Data Science and Engineering (DSE) Record, Volume 5, issue 1.

Thai Spelling Correction Investigation Framework Based on OCR Word Extraction

Jukkrit Mengkaw[1] and Pree Thiengburanathum[2]

[1]Master's Degree Program in Data Science, Chiangmai University, Chiang Mai, Thailand
[2]College of Arts, Media and Technology, Chiangmai University, Chiang Mai, Thailand
jukkrit_me@cmu.ac.th

Abstract. Spelling correction (SC) is used to detect and correct misspelled words. SC is considered a fundamental task in various Natural Language Processing (NLP) applications, such as machine translation, chatbots, Optical Character Recognition (OCR) systems, etc. Currently, there exists a limited number of research works pertaining to spelling correction in low-resource languages, with a specific focus on evaluating the efficacy of Thai OCR in processing PDF documents. The issues pertaining to spelling correction in the Thai language involve not only the availability of benchmarking data, but also extend to text extraction from PDFs and the performance of models. In this study, we proposed a two-step spelling correction framework that includes detection and correction steps from image dataset. Experiment results revealed that in the error detection, Bi-LSTM revealed the highest performance and achieved an F1-score of 93.20%. In the error correction, Bi-LSTM with attention mechanism achieved F1-score of 86.31% and WangchanBERTa achieved F1-score of 81.36%. However, WangchanBERTa has a faster inference time than the Attention mechanism (40 times) and can reduce WER from 11.99% to 4.51%. The experiment results reveal that our proposed method effectively detects and corrects the Thai language.

Keywords: Spelling correction, Tesseract OCR, Deep learning, Transfer learning.

1 Introduction

Spelling correction is an essential part of written communication since misspellings can lead to misconceptions or even miscommunication. The process of spelling correction involves identifying and correcting misspelled words in a text. Spelling correction aims to improve the accuracy and readability of the text by fixing spelling, grammar, and punctuation mistakes. There are various NLP applications that apply spelling correction tasks, such as sentiment analysis [1], machine translation [2], and OCR [3]. The

effectiveness of spelling correction algorithms can be evaluated using various factors, including accuracy and speed [4]. Accuracy is the ability of the algorithm to correctly detect and correct misspelled words, while speed is the time that it takes to complete the correction.

The task of spelling correction is challenging for low-resource languages such as Thai. There are a few studies on Thai spelling correction, especially spelling correction from OCR results. Due to the ambiguity and lack of data, as well as the complexity of the language. The structure of the Thai language is quite complex due to the characters of the Thai language consisting of 44 Thai consonants, 16 Thai vowels, 10 Thai numbers, 4 Thai tones, and 10 Thai special characters, as shown in Table 1. The arrangement of Thai characters in Thai writing has four levels, including tone level, upper vowel level, consonant level, and lower vowel level. Furthermore, there are no spaces between words in Thai, making it challenging to tell where one-word ends, and another begins. This can lead to incorrect spelling when attempting to separate words and correct misspellings, as shown in Fig. 1. [5].

Table 1: *Thai characters.*

Type	List of characters
Consonants	กขฃคฅฆงจฉชซฌญฎฏฐฑฒณดตถทธนบปผฝพฟภมยรลวศษสหฬอฮ
Vowels	◌ะ ◌า ◌ิ ◌ี ◌ึ ◌ื ◌ุ ◌ู เ แ โ ◌ำ ไ ใ ฤ ๅ
Tones	◌่ ◌้ ◌๊ ◌๋
Numerals	๐๑๒๓๔๕๖๗๘๙
Special symbols	◌็ ◌์ ๆ ฯ

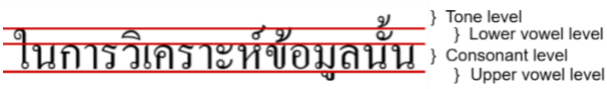

Fig.1: *Example of a Thai sentence.*

Spelling correction algorithms can be split into two main types: dictionary-based [6], [7] and deep learning-based [8]. Dictionary-based spelling correction algorithms rely on a pre-existing dictionary of correctly spelled words. When a word is misspelled, the algorithm compares it to words in the dictionary to find the closest match, which is then suggested as the correction. Hunspell is popular spell-check (dictionary-based) and support multi-languages Deep learning-based spelling correction algorithms, on the other hand, use statistical models to determine the likelihood of a word being spelled correctly. Currently, Bidirectional Encoder Representations from Transformers (BERT) [9] has been used to develop the spelling correction task [10], [11]. In Thai spelling correction, there are a few

studies, such as using Genetic algorithm (GA) to correct the sentence that generated from OCR [12] and using the sequence-to-sequence (Seq2seq) neural networks with context-aware in error correction [13].

