bi-LSTM	91.730	91.000	91.360	78.036	59.806	67.715
+ local + distant	92.262	91.196	91.718	78.578	59.952	68.012

B. Performance on LST20 dataset [87]

LST20 Corpus is a news dataset developed by National Electronics and Computer Technology Center (NECTEC), Thailand. The dataset includes word boundaries, part of speech tagging, named entities, clause boundaries, and sentence boundaries. For data preprocessing, sentences with more than 100 words are filtered out. Each instance is selected from consecutive sentences in the same file with no filtered sentence between them. If the instance is longer than ten words, the instance will be divided into two or more instances consisting of ten sentences or less. After that, the instances that consist of only one sentence are filtered out. As a result, the data statistics of preprocessed datasets are shown in Table 11. The code for data preprocessing is publicly available on GitHub. [1]

Table 11. The number of instances, sentences, and words in preprocessed LST20 dataset.

Dataset	# instances	# sentences	# words
Train	8,315	60,269	2,290,655
Eval	784	5,452	217,652
Test	699	5,201	200,079

[1] https://github.com/ChanatipSaetia/PreprocessLST20/blob/main/PrepareLST20.ipynb

To apply to our proposed model, sentences in each instance are concatenated with a space token (‘_’) into one sequence of tokens. Each token is labeled as a sentence boundary when it is the end of sentences or the added space token. Similar to the Orchid dataset, in which there is no unlabeled dataset using the same word tokenization criteria, so the semi-supervised method is not adopted. Thus, only the model with local and distant representation is evaluated. Its performance is shown in Table 10.

REFERENCES

1. Kübler, S., R. McDonald, and J. Nivre, *Dependency parsing.* Synthesis lectures on human language technologies, 2009. **1**(1): p. 1-127.

2. Gillick, D., *Sentence boundary detection and the problem with the US*, in *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics, Companion Volume: Short Papers*. 2009. p. 241-244.

3. Aharoni, R., M. Johnson, and O. Firat, *Massively multilingual neural machine translation.* arXiv preprint arXiv:1903.00089, 2019.

4. Bahdanau, D., K. Cho, and Y. Bengio. *Neural machine translation by jointly learning to align and translate*. in *3rd International Conference on Learning Representations, ICLR 2015*. 2015.

5. Brown, P.F., et al., *A statistical approach to machine translation.* Computational linguistics, 1990. **16**(2): p. 79-85.

6. Mihalcea, R., *Graph-based ranking algorithms for sentence extraction, applied to text summarization*, in *Proceedings of the ACL Interactive Poster and Demonstration Sessions*. 2004.

7. Afsharizadeh, M., H. Ebrahimpour-Komleh, and A. Bagheri, *Query-oriented text summarization using sentence extraction technique*, in *2018 4th International Conference on Web Research (ICWR)*. 2018. p. 128-132.

8. Joshi, A., et al., *SummCoder: An unsupervised framework for extractive text summarization based on deep auto-encoders.* Expert Systems with Applications, 2019. **129**: p. 200-215.

9. Li, Z., et al. *Semi-supervised domain adaptation for dependency parsing*. in

Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics. 2019.

10. Qi, P., et al., *Universal dependency parsing from scratch.* arXiv preprint arXiv:1901.10457, 2019.

11. Aroonmanakun, W. and others, *Thoughts on word and sentence segmentation in Thai*, in *Proceedings of the Seventh Symposium on Natural language Processing, Pattaya, Thailand, December 13--15*. 2007. p. 85-90.

12. Charoenpornsawat, P. and V. Sornlertlamvanich, *Automatic sentence break disambiguation for Thai*, in *International Conference on Computer Processing of Oriental Languages (ICCPOL)*. 2001. p. 231-235.

13. Mittrapiyanuruk, P. and V. Sornlertlamvanich, *The automatic Thai sentence extraction*, in *Proceedings of the fourth symposium on Natural Language Processing*. 2000. p. 23-28.

14. Slayden, G., M.-Y. Hwang, and L. Schwartz, *Thai sentence-breaking for large-scale SMT*, in *Proceedings of the 1st Workshop on South and Southeast Asian Natural Language Processing*. 2010. p. 8-16.

15. Zhou, N., et al., *A Word Labeling Approach to Thai Sentence Boundary Detection and POS Tagging*, in *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*. 2016, The COLING 2016 Organizing Committee: Osaka, Japan. p. 319-327.

