Using Label Noise Filtering and Ensemble Method for Sentiment Analysis on Thai Social Data

Chayanont Eamwiwat*, Pongpisit Thanasutives*, Chanatip Saetia, Tawunrat Chalothorn
Kasikorn Labs
Nonthaburi, Thailand
{chayanont.eam, pongpisit.tha}@gmail.com, {chanatip.sae,tawunrat.c}@kbtg.tech
* *equally contributed authors*

***Abstract*—Sentiment analysis is an essential task for social listening, especially in service and product analysis. Prior works on sentiment analysis, especially in Thai language, mostly focus on the improvement of model architecture without considering error propagation from word tokenizers or noisy text from social media. In this paper, three contributions are proposed for implementing social analysis model. First, text pre-processing is used to mitigate noise from input texts. Second, robustness towards word segmentation is enhanced by using an ensemble process with two tokenizers. Lastly, the training process inspired by Co-training method is proposed in order to filter label noise within the data. In the experiments, the model achieves 2.56% improvement on the average macro f-1 score when compared with the baseline models in social media data.**

***Keywords*—Sentiment analysis, Text classification, Text preprocessing, Word segmentation, Noise-cancelling algorithm**

I. Introduction

Recently, social listening has become a crucial method to gather information of customers from public channels such as Twitter and Facebook. Social listening, in its usual form, retrieves information and context from social channel. It then performs sentiment analysis to inspect each message and label its sentiment into "neutral", "positive", or "negative".

Prior studies on sentiment analysis proposed deep learning models, which outperforms traditional approaches such as Support Vector Machine and Naive Bayes [1], [2]. However, most of the studies do not focus on the impact of error propagation from word segmentation and noisy text. Moreover, the labels are inconsistent due to the subjectivity of the task, which focuses on detecting positive and negative instances. As a result, there are some instances that are annotated as neutral because they do not relate to the target domain and expected to be ignored.

To handle the mentioned problems, three contributions to improve sentiment analysis are proposed.

First, text processing is designed to reduce noises in Thai social text. Unnecessary parts of the input message, such as URLs and usernames, are removed. Duplicated spaces are replaced with one space. Moreover, a space is inserted between Thai and English characters to help word tokenizers identify word boundaries between them.

The second contribution is the ensemble model, which combines the output from different tokenizers. The ensemble model adds more perspectives on input tokens and therefore gains the benefit from both input tokens while tolerating false word segments caused by each tokenizer.

Finally, a label-noise filtering algorithm inspired by Cotraining [3] is adopted to explore label errors and imbalanced data set. This process is performed to avoid confusion caused by subjectivity from the labeling process. In the observed dataset, there are labeled instances which are annotated as neutral because they are not related to the target domain (although they contain positive or negative keywords). To clean up this ambiguous labeling process, training instances are re-labelled via probabilistic prediction from a trained model. Then, the re-labelled instances are used to train the new model. This process is our third contribution.

The remainder of this paper is structured as follows; the theoretical background of the related works is reviewed in Section II. Section III describes the proposed algorithms and architectures. The datasets, baseline models and evaluation metrics are explained in Section IV. After that, the results are analysed and discussed in Section V. Finally, the conclusion is in Section VI.

II. Related works

In this section, related works are presented in three subsections. They consist of sentiment analysis, the models for Thai word segmentation and label noise, and semi-supervised learning methods.

A. Sentiment analysis

Sentiment analysis is a method to retrieve emotional, opinion-oriented information from input texts [3]. Thai text classification and sentiment analysis using deep learning and statistical learning techniques have been studied in the recent years [1], [2]. For example, Kim Y. [4] proposed multiple branches of CNN with different kernel sizes to find grams (multiple sizes) for capturing the sentiment of messages. Meanwhile, Vateekul P. et al. [1] showed that deep neural network outperforms most of the traditional approaches such as Naïve Bayes and Logistic regression in social media data. Besides that, Yang Z. et al. [5] proposed the model called Hierarchical

Attention Network (HAN), which outperforms LSTM [6] in document classification. HAN utilizes a hierarchical structure

of Gated Recurrent Unit (GRU) with self-attention mechanism to understand the sentiment of messages by considering words in order and focusing on the important context related to the message.

