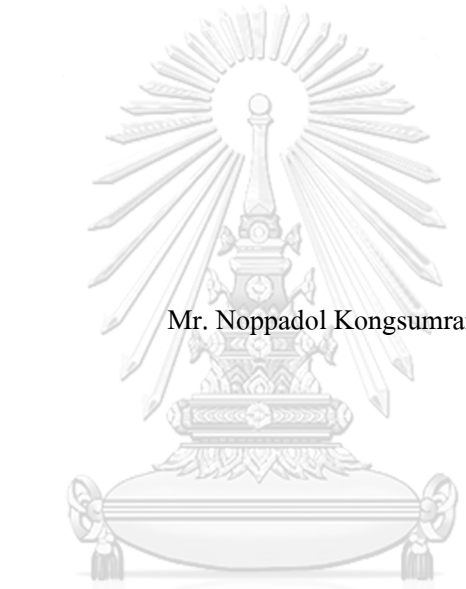
THAI TOKENIZER INVARIANT CLASSIFICATION BASED ON BI-LSTM AND DISTILBERT ENCODERS

Mr. Noppadol Kongsumran



A Thesis Submitted in Partial Fulfillment of the Requirements

for the Degree of Master of Science in Computer Science and Information Technology

Department of Mathematics and Computer Science

FACULTY OF SCIENCE

Chulalongkorn University

Academic Year 2021

การจำแนกที่ไม่แปรเปลี่ยนตามโทเคนไนเซอร์ภาษาไทยบนฐานของตัวเข้ารหัสไบแอลเอสทีเอ็มและดิสทิลเบิร์ต

นายนพดล คงสำราญ

วิทยานิพนธ์นี้เป็นส่วนหนึ่งของการศึกษาตามหลักสูตรปริญญาวิทยาศาสตรมหาบัณฑิต
สาขาวิชาวิทยาการคอมพิวเตอร์และเทคโนโลยีสารสนเทศ ภาควิชาคณิตศาสตร์และวิทยาการคอมพิวเตอร์
คณะวิทยาศาสตร์ จุฬาลงกรณ์มหาวิทยาลัย
ปีการศึกษา 2564

Thesis Title	THAI TOKENIZER INVARIANT CLASSIFICATION BASED ON BI-LSTM AND DISTILBERT ENCODERS
By	Mr. Noppadol Kongsumran
Field of Study	Computer Science and Information Technology
Thesis Advisor	Associate Professor SUPHAKANT PHIMOLTARES, Ph.D.

Accepted by the FACULTY OF SCIENCE, Chulalongkorn University in Partial Fulfillment of the Requirement for the Master of Science

..... Dean of the FACULTY OF SCIENCE

(Professor POLKIT SANGVANICH, Ph.D.)

THESIS COMMITTEE

..... Chairman

(Professor CHIDCHANOK LURSINSAP, Ph.D.)

..... Thesis Advisor

(Associate Professor SUPHAKANT PHIMOLTARES, Ph.D.)

..... Examiner

(Prem Junsawang, Ph.D.)

นพดล คงสำราญ : การจำแนกที่ไม่แปรเปลี่ยนตามโทเคนไนเซอร์ภาษาไทยบนฐานของตัวเข้ารหัสไบแอลเอสทีเอ็มและดิสทิลเบิร์ต. (THAI TOKENIZER INVARIANT CLASSIFICATION BASED ON BI-LSTM AND DISTILBERT ENCODERS) อ.ที่ปรึกษาหลัก : รศ. ดร.ศุภกานต์ พิมลธเรศ

การประมวลผลภาษาธรรมชาติเป็นหัวข้อในปัญญาประดิษฐ์เพื่อสอนคอมพิวเตอร์ให้เข้าใจภาษามนุษย์ นักวิจัยสามารถป้อนข้อความของภาษาที่เจาะจงในความยาวและประเภทใดๆ เช่น อักขระ คำ และประโยคไปยังขั้นตอนวิธี เพื่อแยกบริบทที่สรุปในรูปของตัวเลข และเพื่อให้ยอมรับอาร์เรย์ของคำในภาษาไทย กระบวนการตัดคำจึงจำเป็นใช้แยกข้อความเป็นคำ เนื่องจากประโยคแต่ละประโยคเขียนต่อเนื่องกันโดยไม่มีช่องว่างระหว่างคำ โดยทั่วไปแล้วตัวตัดคำที่ต่างกันสามารถสร้างชุดคำที่ต่างกันจากประโยคเดียวได้ ส่งผลให้ไม่สามารถควบคุมความแม่นยำในการประมวลผลภาษาธรรมชาติและปัญหาที่สัมพันธ์ได้ ในงานวิจัยนี้วิธีที่ใช้แก้ปัญหาผลลัพธ์ที่แตกต่างกันจากตัวตัดคำภาษาไทยที่แตกต่างกันได้ถูกนำเสนอโดยวางแนวผลการตัดคำให้อยู่ในทิศทางเดียวกันโดยใช้ตัวเข้ารหัสโครงข่ายประสาท ไบแอลเอสทีเอ็มและดิสทิลเบิร์ตร่วมกับค่าสูญเสียถาวรทริปเลตใช้เพื่อฝึกและแปลงชุดคำให้เป็นข้อมูลในโดเมนใหม่ที่เวกเตอร์ของแต่ละประโยคที่คล้ายกันอยู่ใกล้กันมากขึ้น ในท้ายที่สุดตัวจำแนกจำนวนยี่สิบแปดตัวถูกสร้างขึ้นมาโดยใช้ตัวเข้ารหัสสองประเภท ตัวตัดคำเจ็ดตัว โดยใช้หรือไม่ใช้วิธีที่เสนอเพื่อการเปรียบเทียบและการวิเคราะห์ และเพื่อแสดงว่าวิธีที่เสนอสามารถใช้เป็นวิธีเริ่มฝึกฝนสำหรับงานอื่นๆ ได้ ชุดข้อมูลความรู้สึกถูกใช้เพื่อวัดความแม่นการจำแนกและตรวจสอบความเหมือนของผลที่ได้จากตัวจำแนกทั้งหมด

สาขาวิชา	วิทยาการคอมพิวเตอร์และเทคโนโลยีสารสนเทศ	ลายมือชื่อนิสิต .....
ปีการศึกษา	2564	ลายมือชื่อ อ.ที่ปรึกษาหลัก .....

6378015423 : MAJOR COMPUTER SCIENCE AND INFORMATION TECHNOLOGY

KEYWORD: Bi-LSTM Encoder, DistilBERT Encoder, Triplet Hard Loss, Sentiment Classification

Noppadol Kongsumran : THAI TOKENIZER INVARIANT CLASSIFICATION BASED ON BI-LSTM AND DISTILBERT ENCODERS. Advisor: Assoc. Prof. SUPHAKANT PHIMOLTARES, Ph.D.

Natural language processing (NLP) is a topic in artificial intelligence to teach computer to understand human language. Researchers can feed text of some particular language in any length and type such as characters, words, and sentences into the algorithm to extract a summarized context in terms of numbers. To accept a word array in Thai language, tokenization process is needed to split a text into words because each sentence is written consecutively without any space between words. In general, different tokenizers can produce different sets of words from a single sentence, resulting in uncontrolled accuracies in NLP and related tasks. In this research, a method to solve the different results from different Thai tokenizers is introduced by aligning tokenization results together in the similar direction using neural networks encoders. Bi-LSTM and DistilBERT with triplet hard loss are used to train and transform sets of words to data in a new domain where vectors of each similar sentence are significantly closer. Finally, twenty-eight classifiers are created using two types of encoders, seven different tokenizers, with and without using the proposed method for comparative and analysis purposes. To demonstrate that the proposed approach can be used as a pre-trained method for other tasks, the sentiment datasets are used to measure the classification accuracy and investigate similarities of results from all classifiers.

Field of Study:	Computer Science and Information Technology	Student's Signature .....
Academic Year:	2021	Advisor's Signature .....

ACKNOWLEDGEMENTS

It is a great pleasure to express my gratitude to everyone who made it possible for me to finish my thesis.

This project would not have been possible without Associate Professor Dr. Suphakant Phimoltares for the continuous support of my study, research, and specialized knowledge. Especially, I would like to thank him for his writing guidance that makes this thesis understandable from an outsider's perspective.

I should also thank to all my thesis committee members: Professor Chidchanok Lursinsap and Dr. Prem Junsawang for their comments and suggestions.

Moreover, I am very grateful to thank to the help of Assistant Professor Sasipa Panthuwadeethorn for her illustration ideas.

Finally, I would like to say thanks to my friends for their cheering and encouraging me to finish this thesis study.

Noppadol Kongsumran

TABLE OF CONTENTS

Page

CHAPTER I
INTRODUCTION

1.1 Background and Rationale

In this era, internet services like online shopping, online booking, or online food ordering have a significant impact on the new generation people. These services collect users' interactions, such as ratings and reviews to improve their application. In Thailand, when customers are going to buy products, they are more likely to read Thai reviews first because Thai people use Thai as a primary language. To indicate which review is more influence on users, Natural Language Processing (NLP) is needed.

To successful Natural Language Processing in Thai, word boundaries problem is needed to be solved. For English, spacebar is a powerful word splitter; it can separate a sentence into an ordered set of words. For example, "I love NLP" can be split into "I / love / NLP". Unfortunately, Thai does not have a word splitter like English. Thai sentences are written consecutively without any spaces so that the tokenizer is used as a process to split the sentence into an ordered set of words. For example, "จังหวัดเชียงรายตั้งอยู่ตอนเหนือสุดของประเทศไทย" can be broken into "จังหวัดเชียงราย / ตั้ง / อยู่ / ตอนเหนือ / สุด / ของ/ ประเทศไทย" or "จังหวัด / เชียงราย / ตั้งอยู่ / ตอนเหนือ / สุด / ของ / ประเทศไทย". But the real challenge comes when the Tokenizer is facing with complicated sentence, e.g., "ฉันนั่งตากลมอยู่ริมตลิ่ง". Such sentence can be broken into "ฉัน / นั่ง / ตาก / ลม / อยู่ / ริม / ตลิ่ง" or "ฉัน / นั่ง / ตา / กลม / อยู่ / ริม / ตลิ่ง" where the meanings of these two sets of words are different.

A famous python package for Thai NLP named "pythainlp" [1] provides seven Thai tokenizers: newmm, newmm-safe, longest, icu, nercut, attacut, and deepcut. These seven tokenizers use different algorithms to split Thai sentences. A document of pythainlp in version 2.0 [2] shows that two tokenizers produce different results from the same sentence.

Moreover, many research papers show different results from different tokenizers. Domingo et al. [3], Park et al. [4] , and Kim et al. [5] studied neural machine translation quality with different tokenization techniques compared by BLEU score, TER score, and other evaluation scores that are related to the dataset.

1.2 Research Objectives

To develop encoding methods to solve Thai word boundaries in NLP approach. With these methods, multiple different sets of words from a sentence can be perceived in the same meaning by a neural network.