In this paper, we investigate suitable detection and correction models for the two-stage spelling correction framework consisting of error detection and error correction. In error detection, we provided a comparative benchmarking of the performance of two deep learning models (e.g., LSTM and Bi-LSTM), and one machine learning model (CRF). In error correction, we compared the performance of three deep learning models (e.g., LSTM, Bi-LSTM, and Bi-LSTM with Attention mechanism) and one transfer learning model (WangchanBERTa) for Thai spelling correction on a Thai academic writing dataset.

2 Literature Review

2.1 Optical Characteristic Recognition

Nowadays, PDF files are widely used as a format for storing and sharing information from academic papers and reports [14]. The convenience and portability of PDF files make them a popular choice for sharing information. However, the format of PDFs is not as easily editable as text in other formats. To perform the text analysis in PDF or text as image files, the first requirement is extracting text from those files.

OCR is a well-known technology that assists the user in terms of recognizing and retrieving full alphanumeric recognition of printed, as well as handwritten character units [15], [16]. OCR technology has been applied to many applications, such as license plate recognition [17], [18], invoice processing [19], [20], and ID card identification [21], [22]. One of the most important aspects of OCR is the accuracy of text extraction. OCR accuracy can be increased using a variety of techniques, including image processing [23], [24] and Natural Language Processing (NLP) [3], [25]. OCR was utilized for a wide range of applications and languages; hence, several techniques were employed to enhance its performance. The differences in the structure of each language lead to different OCR performances, especially in Thai, which has a rather complicated language structure.

OCR is the process of transforming the text contained in an image into text. In other words, OCR creates a message from scanned documents in either type of handwritten form [16]. There are currently several OCR tools available, such as Tesseract. The Tesseract OCR engine [26] is a free and open-source technology developed by HP between 1985 and 1995. Tesseract-OCR can be used for text extraction from image files and scanned documents efficiently in English, and there are several studies in other languages. Nevertheless, there are few studies in the Thai language with regard to Tesseract OCR, and the performance of the Thai language extraction is still required to be improved.

The performance of OCR was improved with many methods, such as image processing. Li et al. proposed the adjustment of the Tesseract engine and preprocessing of the input image to increase the accuracy of offline handwritten Chinese characters [15]. They adjusted the Tesseract engine, including page layout analysis, blob finding, text lines and

word finding, two-pass recognition, and recognition adjustment by generating multiple candidate recognition results. Furthermore, image preprocessing techniques were used in their preprocessing strategy, such as binarization, outline dilation and erosion, and image enlargement and reduction. Their improvement strategy can increase the average accuracy rate from 64% to more than 88%.

Lu et al. proposed a method that can get rid of the shadows that appear in the text images for the Tesseract OCR engine [24]. They used the local adaptive threshold algorithm to transform a grayscale image into a binary image. The noise between the texts and characters was then removed using a projection technique and a median filter. After that, the image was fed to the Tesseract engine. Their pre-processing method can increase the accuracy rate of extraction by using Tesseract to 88.8 - 97.8%.

In addition, NLP tasks have been implemented to improve the efficiency of word extraction. ONI and ASAHIAH et al. proposed a novel framework to recognize Yorùbá text [3]. They generated the Yorùbá printed text image dataset that contains 4,800 images and their corpus has more than 7,000 sentences. They developed an OCR model to recognize Yorùbá text by using LSTM and RNN methods. After that, they used the language model based on Standard Yorùbá as a spelling corrector to increase the performance of their model. Their results reveal that the Character Error Rate (CER) was 1.182%.