In this paper, CNN and HAN are considered as the baseline models. CNN is widely considered as the state-of-the-art method for Thai sentiment analysis model [1]. HAN is also a competitive model for English document classification [5].

B. Word segmentation

Word segmentation is the process of dividing the written text into meaningful words. Most of the prior word segmentation models, such as Deepcut [7] and Sertis [8], are based on deep learning modules, including CNN and GRU. These models achieved 0.992 F1-score on NECTEC's BEST corpus [9]. However, employing these word segmentation models causes a significant problem. Since the models were trained on the NECTEC's BEST corpus containing formal Thai language, they are not suitable for texts from social media. On the contrary, KBTGTK [10] deep learning tokenizer was specially designed and trained with the UGWC dataset from social media. Due to the similarity between UGWC and our dataset, KBTGTK is chosen to be used in the experiment in this paper.

Besides deep learning word segmentation models, another approach of word segmentation called NEWMM [11] tokenizes the text to achieve the fewest number of words found in the dictionary.

NEWMM is a dictionary-based maximal matching algorithm, which is fast and only produces words appearing in the specified dictionary. Hence, NEWMM accurately segments known words and ignores noisy texts around the known words. While NEWMM does not face the same scalability problem other deep learning-based tokenizers, it cannot handle unknown words, such as name entities, that are not included in the specified dictionary.

Since KBTGTK and NEWMM have different advantages and drawbacks, both algorithms are explored in the experiments in Section V-B.

C. Label noise and semi-supervised learning

Large human-annotated dataset suffers from labelling errors due to various problems such as subjectivity, human errors, and notably the presence of label noise [12]. Label noise becomes a common problem when performing classification and causes many potential negative consequences. [12] shows that there are various approaches that could be used to handle label noise such as applying algorithms to avoid overfitting, improving quality of data using filtering approaches, and incorporating label noise as embedding in the model. The filtering approach is simple but efficient since it removes noisy data explicitly whereas the other approaches do not [12]. Hence, this paper uses a filtering approach to determine and relabel noisy instances to improve the quality of data.

Specifically, this paper applies a noise-filtering algorithm similar to Co-training [13], which is a semi-supervised learning method to utilize unlabelled data. The first step of Co-training is training two models with labelled data. After that, unlabelled data, which is predicted by the first-step models with high confidence, is labelled and used for training the models again. The method is performed repeatedly until there is no more instances of unlabelled data with prediction confidence level higher than the specified threshold. In this work, the method is used to correct noisy labelled data rather than using it to label unlabelled data.

III. Methodology

For this section, the methods that are used in the experiment are described, and the workflow of the process is shown in Figure 1.

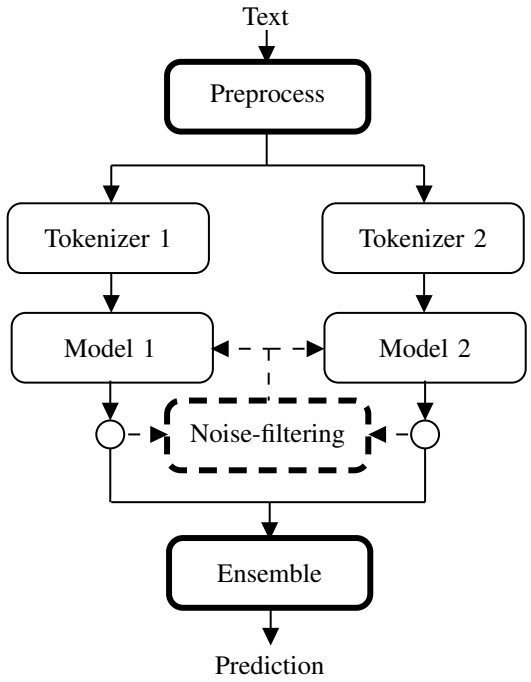

Figure 1. Work procedure of purposed methods. Bold line represents the proposed methods of this paper. Dash line represents the method applied only over training phase.