1.3 Scope of the work

1. The proposed methods are designed to support Thai language.
2. Special token is needed to be replaced in preprocessing step.
3. Default versions of tokenizers from pythainlp [1] are considered in this research.
4. The maximum number of tokens is defined to the training dataset.
5. Synonyms and other similar words are not considered.

1.4 Expected Outcomes

Encoding methods that can solve Thai word boundary problem. These methods can be applied in other problems properly.

CHAPTER II

LITERATURE REVIEW

In this literature review, many related researchers attempt to solve an image classification task rather than to solve the formulated problem. However, there are two similar concepts that can be applied to the NLP. First, a set of words can be formed into a vector where dimension is the same as that of image vector and dimension meaning has closely resembled. This also means that a set of words and an image can be considered as a vector that can be used in any classification model. Second, loss function of similarity and identification techniques can be applied to NLP problem where the input of cost function is a vector from the first concept. The related works corresponding to both similarity and identification techniques are described as follows.

2.1 Similarity Technique

G. Koch, et al. [6] proposed Siamese neural networks. This method solves image recognition problems using convolutional neural networks (CNN). This Siamese technique applies a single CNN to two images and computes its feature distribution. Using two feature vectors, this architecture computes a similarity score from a merged image vector and calculates loss using cross-entropy function. The results of this method also provided a similar ranking from two inputs.

2.2 Identification Technique

A study from F. Schroff, et al. [7] named FaceNet: A Unified Embedding for Face Recognition and Clustering was proposed as a new clustering method. This FaceNet uses CNN architecture followed by L2 normalization to produce a vector from a face image. Then this model is applied to every face in the dataset to create multiple clusters of unique face identities. The special part of this technique is triplet loss. Triplet loss computes the cost function from three images in which a distance between two images in the same person of the group must be lower than a distance between two images

from the different persons of the group. The proposed system yielded high performance in the clustering task among eight million faces.

Moreover, there is an enhanced version of triplet loss named triplet hard loss [8]. In this triplet hard loss, instead of creating a group of three images and computing cost function, it permutes pairs of input in a mini-batch. Only a maximum distance between a pair of images in the same class and a minimum distance between a pair of images in the different classes are considered to compute cost value. This research showed the superior performance of re-identification task of 1500 persons. The study also indicated that using non-Euclidean distance gives a stable performance.

2.3 Classification Technique

After a latent vector of sentences is obtained using the proposed method based on the concepts mentioned above, the method must be tested. One way that can be used is connecting the output from the proposed method with the classification task due to its simplicity and interpretation.

The classification model called Support Vector Machine (SVM) is used to assign a class from a given vector. In this study, a linear layer without any activation function, without kernel function, and hinge loss is put on top of the proposed encoders to simulate SVM on neural networks. By doing this, only a linearly separable dataset can be solved because its behavior involves with using a straight line to classify the data.

In a hypothesis that there might be some gap to draw a straight line when data are projected on higher dimensions, kernel trick can be used to get this result. One method that can be used in this situation is using Random Fourier Features [9], which are transformed from the original data. This research showed good accuracy with lower training and testing time.

2.4 Tokenization Technique

Deepcut [10] and Attacut [11] are two remarkable tokenizers in this thesis. Deepcut is a state-of-the-art technique using characters embedding with 1d-convolutional network and predicting the first character of words in a sentence while Attacut proposed using syllable boundaries instead of using word boundaries. Moreover, Attacut model follows Deepcut CNN architecture with a smaller convolution size and uses the concatenation of character and syllable embeddings as the input. Attacut model also uses Conditional Random Fields (CRF) to capture sequential information.

In addition to word-based tokenization, Byte-Pair Encoding (BPE) [12] is subword-based tokenization used by well-known NLP model [13]. BPE allows all characters of a word in the dataset as an individual symbol, then chooses two adjacent symbols which frequently occur as a new symbol. After that, the individual symbols are replaced by those new frequent symbols. This replacement process is repeated until the number of new symbols reaches the limit.

2.5 Overall Process

For those previous papers, their scopes do not directly align to NLP. According to the experiments, the properties of each previous work will be used in some particular domain of this research.

Firstly, the clustering model is built and trained using a tokenized sentence as input. Multiple sentences with the same sentence ID represent face images with person identity in FaceNet [7] where triplet hard loss is used to guide model in clustering direction. Secondly, the sentiment classification based on clustering model and Random Fourier Feature [9] are used to test the proposed architecture when using as pre-trained model. For comparison, sentiment classifiers with and without applying the proposed method are created. Similarity metric is computed between two sentiment classifiers. It is used to validate the proposed method after the weights are transferred. Moreover,

accuracy score is also computed to test the method if it can be used for combining with the classification task.

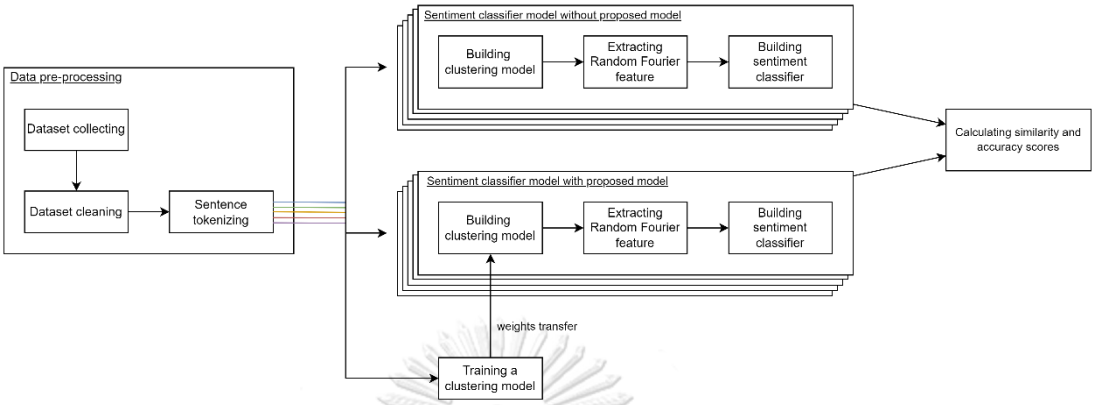


Figure 1. Overall process of this thesis.

CHAPTER III

METHODS

This study proposed a neural architecture designed for sentence encoder based on clustering technique. The encoder is on top with a linear layer followed by L2 normalization and triplet hard loss. Triplet loss is used to guide neural networks to group sets of words from the same sentence together and distinguish different sentences. Bidirectional Long Short-Term Memory (Bi-LSTM) and DistilBERT are used for comparative purpose in this proposed encoder. These two types of encoders represent traditional and current trending methods.

3.1 Bidirectional LSTM

Bidirectional LSTM is a type of Recurrent Neural Network (RNN). Generally, RNN suits a vector of sentence or time-series data such as stock and weather. Each timestep or each token of a sequence is combined with the previous result of the prior timestep and fed into RNN cell. RNN iterates over input from the first to the last token and produces a summary vector after the last timestep is computed. Moreover, RNN has an option to yield the result of every iteration and produce a new sequence to use later in another RNN variant layer.

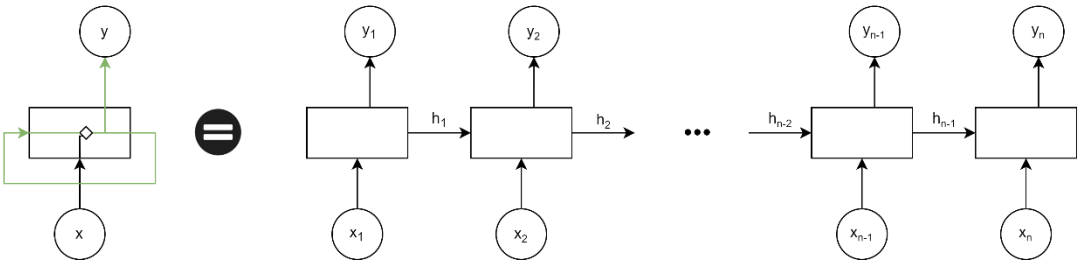


Figure 2. Simplified RNN and unfolded RNN architecture. y_i and h_i are the same value where i represents each token index.

Instead of passing only one value to the next timestep, LSTM passes one more value into the next iteration, called long-term state. This long-term state can forget or update

itself using a normal state. With this mechanism, LSTM cells can still use features from the early iterations at the late iteration. Moreover, bidirectional extension is used to capture both backward and forward information.

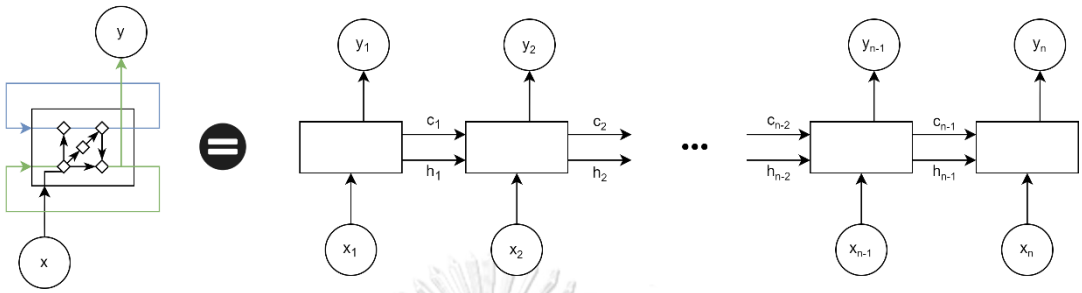


Figure 3. Simplified LSTM and unfolded LSTM architecture. h_i and c_i line represent short and long-term state; y_i and h_i are the same value where i represents each token index.

To summarize, Bi-LSTM represents an RNN architecture with an additional passing state named long-term. It iterates the sentence in both forward and backward parts to get the most information possible.

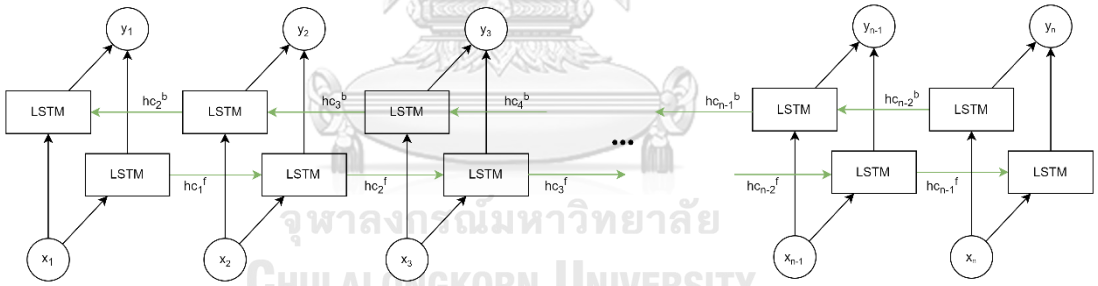


Figure 4. Unfolded Bi-LSTM; $hc_i^{\{f,b\}}$ line represents long short-term state where i represents each token index and f/b represent forward/backward direction.