2.2 Spelling Correction

Spelling correction is the correction of misspelled words in the text. The first step is identifying the misspelled words and labeling each word in the text, whether it is spelled correctly or incorrectly. After that, correct the misspelled words. It is the same with the computer. In spelling correction, there are two main processes consist of error detection and error correction.

Error detection is the process of extracting the words from the input text. There are many approaches that can be used in error detection such as dictionary-based [27]. They used an open-source dictionary, namely AraComLex, that contains nine million Arabic words. The words will be checked against the AraComLex to see if they are included. Because the dictionary does not contain special characters such as diacritical marks, text preprocessing is required in this step.

Error correction is the process after error detection. The words that were labeled as erroneous words will be corrected in this process. Traditional error correction techniques, such as n-gram and noisy channel models [28]. Recently, n-gram technique was used to understand the context of multiple languages [29]. They generated the candidates and ranked them using Symmetric Delete Algorithm (SDA). Their proposed method was tested with 24 languages and the Thai language showed the lowest accuracy.

Nowadays, sequence-to-sequence (Seq2Seq) neural networks (i.e., Bi-GRU [30], Bi-LSTM) are utilized in spelling correction for end-to-end systems [13]. They proposed a model with contextual Attention to error correction. The model was trained using data from the Thai UGWC and outperformed the off-the-shelf models, such as Hunspell and

PyThaiNLP. Furthermore, Zhang et al. proposed Chinese spelling error correction (CSC) using BERT [11]. They proposed a novel architecture to increase the performance of CSC by using soft-masking technique. The architecture consists of an error detection network and an error correction network based on BERT. Their method outperformed the method using only BERT.

3 Methodology

In this section, we summarize the research method and framework of this study. This section is divided into three sub-sections including dataset, error detection, and error correction.

3.1 Dataset

In this study, we emphasize the correctable errors in our dataset, which contains 133,757 different error sentences derived from the actual OCR results. The corpus has 11,591 unique syllables, the total number of erroneous tokens is 224,361 tokens. The maximum syllable length is 25, while the minimum syllable length is 4, and the average syllable is 14. We split the dataset into 80,254 for the training set, 26,752 for the validation set, and 26,751 for the test set. Due to privacy considerations, the data is a private dataset. The corpus statistics are shown in Table 2.

Table 2: *Statistical distribution of the corpus.*

Descriptive Statistics	Count
Count	133,757
Mean	14.58
Standard Deviation	3.64
Minimum	4
25%	12
50%	15
75%	18
Maximum	25

The data in this research was collected from the title and abstract of the public theses in the Thai language that is available on the Thai Digital Collection (TDC) of ThaiLIS. There are many academic documents in the Thai language from various universities and institutes in Thailand. The keywords that were used are “study” (ศึกษา) and “analyze” (วิเคราะห์) to find the theses that have relevant titles and were created within 2006-2021. There are 10,262

academic writings that were collected. After data collection, we removed the duplicated writings and performed data cleaning by using NLP tasks such as removing non-Thai characters, removing special characters, sentence tokenization, and word tokenization.

We generated the text image dataset from our corpus by using three different font types including Cordia New, Angsana UPC, and THSarabunPSK, and different font sizes, including 12, 18, and 24 points. The text images were generated with two image types such as JPG and PNG. For example, see Fig. 2.

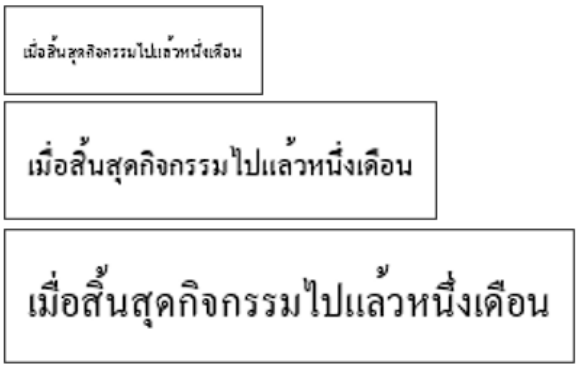

Fig.2: *Sample of text image regarding to the mention fonts and points.*

We performed Tesseract OCR to extract the text and collect the error sentences from extracting text images. There are frequent errors in text extraction such as newlines and some characters cannot be extracted. These errors significantly impact the error detection process, and the number of erroneous characters is notably high. The cause of these errors can be attributed to various factors, not only the complexity of Thai structure but also the performance of Thai OCR, image quality, and the diversity of Thai font styles. Therefore, we performed the postprocessing method for the results of text extraction to get rid of these problems. The sample of results after performing the post-process method was shown in Fig. 3.