16. Lafferty, J., A. McCallum, and F.C.N. Pereira, *Conditional random fields: Probabilistic models for segmenting and labeling sequence data.* 2001.

17. Jurafsky, D. and J.H. Martin, *Speech and Language Processing: An introduction to speech recognition, computational linguistics and natural language processing.* Upper Saddle River, NJ: Prentice Hall, 2008.

18. Sornlertlamvanich, V., T. Charoenporn, and H. Isahara, *ORCHID: Thai part-of-speech tagged corpus.* National Electronics and Computer Technology Center Technical Report, 1997: p. 5-19.

19. Hochreiter, S. and J. Schmidhuber, *Long short-term memory.* Neural computation, 1997. **9**: p. 1735-1780.

20. Vaswani, A., et al., *Attention is all you need*, in *Advances in neural information processing systems*. 2017. p. 5998-6008.

21. Huang, Z., W. Xu, and K. Yu, *Bidirectional LSTM-CRF models for sequence tagging.* arXiv preprint arXiv:1508.01991, 2015.

22. Yi, J., et al., *Adversarial transfer learning for punctuation restoration.* arXiv preprint arXiv:2004.00248, 2020.

23. Akbik, A., T. Bergmann, and R. Vollgraf. *Pooled contextualized embeddings for named entity recognition*. 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)*. 2019.

24. Akbik, A., D. Blythe, and R. Vollgraf. *Contextual string embeddings for sequence labeling*. in *Proceedings of the 27th international conference on computational linguistics*. 2018.

25. Straková, J., M. Straka, and J. Hajič, *Neural architectures for nested NER through linearization.* arXiv preprint arXiv:1908.06926, 2019.

26. Wang, Z., et al. *CrossWeigh: Training Named Entity Tagger from Imperfect Annotations*. in *Proc. 2019 Conferenc. on Empirical Methods in Natural Language Processing and the 9th International Joint Conf. on Natural Language Processing, EMNLP-IJCNLP 2019*. 2019.

27. Zhou, N., et al., *A Word Labeling Approach to {T}hai Sentence Boundary*

Detection and {POS} Tagging, in *Proceedings of {COLING} 2016, the 26th International Conference on Computational Linguistics: Technical Papers*. 2016, The COLING 2016 Organizing Committee: Osaka, Japan. p. 319-327.

28. Jacovi, A., O.S. Shalom, and Y. Goldberg, *Understanding convolutional neural networks for text classification.* arXiv preprint arXiv:1809.08037, 2018.

29. Devlin, J., et al., *Bert: Pre-training of deep bidirectional transformers for language understanding.* arXiv preprint arXiv:1810.04805, 2018.

30. Zhu, X.J. *Semi-supervised learning literature survey*. 2005.

31. Miyato, T., et al., *Virtual adversarial training: a regularization method for supervised and semi-supervised learning.* IEEE transactions on pattern analysis and machine intelligence, 2018.

32. Rasmus, A., et al., *Semi-supervised learning with ladder networks*, in *Advances in neural information processing systems*. 2015. p. 3546-3554.

33. Clark, K., et al., *Semi-Supervised Sequence Modeling with Cross-View Training*, in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. 2018. p. 1914-1925.

34. Miyato, T., A.M. Dai, and I. Goodfellow, *Adversarial training methods for semi-supervised text classification.* arXiv preprint arXiv:1605.07725, 2016.

35. Ruder, S. and B. Plank, *Strong baselines for neural semi-supervised learning under domain shift.* arXiv preprint arXiv:1804.09530, 2018.

36. Peters, M.E., et al., *Deep contextualized word representations.* arXiv preprint arXiv:1802.05365, 2018.

37. Mikolov, T., et al. *Distributed representations of words and phrases and their*

compositionality. in *Advances in neural information processing systems*. 2013.

38. Radford, A., et al., *Improving language understanding with unsupervised learning*. 2018, Technical report, OpenAI.

39. Liu, Y., et al., *Roberta: A robustly optimized bert pretraining approach.* arXiv preprint arXiv:1907.11692, 2019.

40. Yang, Z., et al., *Xlnet: Generalized autoregressive pretraining for language understanding.* arXiv preprint arXiv:1906.08237, 2019.

41. Sanh, V., et al., *DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter.* arXiv preprint arXiv:1910.01108, 2019.

42. Lan, Z., et al., *Albert: A lite bert for self-supervised learning of language representations.* arXiv preprint arXiv:1909.11942, 2019.