3.2 Transformer

Before the transformer was invented, machine translation or sequence-to-sequence task was one of the challenging problems. Typically, RNN-based architecture can be used to encode and decode the sentence. Since the input token length might not be equal to that of the output, the sequence result of the encoder's output cannot directly pass into the decoder due to dimensional issues. With this reason, summarized data are needed to pass into every RNN iteration of the decoder instead. By using this, a

bottleneck problem occurs when single data might not be able to represent every sentence's meaning.

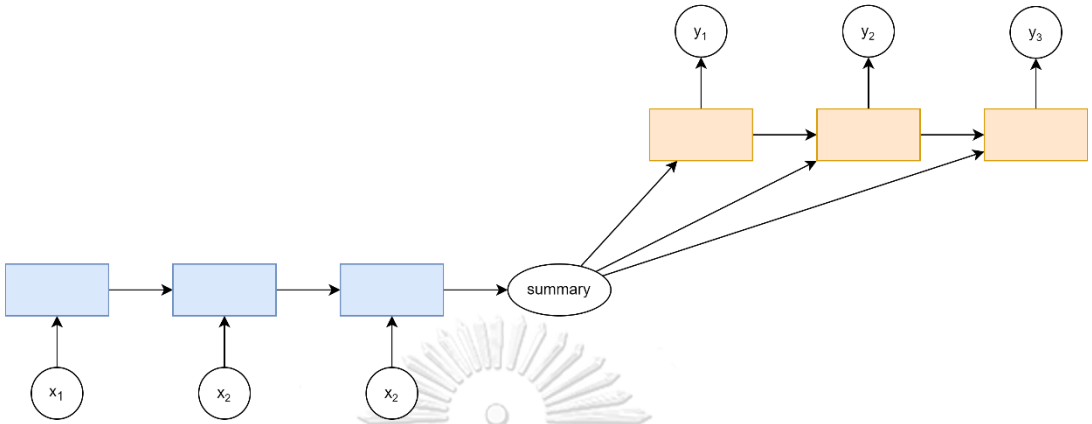


Figure 5. RNN based sequence-to-sequence model; the encoder compresses a sequence into a single vector. (Blue blocks and orange blocks represent RNN encoder and RNN decoder, respectively)

Transformer [14] is used to solve this bottleneck problem. It also increases the parallelization ability compared with RNN variants. Transformer uses encoder-decoder model as same as RNN model but replaces iteration process with attention module.

The attention module requires three parameters query (Q), key (K), and value (V) where these inputs are in a form of 2D matrix. This module calculates the relationship between tokens by multiplying Q and K^T then normalization process (softmax) is applied to get an attention score. After that, a final vector is obtained by multiplication between attention score and V. To get more information, multi-head attention is proposed. By splitting Q, K, and V into multiple parts and feeding them into different attention modules.

In this research, only encoder part of the transformer is considered because there is no text generation in this research. In encoder transformer, attention layer process is calculated by passing embedded sequence into all three inputs of attention. At the final vector of module, skip connection is added, and normalization is applied. After that, time distributed feed-forward neural networks are applied to calculate position-wise

information then skip connection is added, and normalization is applied as well. Moreover, this attention layer can be stacked for more complex meaning extraction.

3.3 BERT and DistilBERT

Bidirectional Encoder Representations from Transformers (BERT) [15] is a method of learning a representation of language from only encoder parts of the transformer. BERT is originally trained on language model task, by predicting the 15% missing input tokens from the remaining tokens. As a result, after BERT is trained, it can be used for a smaller task like a pre-trained model.

DistilBERT [16] uses a distillation technique to replicate BERT with only 40% fewer parameters and ignore token type embedding, but it can preserve 97% of BERT performance. Instead of calculating loss to maximize the probability of a one-hot value, DistilBERT calculates loss from the difference between the output distribution of itself and BERT.

3.4 Proposed Method

As aforementioned, Bi-LSTM and DistilBERT are used for comparative purpose since the proposed method can be constructed by either Bi-LSTM or DistilBERT. In this study, Bi-LSTM model is made up of a sequential architecture, as shown in Figure 6, which begins with the embedding layer to transform word id into a vector. Then stacked Bi-LSTM layer is applied to extract sentence features. Finally, the feed-forward and L2 norm layer assign sequence features into a cluster. The output dimension of both Bi-LSTM layers and the feed-forward layer is set to 768, while the output dimension of embedding layer is set to 300. On the other hand, DistilBERT model in this study does not use a pre-trained versions from the hub, but only model architecture is used. Like Bi-WLSTM model, DistilBERT model is connected to the feed-forward and L2 norm layer, as shown in Figure 7.

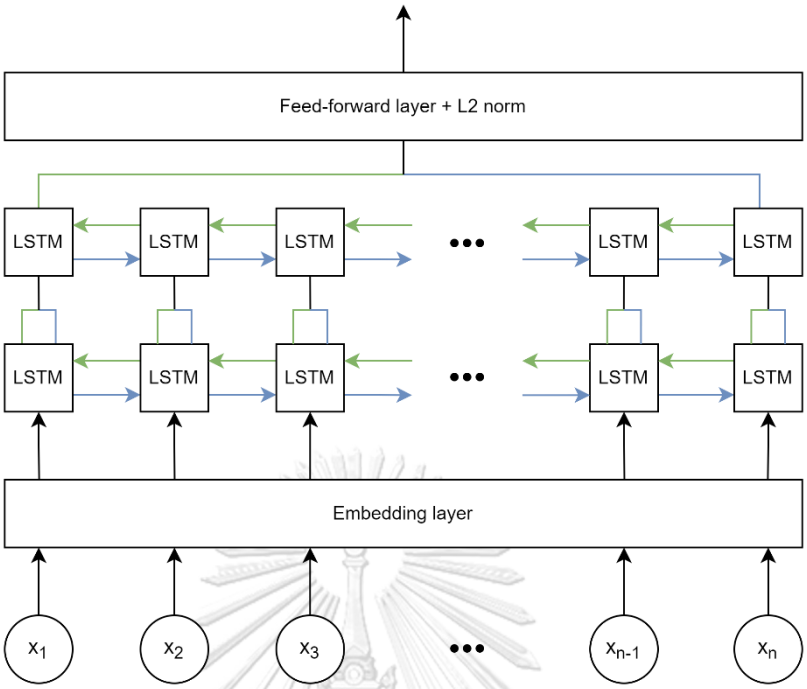


Figure 6. The Bi-LSTM encoder; Embedding layer connected to Bi-LSTM with return sequence and Bi-LSTM with no return sequence; x_1 and x_n indicate first and last tokens of a sentence

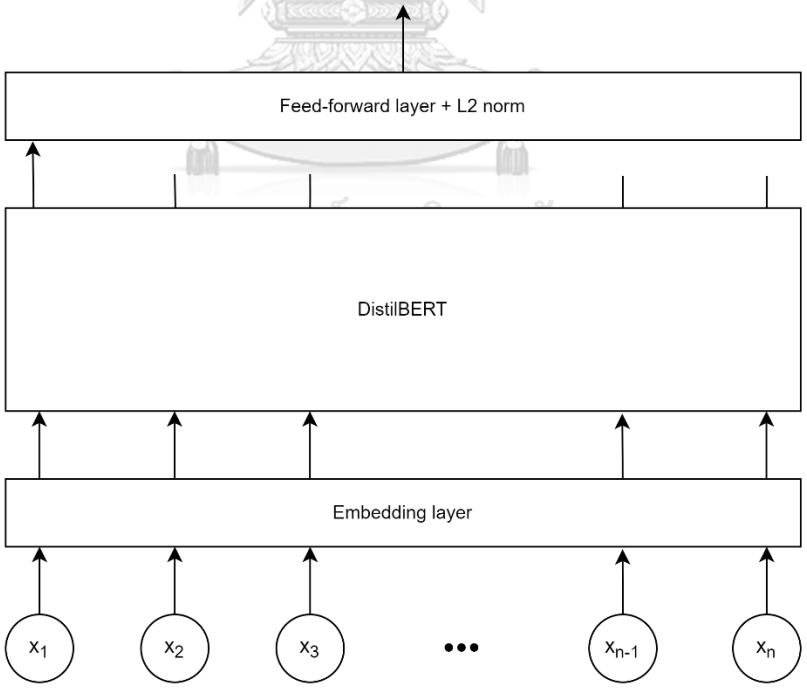


Figure 7. The DistilBERT encoder; The first timestep of output is used; x_1 and x_n indicate first and last tokens of a sentence

3.5 Sentiment Classification

Sentiment classification is a classification task to test whether the proposed method can be employed in other domain problems. The output of both Bi-LSTM encoder and DistilBERT encoder are fed into a process of Random Fourier Feature extraction [9] followed by a linear layer without any activation function. The encoder weights except the last linear layer are frozen, similar to other learning techniques.

CHAPTER IV
IMPLEMENTATION

4.1 Dataset

Two sentiment classification datasets are used for comparative purposes. The first dataset was collected from Google map place reviews and the second dataset was collected from Google play reviews. Google map place reviews are collected from five well-known fast-food restaurants. As a result, 20,000 reviews were collected, resulting in restaurant ratings dataset. For Google play reviews, ten financial and banking applications in Thailand were selected. Then, 2,000 reviews for each application were collected to make dataset size balance among all classes to obtain application ratings dataset. Each dataset was split into three parts, which are 75% for training dataset, 10% for validating dataset, and the 15% for testing dataset.

4.2 Training Process

From the scope of the work in section (1.3), related works that cannot be applied Thai sentence and word were not considered. A special component such as URL, number, emoji, duplicate space, and end of line symbol was replaced using a special token. Note that these tokens were added into custom dictionary in sentence tokenization phase.

Then, each sample was applied with seven tokenizers from pythainlp consisting of newmm, newmm-safe, longest, icu, attacut, deepcut, and nercut. After tokenization was completed, a set of ordered words whose size was greater than 20 and found only once in the training dataset was marked as an unnecessary set and removed afterwards. Since there are seven tokenizers and each tokenizer yields its own result, a training dataset of 15,000 samples was enlarged and 105,000 samples were obtained. The duplicate tokenized set obtained from the same sentence was removed to decrease the size of the whole training set and training time. With this process, a training set finally

contained less than 105,000 samples. Tokenization from this step will be discussed in the last section of chapter 6.

After tokenization step, a big dataset was created where the number of target classes was equal to the number of unique sentences in the training set. In the training step, sentences were fed to the encoder, and then triplet hard loss has calculated the cost to optimize neural networks to the optimal cluster solution. Since a mini-batch is a slice of a shuffled dataset, there was a chance that a batch did not contain two set of ordered words obtained from the same original sentence. To proceed the training step, a custom shuffling must be used to solve the problem, as shown in Figure 8.