First, an input text goes through the text preprocessing as described in Section III-A. Second, for the ensemble process, the cleaned text is used for training and inference with two tokenizers and models. After that, in the training process, the output from both models are utilized to filter label noise as explained in Section III-B. In the inference process, two output from both models are combined using the ensemble method mentioned in Section III-C.

A. Text preprocessing

Text preprocessing is performed before word segmentation in order to clean inputs before tokenization and remove unnecessary parts of the text. Five steps of text preprocessing are performed as follows.

1) Convert all uppercase characters to be lowercase.
2) Remove characters that are not in Thai and English language.

3) Remove URL patterns and usernames, which are usually meaningless for sentiment analysis.
4) Add a space (“ ”) between Thai and English character, which is usually a word boundary.
5) Replace a sequence of duplicated spaces with one space.

B. Ensemble with Different Word Segmentation Algorithms

In this section, trained models are applied with ensemble methods, as shown below. The ensemble model is constructed from different tokenizers and models, as shown in Figure 1. The weighted average of the two probability is calculated from the output probability of each ensemble model.

$$P = \alpha * p_1 + (1 - \alpha) * p_2 \tag{1}$$

In Eq. 1, P denotes the output probability of the ensemble model. Meanwhile, p_1 and p_2 are the prediction probabilities given from different combinations of tokenizers and models. α is determined for weighting between two probabilities where $0 \leq \alpha \leq 1$.

C. Noise-filtering algorithm for imbalanced dataset

This section describes the noise-filtering algorithm used in the paper. The applied algorithm is inspired by Co-training [13]. However, instead of using the algorithm to label the unlabelled data, the algorithm is used to re-label the labelled data.

The algorithm is illustrated in Algorithm 1. m_1 and m_2 are the models trained from different aspects such as architectures or tokenizers. τ is the optimal threshold for relabeling the neutral instance. The threshold is tuned to obtain the best macro-F1 score whereas $\tau \in \{0.05, 0.10, 0.15, ..., 1.00\}$. p_{class} is the probability for identical class given from averaging of probabilities of m_1 and m_2.

Algorithm 1: Noise-filtering algorithm for sentiment analysis dataset

Input: m_1: 1^{st} model to be used as filter, m_2: 2^{nd} model to be used as filter, U: train dataset
Output: m: model after being applied noise-filtering algorithm

```
/* train m1 and m2 on U                                   */
m1 ⟵ train_model(U);
m2 ⟵ train_model(U);
/* Loop to relabel an instance                            */
for x ∈ U do
    if x is "neutral" then
        /* average probabilities from models              */
        p1 ⟵ m1.predict(x);
        p2 ⟵ m2.predict(x);
        p_neutral, p_positive, p_negative = (p1 + p2)/2;
        /* relabel an instance with probability
           lower than the threshold                       */
        if p_neutral ≤ τ then
            x ⟵ max(P_positive, P_negative)
        end
    end
end
/* train model with the relabeled dataset                 */
m ⟵ train_model(U);
```

IV. Experimental Setup

The processes of setting and conducting experiment are explained in this section.

A. Model

In the experiments, Hierarchical attention neural networks (HAN) [5], and the convolutional neural networks (CNN) [4] are used with their weights initialized randomly. Early stopping is also used to avoid overfitting problem based on macro-F1 of validation dataset. Moreover, Adam optimizer [14] with 0.001 initial learning rate is used to optimize the categorical cross-entropy loss. In our experiments, AllenNLP framework is used for implementing the models [15].