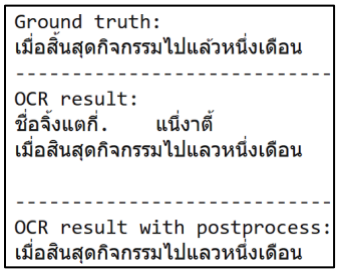

Fig.3: *Different result between original result and after post-processing.*

WER is one of the evaluation metrics that were used to evaluate the performance of OCR. We used WER in data selection to eliminate results with too many errors. WER is the number of errors divided by the total words. The formula is shown in Eq. 1.

$$\boldsymbol{WER} = \frac{\boldsymbol{i+d+s}}{\boldsymbol{n}} \quad (1)$$

Where i represents the number of words inserted, d represents the number of words deleted, s represents the number of words substituted and n presents the number of words in the reference text.

We selected the error sentences from the OCR result to generate the corpus by using three criteria: WER lower than or equal to 20%, syllable length lower than or equal to 25, and no non-Thai characters. This corpus will be used to build both the detection model and the correction model.

In this study, we investigated the performance of machine learning and deep learning techniques such as LSTM, Bi-LSTM, CRF, and WangchanBERTa. Our model consists of an error detection phase and an error correction phase. Each phase will be studied separately and be combined to be the spelling correction model.

3.2 Methodology

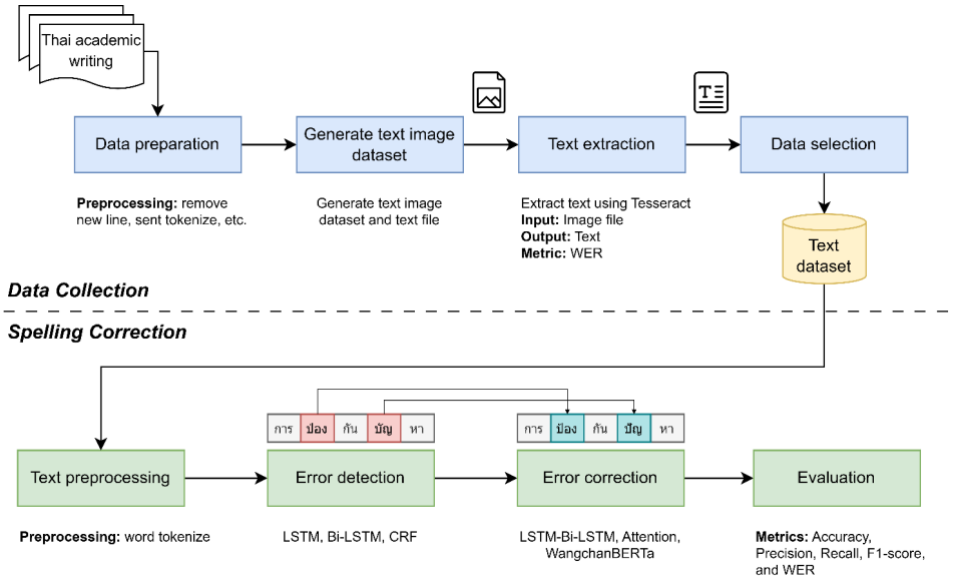

Fig.7: *Overview research framework of our spelling correction model.*

Fig.7 illustrates the proposed research framework including data collection and spelling correction framework. After data collection, there are four steps of spelling correction consist of text preprocessing, error detection, error correction, and evaluation.