The idea of the custom shuffling is to guarantee that each batch contains at least two sets from the same sentence. There are four main parts in this function. First, seven word sets from the same original sentence were relocated. Then, groups of three sets were obtained by extracting three consecutive sets from the streaming sets of words. By doing this, it is certain that there are more than one word set obtained from the same sentence in each group, even though some group may contain two different original sentences. Next, sets in every group was originally shuffled in the dataset. After that, grouping is released to make the dataset ready to generate a mini-batch.

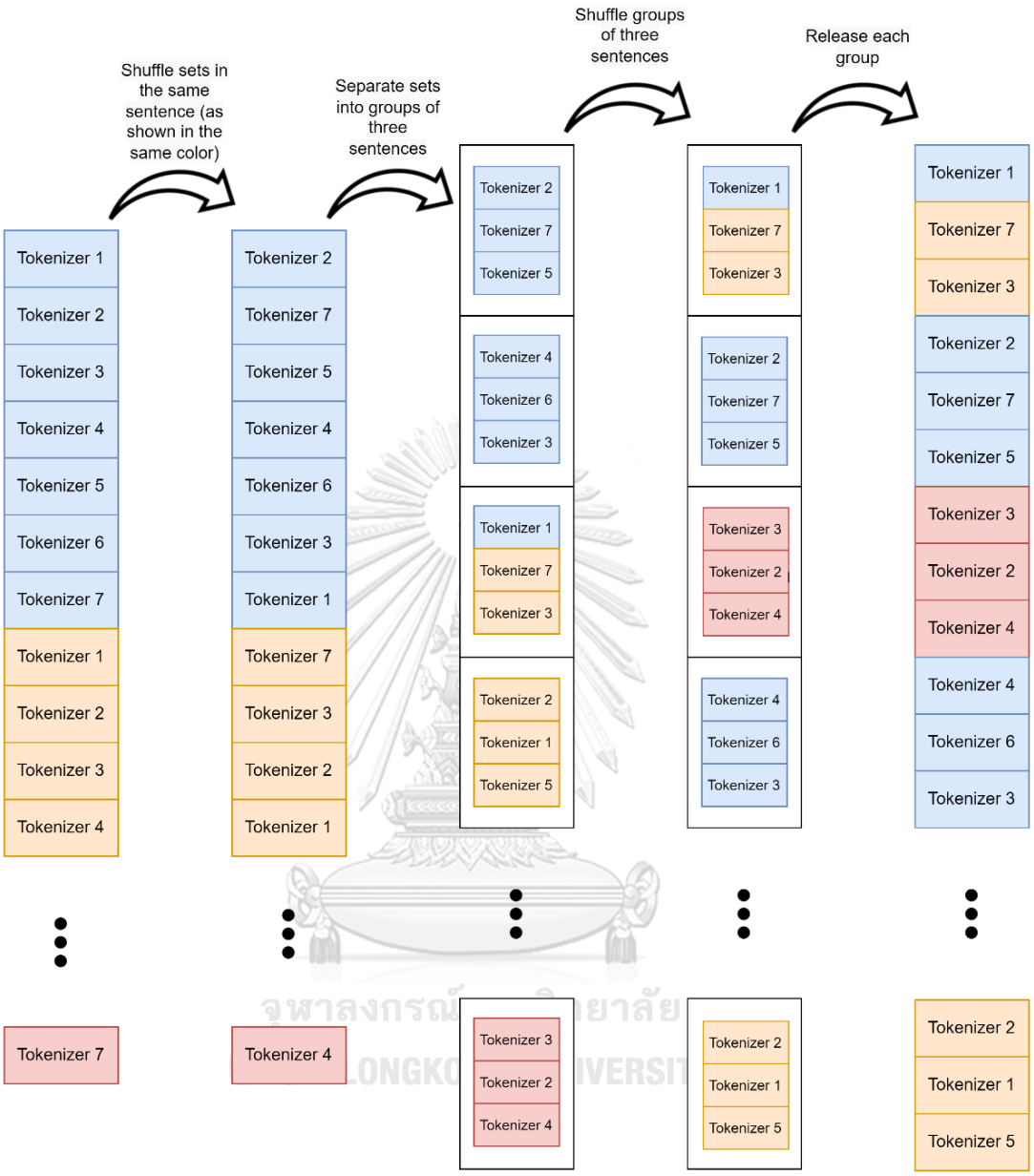


Figure 8. Custom shuffling; Tokenizer n in a cell with a particular color means set n of words obtained from the sentence, using tokenizer n.

After sentence batch was generated and fed to neural networks and loss was applied, validation dataset was fed continuously to indicate whether or not the training neural network is overfitting. Like training dataset, validating dataset also used triplet loss and custom shuffling was needed as well. When validating loss was continuously increased for five times, a stop training signal was sent to neural networks. In this

process, testing dataset was not used to test clustering model. Instead, the proposed model will be tested after sentiment classification model based on this model is finalized. The performance of proposed encoder is shown in Table 1.

Table 1. Triplet hard loss from each encoder model.

	Validation loss	
Encoders / Dataset	Bi-LSTM	DistilBERT
Restaurant ratings	0.1630	0.1107
Application ratings	0.1748	0.1104

4.3 Result Visualization

To visualize the relationship results of sets of words, a dimensionality reduction technique is needed. In this experiment, a technique, namely, t-distributed Stochastic Neighbor Embedding (t-SNE), was selected to identify the data patterns with a good-looking visualization. Unlike Principal Component Analysis (PCA), t-SNE can align non-linear data into the desired vector space by focusing only on a limited distance between points in the dataset. t-SNE visualization of a first mini-batch of the testing set of each dataset can be shown in Figures 9-12. Note that different sets of words from the same sentence are in the same color.

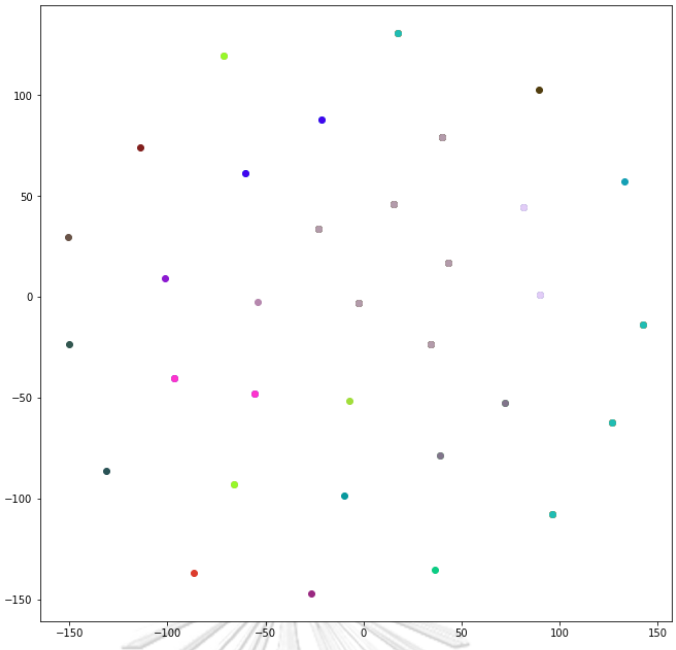


Figure 9. t-SNE visualization of bi-LSTM encoder output from restaurant ratings dataset; Same dot color represents the same sentence.

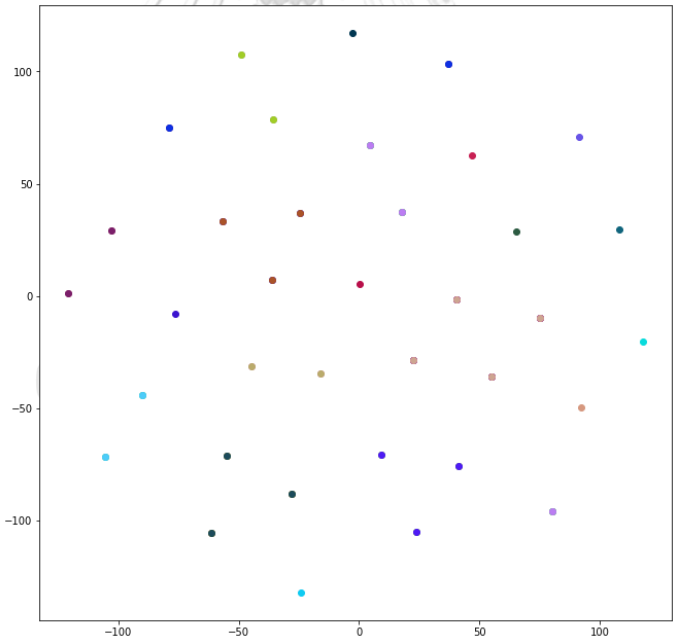


Figure 10. t-SNE visualization of DistilBERT encoder output from restaurant ratings dataset; Same dot color represents the same sentence.

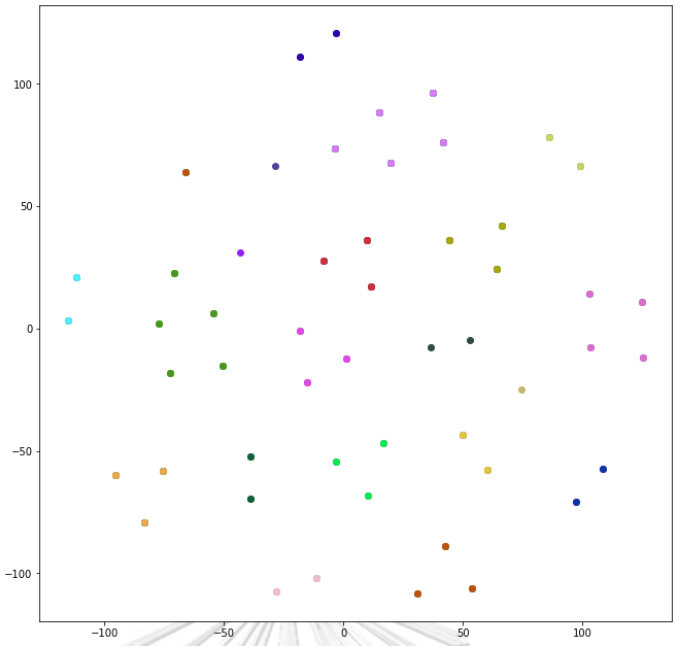


Figure 11. t-SNE visualization of bi-LSTM encoder output from application ratings dataset; Same dot color represents the same sentence.