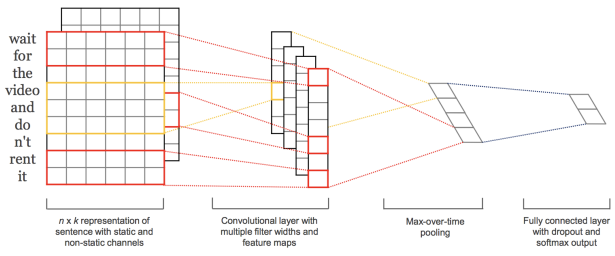

Figure 2. Convolutional neural networks for text classification [4]

Moreover, for both CNN and HAN models, the input is a sequence of 200 embedded words. Each word is embedded into a vector with 256 dimensions without pre-trained word embedding before being dispatched into the models. Each model maps input text to sentiment class probabilities.

The architecture of CNN, which is proposed for text classification [4], is shown in Figure 2. In this work, CNN has three branches of convolutional layers with max pooling. Before applying max-pooling operation, ReLU [16] have been applied to the feature maps output of every convolutional layer. The convolutional layer in each branch has 50 filters with different kernel sizes; 3x256, 4x256 and 5x256. The filters outputs from different branches are concatenated and flattened into a single vector. Then the vector is passed through a softmax layer to get sentiment class probabilities.

Meanwhile, Figure 3 illustrates the architecture of HAN for text classification. In the model, the sequence of word vectors is dispatched into bidirectional GRU with 128 nodes. Then, the attention mechanism (at word level) is applied to create a sequence of 200 vectors with 256 dimensions. After that, the vectors are forwarded into bidirectional GRU and attention mechanism (at the sentence level) with the same procedure and finally projected into a vector used for classification via a softmax layer to retrieve the class probabilities.

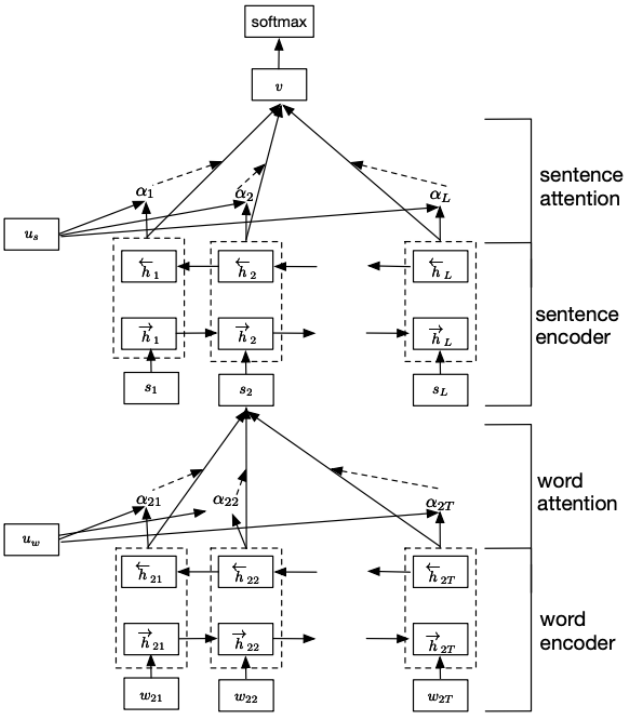

Figure 3. The architecture of Hierarchical Attention Networks (HAN) [5]

B. Dataset

Table I
THE LABEL DISTRIBUTION OF A SENTIMENT DATASET

Class	*Number of Sample*	*Percentage*
Neutral	267,397	82.73
Positive	14,769	4.56
Negative	41,030	12.69

The dataset is collected from social media platforms such as Facebook and Twitter. The 323,196 samples of them are conducted in this experiment and manually labelled by human annotators. The labels are "neutral", "positive" and "negative" of which distributions are shown in Table I.

This dataset contains neutral instances for more than 80%, which causes imbalanced data issue. Most instances are annotated as neutral due to two reasons. First, the data set contains a large portion of advertisement, blogs, and news (which are normally written with unbiased contents). Second, some instances that are not related to the target domain are considered neutral regardless of any positive and negative meaning.

In the experiments, the dataset is divided into train set, validation set, and test set with proportion 80%, 10%, and 10% respectively. The validation set are used for hyperparameter tuning and performing early stopping.