43. Lertpiya, A., et al., *A Preliminary Study on Fundamental Thai NLP Tasks for User-generated Web Content*, in *2018 International Joint Symposium on Artificial Intelligence and Natural Language Processing (iSAI-NLP)*. 2019. p. 1-8.

44. Wathabunditkul, S., *Spacing in the Thai Language*. 2003.

45. Chevalier, G., *The LSTM cell*. 2018: Wikipedia. p. The Long Short-Term Memory (LSTM) cell can process data sequentially and keep its hidden state through time.

46. McClosky, D., E. Charniak, and M. Johnson. *Effective self-training for parsing*. in *Proceedings of the main conference on human language technology conference of the North American Chapter of the Association of Computational Linguistics*. 2006. Association for Computational Linguistics.

47. Yarowsky, D. *Unsupervised word sense disambiguation rivaling supervised methods*. in *33rd annual meeting of the association for computational*

linguistics. 1995.

48. Blum, A. and T. Mitchell. *Combining labeled and unlabeled data with co-training*. in *Proceedings of the eleventh annual conference on Computational learning theory*. 1998. Citeseer.

49. Zhou, Y. and S. Goldman. *Democratic co-learning*. in *16th IEEE International Conference on Tools with Artificial Intelligence*. 2004. IEEE.

50. Zhou, Z.-H. and M. Li, *Tri-training: Exploiting unlabeled data using three classifiers.* IEEE Transactions on Knowledge & Data Engineering, 2005(11): p. 1529-1541.

51. McCann, B., et al. *Learned in translation: Contextualized word vectors*. in *Advances in Neural Information Processing Systems*. 2017.

52. Peters, M.E., et al., *Semi-supervised sequence tagging with bidirectional language models.* arXiv preprint arXiv:1705.00108, 2017.

53. Doshi-Velez, F. and B. Kim, *Towards a rigorous science of interpretable machine learning.* arXiv preprint arXiv:1702.08608, 2017.

54. Gururangan, S., et al., *Annotation artifacts in natural language inference data.* arXiv preprint arXiv:1803.02324, 2018.

55. Simonyan, K., A. Vedaldi, and A. Zisserman, *Deep inside convolutional networks: Visualising image classification models and saliency maps.* arXiv preprint arXiv:1312.6034, 2013.

56. Ebrahimi, J., et al., *Hotflip: White-box adversarial examples for text classification.* arXiv preprint arXiv:1712.06751, 2017.

57. Feng, S., et al. *Pathologies of Neural Models Make Interpretation Difficult*. in

Empirical Methods in Natural Language Processing.

58. Ribeiro, M.T., S. Singh, and C. Guestrin. *" Why should i trust you?" Explaining the predictions of any classifier*. in *Proceedings of the 22nd ACM SIGKDD international conference on knowledge discovery and data mining*. 2016.

59. Lundberg, S.M. and S.-I. Lee. *A unified approach to interpreting model predictions*. in *Advances in neural information processing systems*. 2017.

60. Yosinski, J., et al. *Understanding neural networks through deep visualization*. in *In ICML Workshop on Deep Learning*. Citeseer.

61. Wallace, E., et al., *AllenNLP Interpret: A Framework for Explaining Predictions of NLP Models.* EMNLP-IJCNLP 2019, 2019: p. 7.

62. Sundararajan, M., A. Taly, and Q. Yan. *Axiomatic attribution for deep networks*. in *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. 2017. JMLR. org.

63. Smilkov, D., et al., *Smoothgrad: removing noise by adding noise.* arXiv preprint arXiv:1706.03825, 2017.

64. Tangsirirat, N., et al. *Contextual behaviour features and grammar rules for Thai sentence-breaking*. in *2013 10th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology*. 2013. IEEE.

65. WANG, H., et al., *MAXIMUM ENTROPY THAI SENTENCE SEGMENTATION COMBINED WITH THAI GRAMMAR RULES CORRECTION.*

66. Nararatwong, R., et al., *Improving Thai word and sentence segmentation using linguistic knowledge.* IEICE TRANSACTIONS on Information and Systems, 2018. **101**(12): p. 3218-3225.

67. Gotoh, Y. and S. Renals, *Sentence boundary detection in broadcast speech transcripts*, in *ASR2000-Automatic Speech Recognition: Challenges for the new Millenium ISCA Tutorial and Research Workshop (ITRW)*. 2000.