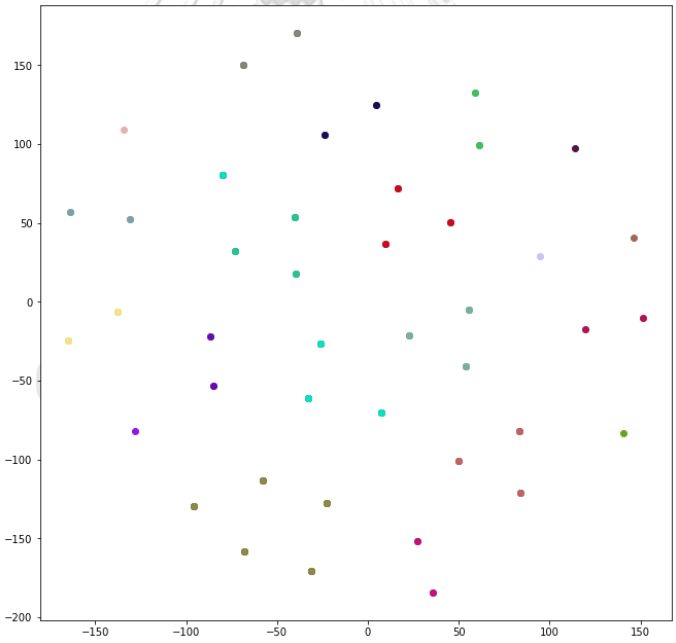


Figure 12. t-SNE visualization of DistilBERT encoder output from application ratings dataset; Same dot color represents the same sentence.

CHAPTER V

EXPERIMENTAL RESULTS

As aforementioned, two proposed encoders are used for comparative purposes. These two encoders were trained separately. Then, their outcome vectors were used as the input of the sentiment classifier. In this classifier, Random Fourier Feature technique [9] was used to transform vector from the proposed model into a feature vector. After that, a linear classifier on top with hinge loss was performed to assign the proper class.

Before the classification model was compiled, encoder layers were frozen and marked as weights were not adjusted. For the upper part of the model, the weights were prepared to be ready for a new training task, as shown in Figure 13-14.

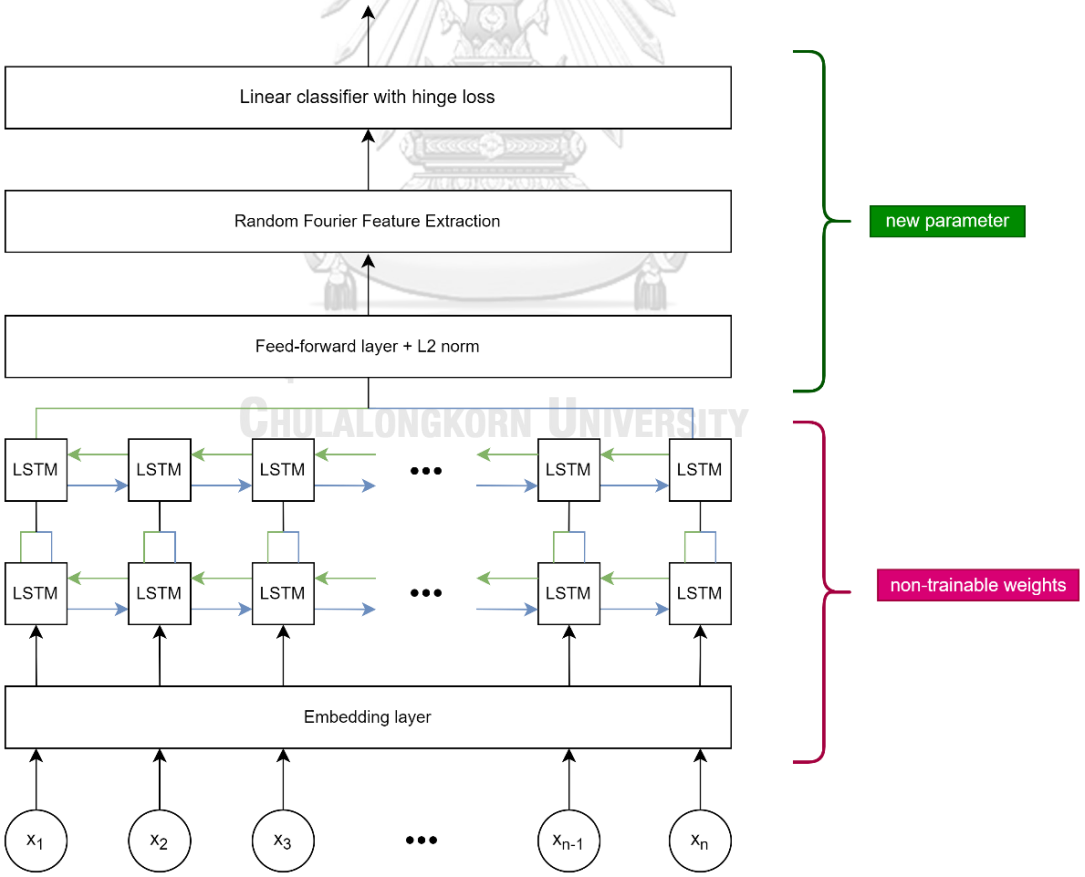


Figure 13. Classifier model extended from Bi-LSTM encoder. The proposed encoder part consists of weights which are not trained in this process while the weights in the upper parts are trained for the classification task.

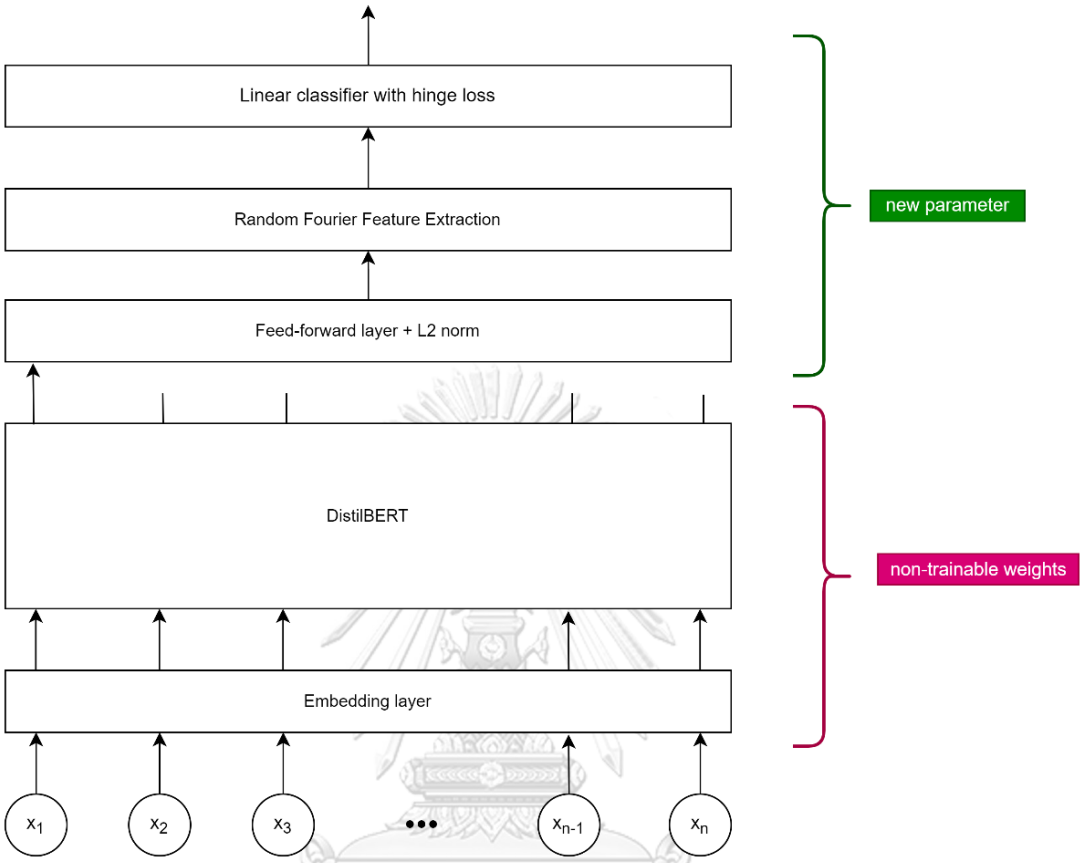


Figure 14. Classifier model extended from DistilBERT encoder. The proposed encoder part consists of weights which are not trained in this process while the weights in the upper parts are trained for the classification task.

Tables 2-5 present the proposed encoders' performance in two datasets. Each cell represents the score of similarity between the results of two different tokenizers. The similarity of the classification results of a specific pair of two tokenizers without using the proposed encoders is represented by the red shaded cell in the upper triangular area of the matrix. In contrast, the similarity of the results obtained by the proposed encoders is represented by the green shaded cell in the lower triangular area. The light color cell presents the lowest similarity score, while the darker color presents more similarity.

Table 2. Similarity scores with and without applying Bi-LSTM-based encoder to restaurant ratings dataset.

	newmm	newmm-safe	longest	nercut	icu	attacut	deepcut
newmm		0.209302	0.430861	0.351980	0.181018	0.477058	0.357322
newmm-safe	0.823696		0.452546	0.493400	0.630735	0.272784	0.357951
longest	0.795412	0.834695		0.398806	0.509114	0.528284	0.338466
nercut	0.727844	0.735387	0.728473		0.518228	0.334695	0.372093
icu	0.783784	0.808925	0.821182	0.768699		0.320867	0.382464
attacut	0.821182	0.884664	0.891578	0.771213	0.874293		0.351980
deepcut	0.826838	0.886235	0.895349	0.774041	0.890006	0.960088	

Table 3. Similarity scores with and without applying DistilBERT-based encoder to application ratings dataset.

	newmm	newmm-safe	longest	nercut	icu	attacut	deepcut
newmm		0.630107	0.565682	0.521999	0.546826	0.527970	0.442489
newmm-safe	0.778441		0.593338	0.636706	0.538655	0.496857	0.496857
longest	0.800126	0.847580		0.538655	0.668762	0.571025	0.523570
nercut	0.792898	0.854808	0.842238		0.523256	0.484601	0.461974
icu	0.826524	0.804840	0.847894	0.814582		0.735072	0.587366
attacut	0.783469	0.883721	0.841923	0.855437	0.827153		0.556882
deepcut	0.832181	0.782841	0.793840	0.782527	0.835009	0.820553	

Table 4. Similarity scores with and without applying Bi-LSTM-based encoder to restaurant ratings dataset.

	newmm	newmm-safe	longest	nercut	icu	attacut	deepcut
newmm		0.722960	0.814532	0.657930	0.542468	0.710683	0.791639
newmm-safe	0.822163		0.733908	0.540146	0.533510	0.729927	0.719973
longest	0.831785	0.746184		0.580292	0.631387	0.745189	0.835435
nercut	0.555740	0.491374	0.616788		0.483411	0.498341	0.548772
icu	0.621101	0.666224	0.597545	0.372263		0.597545	0.527870
attacut	0.675182	0.663902	0.645322	0.635700	0.577306		0.773723
deepcut	0.730259	0.684472	0.727273	0.609489	0.584273	0.735236	

Table 5. Similarity scores with and without applying DistilBERT-based encoder to application ratings dataset.

	newmm	newmm-safe	longest	nercut	icu	attacut	deepcut
newmm		0.572993	0.642336	0.457863	0.899867	0.582946	0.515926
newmm-safe	0.822827		0.664698	0.573325	0.615793	0.561048	0.552090
longest	0.704711	0.691772		0.559390	0.621433	0.572329	0.579628
nercut	0.894492	0.793298	0.690445		0.494691	0.472130	0.592568
icu	0.708693	0.714665	0.624751	0.676841		0.534837	0.524884
attacut	0.748175	0.782681	0.662906	0.719642	0.706038		0.524884
deepcut	0.894492	0.817518	0.703052	0.876576	0.711679	0.758461	

With applying proposed method, top three highest scores from each table were (attacut vs deepcut) / (longest vs deepcut) / (longest vs attacut) from Table 2, (newmm-safe vs attacut) / (nercut vs attacut) / (newmm-safe vs nercut) from Table 3, (newmm vs newmm-safe) / (newmm vs longest) / (newmm-safe vs longest) from Table 4, (newmm vs nercut) / (newmm vs deepcut) / (nercut vs deepcut) from Table 5. With these highest scores from each table, it can be noticed that three tokenizers had some related mechanism such that a triangular connection can be formed.