C. Evaluation metric

$$precision_{\text{class}} = \frac{\text{True positive}}{\text{True positive + False positive}}$$

$$recall_{\text{class}} = \frac{\text{True positive}}{\text{True positive + False negative}}$$

$$F1_{\text{class}} = \frac{2 \cdot precision_{\text{class}} \cdot recall_{\text{class}}}{precision_{\text{class}} + recall_{\text{class}}}$$

$$F1_{\text{macro}} = \frac{F1_{\text{neutral}} + F1_{\text{positive}} + F1_{\text{negative}}}{3} \quad (2)$$

For all experiments, the models are evaluated based on macro-F1 as each class is treated equally, as shown in Eq. 2. The reported score is the average macro-F1 calculated from three models, which are initialized with different random seeds.

V. RESULTS AND DISCUSSIONS

This section presents the experimental results and discusses the effect of the three proposed ideas on the results.

A. Effect of our text preprocessing on metrics

Text preprocessing is applied toward different combinations of models and tokenizers. The results are shown in Table II which text preprocessing enhances the model macro-F1 in all combinations of models and tokenizers. The models with text preprocessing improve slightly on average macro-F1 score (0.60%). Especially, HAN with KBTGTK gains 1.37% macro-F1 improvement from text preprocessing.

B. Ensemble with different tokenizers

In this experiment, the models that are used for assembling are selected with three different aspects. First, the models, which are assembled, use the same architecture and tokenizer, but they are trained with different random seeds. Second, the ensemble model is constructed from the models with different tokenizers but with the same architecture. Finally, the models with different architectures and tokenizers are assembled.

From Table II, the ensemble model with identical word segmentation algorithms (NEWMM + NEWMM and KBTGTK + KBTGTK) slightly improves the macro-F1 in both models. When NEWMM + NEWMM is applied, the macro-F1 scores are improved from 71.95% to 72.91% (+ 0.96%) in CNN and 70.85% to 71.25% (+ 0.40%) in HAN. Meanwhile, the macro-F1 scores of the model with KBTGTK + KBTGTk is raised from 70.63% to 71.45% (+ 0.82%) in CNN and 71.25% to 71.75% (+ 0.50%) in HAN.

Moreover, two different word segmentation algorithms are also assembled. The ensemble model with NEWMM + KBTGTK achieves better accuracy in term of the averaged macro-F1, increased from 71.75% to 72.89% (+1.14%) in HAN and 72.91% to 73.06% (+0.15%) in CNN. This result shows that the model exploits the benefit of both tokenizers (NEWMM and KBTGTK). The input, which is tokenized by NEWMM, is ensured that the given words are found in the dictionary.

Table II
THE RESULTS OF TEXT PROCESSING AND ENSEMBLE USED IN EXPERIMENTS

Model 1	**Tokenizer 1**	**Model 1**	**Tokenizer 2**	**F1 macro**	
				- Preprocessing	***+ Preprocessing***
HAN	NEWMM	-	-	70.71	70.85
	KBTGTK	-	-	69.88	71.25
	NEWMM	HAN	NEWMM	71.56	71.25
	KBTGTK		KBTGTK	70.30	71.75
	NEWMM		KBTGTK	71.95	72.89
CNN	NEWMM	-	-	71.55	71.95
	KBTGTK	-	-	70.01	70.63
	NEWMM	CNN	NEWMM	72.47	72.91
	KBTGTK		KBTGTK	70.97	71.45
	NEWMM		KBTGTK	72.78	73.06
HAN	KBTGTK	CNN	NEWMM	**72.90**	**73.67**

Table III
THE RESULTS OF APPLYING NOISE-FILTERING ALGORITHM FOR IMBALANCED DATASET USED IN EXPERIMENTS

Model	**F1 macro**		
	- Pre - NF	***+ Pre - NF***	***+ Pre + NF***
HAN + KBTGTK	70.71	70.85	71.51
CNN + NEWMM	71.55	71.95	72.44
Ensemble Model	72.90	73.67	73.69

Note: **Pre** denotes preprocessing process. Meanwhile, **NF** denotes noise filtering process.