68. Kolá\vr, J. and L. Lamel, *Development and evaluation of automatic punctuation for French and English speech-to-text*, in *Thirteenth Annual Conference of the International Speech Communication Association*. 2012.

69. Kolá\vr, J., E. Shriberg, and Y. Liu, *Using prosody for automatic sentence segmentation of multi-party meetings*, in *International Conference on Text, Speech and Dialogue*. 2006. p. 629-636.

70. Tilk, O. and T. Alumäe, *LSTM for punctuation restoration in speech transcripts*, in *Sixteenth annual conference of the international speech communication association*. 2015.

71. Cho, E., et al., *Punctuation insertion for real-time spoken language translation*, in *Proceedings of the Eleventh International Workshop on Spoken Language Translation*. 2015.

72. Peitz, S., et al., *Modeling punctuation prediction as machine translation*, in *International Workshop on Spoken Language Translation (IWSLT) 2011*. 2011.

73. Wang, F., et al., *Self-Attention Based Network for Punctuation Restoration*, in *2018 24th International Conference on Pattern Recognition (ICPR)*. 2018. p. 2803-2808.

74. Gravano, A., M. Jansche, and M. Bacchiani, *Restoring punctuation and capitalization in transcribed speech*, in *2009 IEEE International Conference on Acoustics, Speech and Signal Processing*. 2009. p. 4741-4744.

75. Lu, W. and H.T. Ng, *Better punctuation prediction with dynamic conditional random fields*, in *Proceedings of the 2010 conference on empirical methods in*

natural language processing. 2010. p. 177-186.

76. Ueffing, N., M. Bisani, and P. Vozila, *Improved models for automatic punctuation prediction for spoken and written text.*, in *Interspeech*. 2013. p. 3097-3101.

77. Che, X., et al., *Punctuation Prediction for Unsegmented Transcript Based on Word Vector.*, in *Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC 2016)*. 2016.

78. Tilk, O. and T. Alumäe, *Bidirectional Recurrent Neural Network with Attention Mechanism for Punctuation Restoration.* 2016.

79. Kim, S. *Deep Recurrent Neural Networks with Layer-wise Multi-head Attentions for Punctuation Restoration*. in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2019. IEEE.

80. Rabiner, L.R., *A tutorial on hidden Markov models and selected applications in speech recognition.* Proceedings of the IEEE, 1989. **77**(2): p. 257-286.

81. Taylor, W.L., *"Cloze procedure": A new tool for measuring readability.* Journalism Bulletin, 1953. **30**: p. 415-433.

82. Fries, C.C., *The structure of English.* 1952.

83. Federico, M., et al., *Overview of the IWSLT 2012 evaluation campaign*, in *IWSLT-International Workshop on Spoken Language Translation*. 2012. p. 12-33.

84. Honnibal, M. and I. Montani, *spacy 2: Natural language understanding with bloom embeddings, convolutional neural networks and incremental parsing.* To appear, 2017.

85. Duchi, J., E. Hazan, and Y. Singer, *Adaptive subgradient methods for online learning and stochastic optimization.* Journal of Machine Learning Research,

2011. **12**: p. 2121-2159.

86. Kingma, D.P. and J. Ba, *Adam: A method for stochastic optimization.* arXiv preprint arXiv:1412.6980, 2014.

87. Boonkwan, P., et al., *The Annotation Guideline of LST20 Corpus.* arXiv preprint arXiv:2008.05055, 2020.

88. Huang, W., et al., *Toward fast and accurate neural chinese word segmentation with multi-criteria learning.* arXiv preprint arXiv:1903.04190, 2019.

89. Chen, X., et al., *Adversarial multi-criteria learning for chinese word segmentation.* arXiv preprint arXiv:1704.07556, 2017.

VITA

NAME	Chanatip Saetia
DATE OF BIRTH	5 November 1995
PLACE OF BIRTH	Bangkok
INSTITUTIONS ATTENDED	Department of Computer Engineering, Faculty of Engineering, Chulalongkorn University
HOME ADDRESS	93-99, Rat Anusorn Road, Bang Phra, Mueang Trat, Trat 23000
PUBLICATION	Saetia, C., & Vateekul, P. (2018, July). Enhance Accuracy of Hierarchical Text Categorization Based on Deep Learning Network Using Embedding Strategies. In 2018 15th International Joint Conference on Computer Science and Software Engineering (JCSSE) (pp. 1-6). IEEE.