Without applying the proposed method, top three highest scores from each table were (newmm-safe vs icu) / (longest vs attacut) / (nercut vs icu) from Table 2, (icu vs attacut) / (longest vs icu) / (newmm-safe vs nercut) from Table 3, (longest vs deepcut) / (newmm vs longest) / (newmm vs deepcut) from Table 4, (newmm vs icu) / (newmm vs longest) / (newmm-safe vs longest) from Table 5. From these results, only Table 4 shows a triangular connection among top three pairs of tokenizers as same as the model with applying proposed method.

Moreover, the number of dataset ratings is reduced from 5 classes to 3 classes for further investigation. These classes of each dataset include negative, neutral, and positive labels. Negative label denotes 1-star and 2-star ratings, neutral label denotes 3-star rating, and positive label denotes 4-star and 5-star ratings. Note that the datasets with new labels are called feelings datasets. Again, the classifiers were newly created from Bi-LSTM and DistilBERT model to support the new set of classes, then the comparative results were collected to show the performance of applying the proposed method as shown in Tables 6-9.

Table 6. Similarity scores with and without applying Bi-LSTM-based encoder to restaurant feelings dataset.

	newmm	newmm-safe	longest	nercut	icu	attacut	deepcut
Newmm		0.712445	0.685418	0.545569	0.681332	0.562854	0.659962
newmm-safe	0.905720		0.701131	0.435261	0.722816	0.673790	0.681332
Longest	0.956317	0.890635		0.449403	0.613451	0.660591	0.631992
Nercut	0.956317	0.877750	0.987115		0.495600	0.363608	0.472344
Icu	0.877121	0.913576	0.869893	0.858894		0.634507	0.739472
Attacut	0.910434	0.921119	0.905720	0.894092	0.918605		0.687618
Deepcut	0.947203	0.914205	0.949717	0.937461	0.909491	0.949717	

Table 7. Similarity scores with and without applying DistilBERT-based encoder to application feelings dataset using modified class labels.

	newmm	newmm-safe	longest	nercut	icu	attacut	deepcut
Newmm		0.84915	0.86895	0.84884	0.82401	0.85261	0.8303
newmm-safe	0.93118		0.84192	0.82652	0.78976	0.84444	0.82118
Longest	0.9648	0.90855		0.8573	0.84601	0.84444	0.82621
Nercut	0.89095	0.92426	0.87492		0.82055	0.83344	0.8149
Icu	0.93275	0.91169	0.93149	0.87681		0.82275	0.8061
Attacut	0.89912	0.92489	0.88561	0.89912	0.91326		0.85387
Deepcut	0.9318	0.91138	0.92646	0.87775	0.93935	0.92175	

Table 8. Similarity scores with and without applying Bi-LSTM-based encoder to restaurant feelings dataset using modified class labels.

	newmm	newmm-safe	longest	nercut	icu	attacut	deepcut
Newmm		0.3497	0.36404	0.86762	0.27505	0.87956	0.87226
newmm-safe	0.96881		0.63537	0.39118	0.85103	0.42634	0.44559
Longest	0.93033	0.93597		0.66954	0.60982	0.70073	0.73125
Nercut	0.89482	0.90345	0.87757		0.31652	0.84605	0.83411
Icu	0.8862	0.88587	0.91871	0.87326		0.35634	0.38222
Attacut	0.83776	0.83046	0.81188	0.84506	0.85435		0.86463
Deepcut	0.88089	0.87492	0.89383	0.87027	0.93298	0.88819	

Table 9. Similarity scores with and without applying DistilBERT-based encoder to application feelings dataset using modified class labels.

	newmm	newmm-safe	longest	nercut	icu	attacut	deepcut
Newmm		0.854015	0.829794	0.786994	0.872926	0.823822	0.874585
newmm-safe	0.988388		0.859323	0.794957	0.869940	0.814532	0.860650
Longest	0.982747	0.979429		0.763769	0.840411	0.836098	0.864300
Nercut	0.947246	0.943265	0.939615		0.804247	0.766092	0.787326
Icu	0.952887	0.949900	0.952223	0.928666		0.829131	0.879894
Attacut	0.956204	0.954877	0.951559	0.929993	0.958195		0.861314
Deepcut	0.956536	0.955209	0.952555	0.930325	0.950564	0.966490	

With applying proposed method, top three highest scores from each table were (longest vs nercut) / (newmm vs nercut) / (newmm vs longest) from Table 6, (newmm vs longest) / (icu vs deepcut) / (newmm vs icu) from Table 7, (newmm vs newmm-safe) / (newmm-safe vs longest) / (icu vs deepcut) from Table 8, (newmm vs newmm-safe) / (newmm vs longest) / (newmm-safe vs longest) from Table 9. From these results, the triangular connection was created only in Table 6 and Table 9.

When considering the case without applying proposed method, top three highest scores from each table were (icu vs deepcut) / (newmm vs newmm-safe) / (newmm-safe vs longest) from Table 6, (newmm vs longest) / (attacut vs deepcut) / (longest vs nercut) from Table 7, (newmm vs attacut) / (newmm vs deepcut) / (newmm vs nercut) from Table 8, (icu vs deepcut) / (newmm vs deepcut) / (icu vs newmm) from Table 9. Among these results, only Table 9 performed triangular connection.

Refer to Tables 2-9; after applying the proposed method, average improvement of similarity can be calculated and displayed as shown in Table 10. In this table, the aggregate results from eight tables above show that this proposed method can improve results similarity among seven different tokenizers.

Table 10. Average similarity improvement.

Dataset	Encoder	Classes label	% Improvement
Restaurant ratings	Bi-LSTM	1-star, 2-star, 3-star, 4-star, 5-star	43.0173
Restaurant ratings	DistilBERT	1-star, 2-star, 3-star, 4-star, 5-star	26.6663
Application ratings	Bi-LSTM	1-star, 2-star, 3-star, 4-star, 5-star	-0.0621
Application ratings	DistilBERT	1-star, 2-star, 3-star, 4-star, 5-star	17.0860
Restaurant feelings	Bi-LSTM	negative, neutral, positive	30.6695
Restaurant feelings	DistilBERT	negative, neutral, positive	7.8793
Application feelings	Bi-LSTM	negative, neutral, positive	28.2219
Application feelings	DistilBERT	negative, neutral, positive	1.2156

In another point of view, the proposed model also yielded higher accuracy than the original model that did not use the proposed method. Tables 11-12 show accuracy results on the ratings dataset with five classes and Tables 13-14 show accuracy results on the feelings dataset with three classes.

Table 11. Accuracies of sentiment classifiers using the proposed method compared with classifiers that do not use the proposed method on the restaurant ratings dataset.

	Bi-LSTM		DistilBERT	
	proposed model	original model	proposed model	original model
newmm	0.45	0.35	0.49	0.36
newmm-safe	0.45	0.19	0.49	0.31
longest	0.45	0.22	0.49	0.51
nercut	0.46	0.28	0.50	0.39
icu	0.47	0.18	0.48	0.49
attacut	0.47	0.22	0.50	0.46
deepcut	0.48	0.27	0.48	0.51

This table presents an accuracy comparison on four classifiers with or without applying the proposed method and using Bi-LSTM or DistilBERT encoder. By using proposed method, the volatility was significantly less than the original version in both encoders. Moreover, in terms of accuracy, the proposed model outperformed the original model except DistilBERT encoder when trained on icu tokenizer.

Table 12. Accuracies of sentiment classifiers using the proposed method compared with classifiers that do not use the proposed method on the application ratings dataset.

	Bi-LSTM		DistilBERT	
	proposed model	original model	proposed model	original model
newmm	0.54	0.25	0.67	0.52
newmm-safe	0.60	0.26	0.63	0.68
longest	0.52	0.28	0.69	0.56
nercut	0.40	0.27	0.67	0.50
icu	0.50	0.34	0.57	0.54
attacut	0.51	0.26	0.60	0.52
deepcut	0.51	0.26	0.67	0.49

For Table 12, the proposed model outperformed most of the original models except DistilBERT encoder when trained on newmm-safe tokenizer.

Table 13. Accuracies of sentiment classifiers using the proposed method compared with classifiers that do not use the proposed method on the restaurant feelings dataset.

	Bi-LSTM		DistilBERT	
	proposed model	original model	proposed model	original model
newmm	0.76	0.50	0.76	0.79
newmm-safe	0.72	0.61	0.77	0.78
longest	0.76	0.46	0.76	0.79
nercut	0.76	0.36	0.76	0.79
icu	0.71	0.57	0.75	0.77
attacut	0.74	0.61	0.77	0.78
deepcut	0.75	0.57	0.77	0.78

Table 13 shows the performance of models on feelings dataset with only three classes. The proposed model did not provide high volatility as the previous model and the accuracies of the proposed DistilBERT model are not higher than those of the original model.

Table 14. Accuracies of sentiment classifiers using the proposed method compared with classifiers that do not use the proposed method on the application ratings dataset.

	Bi-LSTM		DistilBERT	
	proposed model	original model	proposed model	original model
newmm	0.70	0.33	0.78	0.80
newmm-safe	0.69	0.73	0.79	0.77
longest	0.72	0.57	0.78	0.75
nercut	0.68	0.37	0.76	0.75
icu	0.73	0.74	0.77	0.79
attacut	0.67	0.40	0.77	0.76
deepcut	0.70	0.41	0.77	0.79

According to this table, the proposed model did not perform better than the original model in several situations, which are combining newmm-safe or icu with Bi-LSTM encoder and combining newmm, icu, or deepcut with DistilBERT encoder.