Meanwhile, the Out-of-Vocabulary (OOV) words are handled by KBTGTK. Thus, utilizing NEWMM and KBTGTK could enhance the robustness and diminish errors caused by each tokenizer.

In addition, CNN and HAN are assembled together as the ensemble model to gain a higher macro-F1 score. For word segmentation, the tokenizer that is combined with each model and achieves the higher macro-F1 score is selected. NEWMM is selected for CNN. Meanwhile, HAN is used with KBTGTK. The ensemble model achieves the highest averaged macro-F1 at 73.67%.

C. Noise-filtering algorithm for imbalanced dataset

In this experiment, the best ensemble model given the best results (HAN + KBTGTK and CNN + NEWMM) is used as the model for reducing noise in the proposed noise filtering algorithm. The improvements of various models are noticeable after applying our noise-filtering algorithm. In Table III, the results show that the models with noise filtering improve slightly on macro-F1 from 72.67% to 72.69% in the ensemble model. Also, the improvement of the individual models (HAN + KBTG and CNN + NEWMM) are measured, after the noise-filtering algorithm is applied. The macro-F1 of HAN with KBTGTK tokenizer improves from 70.85% to 71.51%. Meanwhile, the macro-F1 of CNN with NEWMM tokenizer increases from 71.95% to 72.44%.

Furthermore, the label distribution after relabelling by the noise filtering is examined and shown in Table IV. The label distributions of both models with noise-filtering are changed in a similar way. The number of neutral instances decreases while the numbers of other label instances increase. These results demonstrate subjectivity in sentiment analysis labelling tasks.

In the data set, some instances not related to the target domain are annotated as neutral regardless of positive and negative keywords. For instance, "สัด กุดูตั้ง 4 นาทีกว่า คิดว่าโฆษณาหนังใหม่เหอะ" is annotated as neutral because the instance is about a released video and is not related to the target domain (i.e., finance). However, the word "สัด" is obviously a negative keyword, Therefore, the trained model is easily confused with these conflicting neutral instances. After integrating label noise-filtering, these neutral instances are relabelled as positive and negative. This leads to a decrease in the number of neutral instances as shown in Table IV.

By performing label-noise filtering, the model can focus on sentiment keywords and ignore the bias from the annotators.

Table IV
DISTRIBUTION OF TRAINING DATASET (BEFORE AND AFTER APPLYING NOISE-FILTERING)

Model	**Tokenizer**	τ	***Class***	**Percentage (before $\rightarrow$ after)**
HAN	KBTGTK	0.60	neutral	82.73 $\rightarrow$ 78.92
			positive	4.56 $\rightarrow$ 5.89
			negative	12.70 $\rightarrow$ 15.17
CNN	NEWMM	0.50	neutral	82.73 $\rightarrow$ 80.38
			positive	4.56 $\rightarrow$ 5.38
			negative	12.70 $\rightarrow$ 14.23

Moreover, the imbalanced data issue is also cured by this process. As a result, the model improves marginally.

D. Overall improvement of the model

In Table III, the model with the combination of all proposed contributions reaches 73.69% macro-F1, which is the highest performance among all models in the experiment. To evaluate the model (HAN + KBTGTK and CNN + NEWMM) and models with none of the three proposed contributions are considered as baseline models. The macro-F1 of the model improves by 2.98% and 2.14% compared to HAN + KBTGTK and CNN + NEWMM baseline models, respectively.

VI. Conclusion

In this paper, three contributions were conducted to improve sentiment analysis. First, five steps of text preprocessing were applied to reduce input noise. After getting rid of the noise, the macro-F1 score of each model improved 0.60% on average. Second, the ensemble model with different tokenizers was also proposed to tolerate the error produced by a single tokenizer. The result showed that assembling two tokenizers (NEWMM and KBTGTK) could improve the macro-F1 of HAN and CNN by 1.14% and 0.15%, respectively. Moreover, by combining two models (HAN and CNN), the macro-F1 increased from 73.06% to 73.67%. Finally, to address the subjectivity of sentiment analysis task, the labelled noise was handled by a noise-filtering algorithm. The model gains the improvement of macro-F1 at 0.58%. The model that contains all three contributions yielded 2.98% and 2.14% improvement on macro-F1 for HAN and CNN, respectively.