To summarize the results, the proposed models can improve five-class classification performance in terms of accuracies obtained from the restaurant ratings dataset, though overall classification results remain unsatisfactory. However, the models give acceptable performance when testing with the application ratings dataset. The proposed models achieve good results in both feelings datasets. Despite this, most of the results from the proposed DistilBERT model are slightly less accurate than the original DistilBERT model. This may be caused by DistilBERT's ability to align itself into the random Fourier feature condition when performing three-class classification. Furthermore, it is remarkably noticed that using the proposed models will increase the consistency of the accuracies. Particularly, when using the proposed models, choosing a tokenizer has a less impact on the classification results.

CHAPTER VI
DISCUSSION

6.1 Effect of surrounding words

According to Chapter I (Background and Rationale), a problem sentence “ฉันนั่งตากลมอยู่ริมตลิ่ง” contains only one section that makes seven tokenizers yield different sets of words. This section is small when compared to total length of the entire sentence. With this reason, a grouping part of proposed method might not be considered this section as important enough to get the different classification result at the end of whole process. A neural network might mark this section as low priority and produce a grouping vector from only the same remaining section. In this chapter, the influence of the surrounding word that affects the grouping section is experimented and discussed in detail.

Surrounding word extraction was conducted from seven set of words obtained by seven tokenizers from one original sentence, the same duplicate sets from different tokenizers will be removed before extraction process. The process was applied to every pair of word sets. Every common consecutive token whose length is more than three were located and extracted from any pair of sets. The example of the extraction process is illustrated in Figure 15.

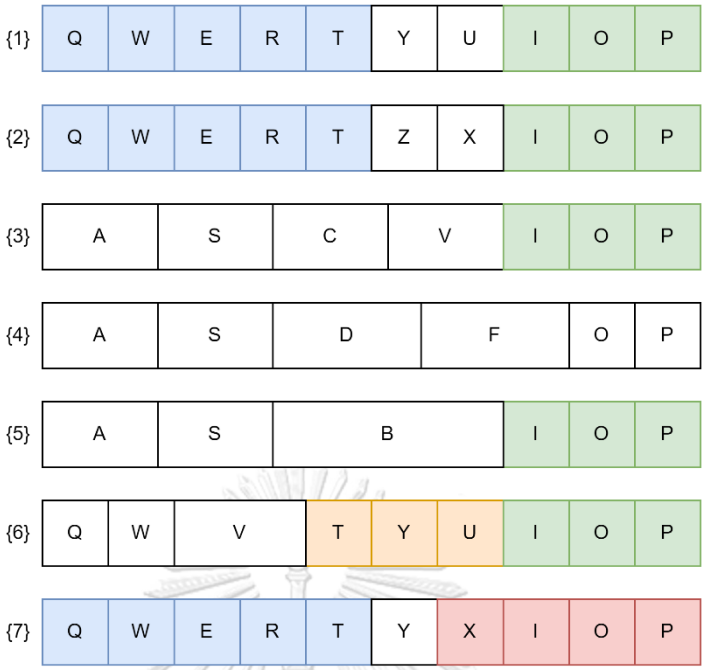


Figure 15. Common consecutive tokens selection from one unique sentence. (QWERT and IOP is selected as consecutive tokens; TYU in in {1} and {6} is selected; OP and AS in {4} is not select because length is less than three.)

This process was applied to datasets of both restaurant and application ratings. It produces 9,292 sets of common consecutive tokens from restaurant ratings dataset and 3,014 sets of common consecutive tokens from application ratings dataset. After that, the classification process was used to test whether the surrounding words affected the results.

The testing process was conducted on two models with and without using proposed encoder. All sets of common consecutive tokens were lengthened by padding with blank tokens until the size of sets was equal to the original input length. Only Bi-LSTM encoder was proceeded in this experiment because DistilBERT encoder can generate one more input named positional embedding which might be bias when the number of common consecutive tokens is not large as the original one. The weights of both proposed model and original model were transferred from the models in the last chapter. The results are illustrated in Tables 15-20.

Table 15. Comparison of classification accuracies between using original sets of tokens and using sets of common consecutive tokens on restaurant ratings dataset.

	proposed model		original model	
	Original set of tokens	Set of common consecutive tokens	Original set of tokens	Set of common consecutive tokens
newmm	0.45	0.33	0.35	0.25
newmm-safe	0.45	0.32	0.19	0.16
longest	0.45	0.33	0.22	0.21
nercut	0.46	0.39	0.28	0.24
icu	0.47	0.36	0.18	0.23
attacut	0.47	0.35	0.22	0.26
deepcut	0.48	0.4	0.27	0.13

Table 16. Comparison of accuracies among seven tokenizers on restaurant ratings dataset split by the length of common consecutive tokens (green row indicates classification accuracies on the original sets of tokens while another row indicates classification accuracies on the sets of common consecutive tokens).

	1-10	11-20	21-30	31-40	41-50	51-60	61-70	70-80
newmm	32.9130	32.6636	31.2080	31.2080	27.0270	61.5384	33.3333	0.0000
	23.9002	29.1666	33.5570	40.3100	29.7297	30.7692	16.6666	33.3333
newmm-safe	32.2290	32.2916	36.5771	33.3333	32.4324	23.0769	16.6666	33.3333
	14.5520	17.3363	23.8255	27.1317	27.0270	30.7692	16.6666	33.3333
longest	32.3095	33.9285	31.8791	37.9844	29.7297	38.4615	16.6666	66.6666
	20.1448	22.5446	27.5167	26.3565	24.3243	30.7692	16.6666	33.3333
nercut	38.5729	40.9226	40.2684	41.0852	37.8378	46.1538	16.6666	66.6666
	26.0729	16.8154	18.7919	8.5271	18.9189	7.6923	16.6666	33.3333
icu	35.7296	36.3839	40.6040	38.7596	37.8378	30.7692	16.6666	66.6666
	22.7736	23.8839	24.8322	23.2558	32.4324	46.1538	16.6666	33.3333
attacut	34.4688	36.5327	37.9194	37.9844	37.8378	38.4615	16.6666	66.6666
	27.3068	23.5119	23.1543	17.0542	27.0270	23.0769	0.0000	33.3333
deepcut	39.9812	40.8482	44.9664	39.5318	37.8378	30.7692	16.6666	66.6666
	12.2854	12.5000	16.4429	19.3798	16.2162	23.0769	16.6666	0.0000

Table 17. Comparison of accuracies among seven tokenizers on restaurant ratings dataset split by the length ratio of common consecutive tokens (green row indicates classification accuracies on the original sets of tokens while another row indicates classification accuracies on the sets of common consecutive tokens)

	0-10	10-20	20-30	30-40	40-50	50-60	60-70	70-80	80-90	90-100
newmm	27.7329	28.5123	34.9333	35.1468	35.8355	35.7558	38.3458	40.3636	36.8876	31.5068
	22.5094	26.3946	24.9333	25.2158	25.8266	27.9069	26.0651	27.4545	24.4956	24.6575
newmm-safe	27.2482	30.0103	33.5333	35.0604	35.0312	33.9147	39.0977	36.1818	33.4293	30.1369
	14.0549	15.7541	14.0000	14.9395	14.8346	17.4418	18.7969	17.0909	14.9855	13.6986
longest	28.2714	31.0950	35.2666	34.7150	32.1715	32.4612	38.4711	34.9090	35.7348	31.5068
	14.7549	19.7830	20.0000	22.6252	24.5755	27.4224	24.9373	22.3636	23.3429	16.4383
nercut	34.0872	36.2086	41.4666	40.8462	39.3208	39.8255	47.9949	42.0000	42.9394	31.5068
	20.7323	19.8863	24.4000	25.2158	27.2564	29.9418	31.9548	30.1818	26.8011	16.4383
icu	31.1254	31.9731	38.0000	37.7374	37.1760	37.7906	45.2380	42.3636	40.0576	36.9863
	21.7555	23.2438	22.9333	22.3661	22.7882	23.2558	25.4385	26.3636	23.6311	21.9178
attacut	29.1330	31.2500	36.8000	37.3056	36.6398	36.8217	43.1077	42.0000	40.3458	34.2465
	21.9709	21.7975	24.9333	28.4110	29.3118	30.6201	36.7167	34.0000	30.8357	20.5479
deepcut	35.8104	37.3966	41.0667	41.4507	40.1251	40.9883	48.4962	46.0000	46.0958	41.0958
	7.3236	11.2086	12.4000	13.5578	15.9964	16.9573	18.6716	14.9090	13.8328	10.9589

Table 18. Comparison of classification accuracies between using original sets of tokens and using sets of common consecutive tokens on application ratings dataset.

	Proposed model		Original model	
	Original set of tokens	Set of common consecutive tokens	original set of tokens	Set of common consecutive tokens
newmm	0.54	0.42	0.25	0.11
newmm-safe	0.60	0.49	0.26	0.11
longest	0.52	0.46	0.28	0.11
nercut	0.40	0.38	0.27	0.13
icu	0.50	0.39	0.34	0.11
attacut	0.51	0.44	0.26	0.12
deepcut	0.51	0.46	0.26	0.16

Table 19. Comparison of accuracies among seven tokenizers on application ratings dataset split by the length of common consecutive tokens (green row indicates classification accuracies on the original sets of tokens while another row indicates classification accuracies on the sets of common consecutive tokens).

	0-10	10-20	20-30	30-40	40-50	50-60	60-70
newmm	36.8672	50.4234	56.7596	59.0163	54.3478	64.7058	57.1428
	11.4250	9.6491	7.2961	6.8306	11.9565	11.7647	14.2857
newmm-safe	44.9050	54.0229	63.4120	64.2076	60.8695	64.7058	42.8571
	11.9210	9.2558	6.9742	8.4699	13.0434	5.8823	0.0000
longest	42.0698	52.8130	61.0515	57.6502	61.9565	58.8235	71.4285
	11.3034	10.7380	9.2274	10.9289	8.6956	0.0000	0.0000
nercut	36.2964	40.4113	42.0600	43.9890	41.3043	52.9411	42.8571
	13.8673	13.0973	9.4420	10.3825	9.7826	17.6470	14.2857
icu	34.3782	48.3363	56.5450	55.4644	53.2608	58.8235	57.1428
	11.5560	9.0744	7.2961	6.0109	6.5217	17.6470	0.0000
attacut	39.6275	50.0302	59.1201	57.9234	58.6956	64.7058	57.1428
	12.1362	11.4337	8.6909	10.9289	9.7826	17.6470	0.0000
deepcut	42.2101	53.1760	61.8025	61.4754	54.3478	70.5882	71.4285
	15.2053	18.3000	20.2789	24.5901	21.7391	29.4117	0.0000

Table 20. Comparison of accuracies among seven tokenizers on application ratings dataset split by the length ratio of common consecutive tokens (green row indicates classification accuracies on the original sets of tokens while another row indicates classification accuracies on the sets of common consecutive tokens).

	0-10	10-20	20-30	30-40	40-50	50-60	60-70	70-80	80-90	90-100
newmm	29.4800	35.7244	40.8932	46.0246	47.5080	52.6446	54.0803	57.3459	56.5048	54.7619
	9.1600	9.5486	9.9790	11.3580	12.8802	14.8760	14.1291	15.9557	11.4563	10.9523
newmm-safe	45.3600	43.0166	47.1039	49.8271	51.1974	56.1157	56.0292	59.8736	62.1359	60.4761
	10.4800	9.4774	9.4556	11.0123	13.3980	15.6198	16.0779	15.6398	12.4271	10.9523
longest	39.0800	41.1638	45.5338	48.8395	48.8025	54.2975	53.8367	56.8720	58.2524	60.9523
	8.6800	10.2137	10.5373	11.5061	12.0388	15.0413	14.8599	15.1658	12.8155	12.8571
nercut	37.4400	35.3681	36.0083	37.9259	39.8705	42.9752	37.8806	43.6018	42.5242	42.8571
	9.8799	12.0190	13.9218	13.9753	15.7281	17.7685	16.3215	17.8515	15.7281	10.9523
icu	27.0000	33.1591	39.9511	43.0123	45.3721	51.0743	52.1315	52.6066	54.7572	55.7142
	10.2800	8.7173	9.6999	11.7037	13.0744	14.8760	14.3727	15.4818	11.8446	9.5238
attacut	40.4000	37.5296	42.2889	45.2345	45.5016	51.6528	53.3495	55.2922	55.7281	52.8571
	11.0800	9.5249	10.3628	12.6913	14.4336	17.4380	16.3215	17.5355	13.2038	9.0476
deepcut	41.9200	40.1900	44.1381	48.8395	50.6148	54.6280	56.2728	57.9778	58.2524	54.7619
	13.0800	12.6603	15.1081	17.5802	20.0647	22.9752	25.0913	25.9083	25.6310	20.4761

Table 15 and table 18 show that common consecutive tokens have different meanings in terms of classification tasks. The accuracies of both proposed model and original model demonstrate higher performance when compared with the models using only common consecutive tokens.

According to Tables 16, 17, 19, and 20, after dividing the sets of tokens by their length or length ratio, all classification accuracies of using the entire set of tokens were higher than using only the surrounding tokens. It can be implied that using the common consecutive tokens might has some impact on the classification accuracies but using the whole set of tokens significantly influences to the overall accuracies.

6.2 Tokenization Results

After two datasets collected from restaurant ratings and application ratings were cleaned, the number of sentences were 21,211 and 20,258, respectively. These

datasets were split into three parts of each training, validating, and testing set. After that tokenization process was applied, seven tokenizers including newmm, newmm-safe, longest, nercut, icu, attacut and deepcut were based on different tokenization techniques. Some results of tokenization are shown in Figure 16.

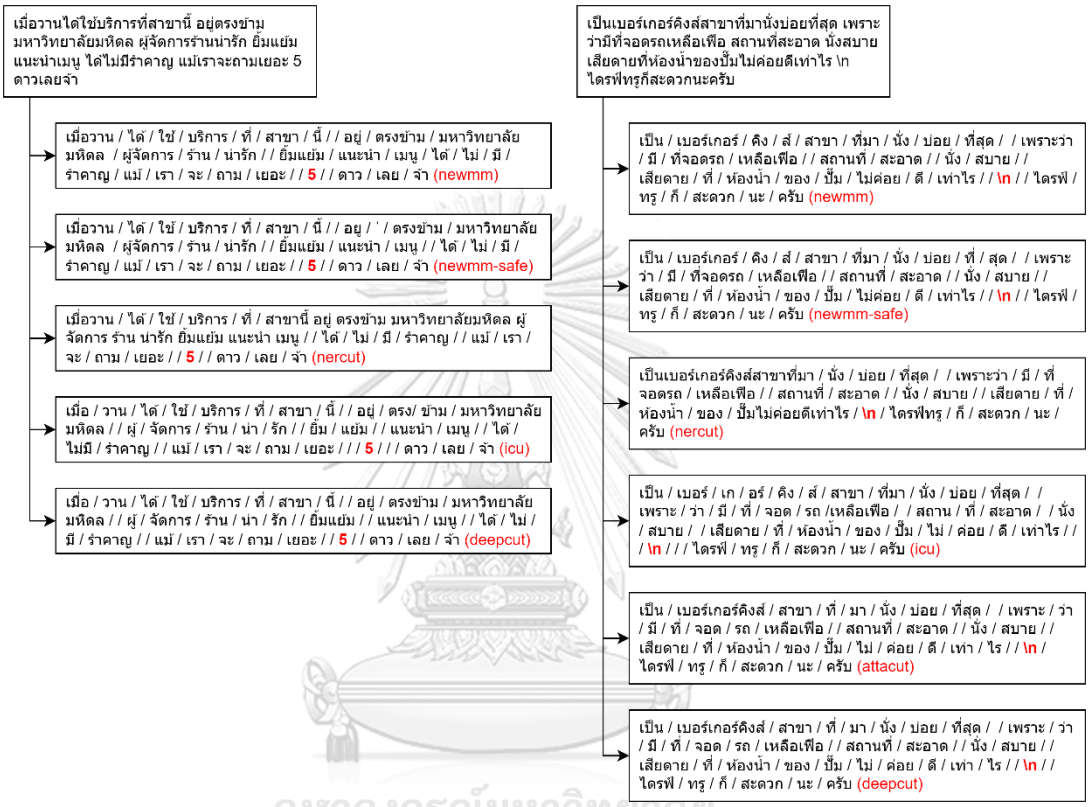


Figure 16. Samples of tokenized sentences obtained from two original sentences.

From Figure 16, left sentence can be tokenized into 5 different sets of words. Newmm and newmm-safe tokenizers yielded mostly same result but, newmm-safe had miscalculation on word “อยู่” which was tokenized into “อยู” and “-่”, nercut tokenizer groups some consecutive words that are next to each other as one token, icu and deepcut generate tokens from a small unit of word. In the right sentence, a proper noun, which is the name of restaurant in this sample, was differently tokenized into a set of words. These results are a good example affected by the tokenizers.

Table 21. Comparison of classification results of the right side of the sampled sentence

	Predicted Class				Actual Class
	Bi-LSTM		DistilBERT		
	proposed model	original model	proposed model	original model	
newmm	5-star	4-star	5-star	5-star	5-star
newmm-safe	5-star	5-star	4-star	4-star	
longest	5-star	5-star	5-star	5-star	
nercut	5-star	4-star	5-star	4-star	
icu	5-star	5-star	5-star	5-star	
attacut	5-star	3-star	5-star	3-star	
deepcut	5-star	3-star	5-star	4-star	

Noticeably, the volatility of the classification results using the proposed method is less than those of the original method and the classification results are also more correct.

CHAPTER VII.

SUMMARY

This thesis aimed to solve Thai word boundaries problem that cause results from different sentence tokenizer to be different. This study proposed a neural networks architecture for sentence encoder based on clustering technique. The tokenized dataset was generated from seven different tokenizer from pythainlp [1] and use custom shuffling in training phase to maintain triplet hard loss requirement. After clustering architecture is trained, t-SNE visualization shown good clustering of sentences. Chapter V demonstrated that the proposed architecture can be used as pre-trained classification networks. To build the classifier, a trained clustering architecture is frozen, then Random Fourier Feature [9], linear classifier, and hinge loss is appended to that architecture. The experiments calculated similarity and accuracy scores from with and without applying a trained parameter from clustering. The analysis of the scores has shown that, the average improvement of similarity and sentiment accuracy can be improved when the proposed model is transferred. Chapter VI analyzed the effect of surrounding words from consecutive tokens that is not affect by Thai word boundaries problem. Sentiment classifiers were built to test the classification that surrounding words affect classification results or not. Only consecutive tokens and its rating was fed to train classifier, the results shown that proposed model is affected by surrounding words, but it does not significantly impact sentiment result.

REFERENCES

[1] W. Phatthiyaphaibun, A. Suriyawongkul, P. Chormai, Charin, L. Lowphansirikul, and P. Siwatammarat, *PyThaiNLP/pythainlp: PyThaiNLP v3.0.5 Released!* Zenodo, 2022. doi: 10.5281/zenodo.6075269.

[2] "pythainlp.tokenize — PyThaiNLP 2.0.3 documentation." https://pythainlp.github.io/docs/2.0/api/tokenize.html (accessed May 19, 2022).

[3] M. Domingo, M. Garcıa-Martınez, A. Helle, F. Casacuberta, and M. Herranz, "How much does tokenization affect neural machine translation?," *arXiv preprint arXiv:1812.08621*, 2018.

[4] K. Park, J. Lee, S. Jang, and D. Jung, "An empirical study of tokenization strategies for various korean nlp tasks," *arXiv preprint arXiv:2010.02534*, 2020.

[5] G. Kim and S.-H. Lee, "Comparison of Korean Preprocessing Performance according to Tokenizer in NMT Transformer Model," *Journal of Advances in Information Technology Vol*, vol. 11, no. 4, 2020.

[6] G. Koch, R. Zemel, and R. Salakhutdinov, "Siamese neural networks for one-shot image recognition," in *ICML deep learning workshop*, 2015, vol. 2, p. 0.

[7] F. Schroff, D. Kalenichenko, and J. Philbin, "Facenet: A unified embedding for face recognition and clustering," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 815–823.

[8] A. Hermans, L. Beyer, and B. Leibe, "In defense of the triplet loss for person re-identification," *arXiv preprint arXiv:1703.07737*, 2017.

[9] A. Rahimi and B. Recht, "Random features for large-scale kernel machines," *Advances in neural information processing systems*, vol. 20, 2007.

[10] R. Kittinaradorn *et al.*, *DeepCut: A Thai word tokenization library using Deep Neural Network.* Zenodo, 2019. doi: 10.5281/zenodo.3457707.

[11] P. Chormai, P. Prasertsom, and A. Rutherford, "Attacut: A fast and accurate neural thai word segmenter," *arXiv preprint arXiv:1911.07056*, 2019.

[12] P. Gage, "A new algorithm for data compression," *C Users Journal*, vol. 12, no. 2, pp. 23–38, 1994.

[13] R. Sennrich, B. Haddow, and A. Birch, "Neural machine translation of rare words with subword

units," *arXiv preprint arXiv:1508.07909*, 2015.

[14] A. Vaswani *et al.*, "Attention is all you need," *Advances in neural information processing systems*, vol. 30, 2017.

[15] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[16] V. Sanh, L. Debut, J. Chaumond, and T. Wolf, "DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter," *arXiv preprint arXiv:1910.01108*, 2019.

VITA

NAME	Noppadol Kongsumran
DATE OF BIRTH	8 April 1998
PLACE OF BIRTH	Chiang Rai, Thailand
INSTITUTIONS ATTENDED	B.Sc. in Computer Science, Department of Mathematics and Computer Science, Chulalongkorn University, 2019.
HOME ADDRESS	1356/201 M.9 Samrong Nuea, Mueang Samut Prakan, Samut Prakan 10270