However, the model described in this paper could be further integrated with intention classifiers. Therefore, intention and sentiment could be an automated system for social listening that can evaluate the branding image of products and companies.

Acknowledgment

We would like to thank you the linguist team: Dr. Supawat Taerunruang and Dr. Nutcha Tirasaroj for their invaluable data analysis. Also, this work was supported by Kasikorn Business-Technology Group (KBTG).

References

[1] P. Vateekul and T. Koomsubha, "A study of sentiment analysis using deep learning techniques on thai twitter data," in *2016 13th International Joint Conference on Computer Science and Software Engineering (JCSSE)*. IEEE, 2016, pp. 1–6.

[2] Q. T. Ain, M. Ali, A. Riaz, A. Noureen, M. Kamran, B. Hayat, and A. Rehman, "Sentiment analysis using deep learning techniques: a review," *Int J Adv Comput Sci Appl*, vol. 8, no. 6, p. 424, 2017.

[3] B. Pang, L. Lee *et al.*, "Opinion mining and sentiment analysis," *Foundations and Trends® in Information Retrieval*, vol. 2, no. 1–2, pp. 1–135, 2008.

[4] Y. Kim, "Convolutional neural networks for sentence classification," *arXiv preprint arXiv:1408.5882*, 2014.

[5] Z. Yang, D. Yang, C. Dyer, X. He, A. Smola, and E. Hovy, "Hierarchical attention networks for document classification," in *Proceedings of the 2016 conference of the North American chapter of the association for computational linguistics: human language technologies*, 2016, pp. 1480–1489.

[6] D. Tang, B. Qin, and T. Liu, "Document modeling with gated recurrent neural network for sentiment classification," in *Proceedings of the 2015 conference on empirical methods in natural language processing*, 2015, pp. 1422–1432.

[7] R. Kittinaradorn, "A thai word tokenization library using deep neural network." [Online]. Available: https://github.com/rkcosmos/deepcut

[8] J. Jousimo, "Thai word segmentation with bi-directional rnn." [Online]. Available: https://sertiscorp.com/thai-word-segmentation-with-bi-directional_rnn

[9] K. Kosawat, M. Boriboon, P. Chootrakool, A. Chotimongkol, S. Klaithin, S. Kongyoung, K. Kriengket, S. Phaholphinyo, S. Purodakananda, T. Thanakulwarapas *et al.*, "Best 2009: Thai word segmentation software contest," in *2009 Eighth International Symposium on Natural Language Processing*. IEEE, 2009, pp. 83–88.

[10] A. Lertpiya, T. Chaiwachirasak, N. Maharattanamalai, T. Lapjaturapit, T. Chalothorn, N. Tirasaroj, and E. Chuangsuwanich, "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)*. IEEE, 2018, pp. 1–8.

[11] K. Chaovavanich, "Dictionary-based thai word segmentation using maximal matching algorithm and thai character cluster." [Online]. Available: https://github.com/PyThaiNLP/pythainlp

[12] B. Frénay and M. Verleysen, "Classification in the presence of label noise: a survey," *IEEE transactions on neural networks and learning systems*, vol. 25, no. 5, pp. 845–869, 2013.

[13] A. Blum and T. Mitchell, "Combining labeled and unlabeled data with co-training," in *Proceedings of the eleventh annual conference on Computational learning theory*. Citeseer, 1998, pp. 92–100.

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

[15] M. Gardner, J. Grus, M. Neumann, O. Tafjord, P. Dasigi, N. F. Liu, M. Peters, M. Schmitz, and L. S. Zettlemoyer, "Allennlp: A deep semantic natural language processing platform," 2017.

[16] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines," in *Proceedings of the 27th international conference on machine learning (ICML-10)*, 2010, pp. 807–814.