The first step is text preprocessing, we processed our dataset by performing the syllable tokenization to create a function for masking error words. Each sentence was split into syllable structures and masked as erroneous syllables. For example, see Fig. 9. The result from masking words was split into syllables and each syllable was labeled as either a correct or erroneous syllable. The data from pre-processing will be used in the next step for each model.

```
Sentence :  รวมถึงแนวทางการป้องกันปัญหาที่เกิดในโครงการ
OCR res :   รวมถึงแนวทางการป่องกันบัญหาที่เกิดในโครงการ
Mask word : รวมถึงแนวทางการ<mask>ป่อง</mask>กัน<mask>บัญ</mask>หาที่เกิดในโครงการ
```

Fig.9: *Sample of masking word.*

The second step is error detection. To build the error detection model, we constructed the neural network architecture and employed machine learning and deep learning techniques such as CRF, LSTM, and Bi-LSTM to investigate their performance. The input of the detection model is the text that contains both erroneous text and correct text. We performed a syllable tokenization to split the text into a sequence of syllables. A sequence of syllables including incorrect input text $\vec{X} = \{x_1, x_2, \ldots, x_n\}$ where n is the total number

of syllables. They are fed into the error detection model to predict the label for each sequence $\vec{Y} = \{y_1, y_2, \ldots, y_n\}$ where y_i is the label prediction of the corresponding syllable x_i. A syllable is labeled as correct or incorrect.

The third step is error correction. A sequence of syllable $\vec{X}$ and label $\vec{Y}$ from the error detection model is the input of the correction model. The error correction model should generate the proper corrected sequence while maintaining the originality of each correct input. We performed Seq2seq and pre-trained models to investigate the performance of error correction. The evaluation metrics for the error correction model are WER to compare with the original result from OCR and correction from the model. We employed deep learning approaches such as LSTM, Bi-LSTM, and Bi-LSTM with Attention mechanism [31] models to investigate the performance of the correction model. Attention mechanism is currently state-if-the-art NLP models which outperformed in machine translation task. We implemented the Attention mechanism in seq2seq model with Bi-LSTM by performing dot scoring function to calculate alignment score. The overview process of error detection and correction was shown in Fig. 8.

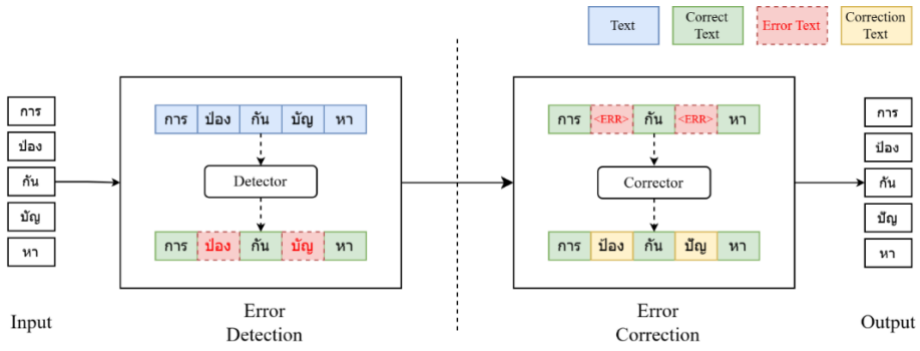

Fig.8: *Overview process of error detection and error correction.*

Finally, model evaluation is the last step. WER was used as the evaluation metric for evaluating the correction model. In addition, the accuracy, precision, recall, and F1-score are also used in the evaluation that are calculated as follow:

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{2}$$

$$Precision = \frac{TP}{TP + FN} \tag{3}$$

$$Recall = \frac{TP}{TP + FP} \tag{4}$$

$$F1\text{-}score = 2 \text{ x } \frac{Precision \times Recell}{Precision + Recell} \tag{5}$$

Where True Positive (TP) represents a spelling error has been detected, False Negative (FN) represents a spelling error has not been detected, False Positive (FP) represents a correctly spelled word was detected as being a misspelled word, and True Negative (TN) represents a correctly spelled word was detected as being a correct word.

For the correction model, True Positive (TP) represents a misspelled word has been corrected, False Negative (FN) represents a misspelled word has not been corrected, False Positive (FP) represents a corrected word has been changed, and True Negative (TN) represents a corrected word has not changed.

4 Results

In this section, we split the experimental result into two phases: error detection and error correction. We compared the performance of each technique with existing techniques from Thai text correction.

4.1 Error Detection

In the error detection phase, we employed deep learning model such as LSTM and Bi-LSTM to train with our training set. We trained the LSTM and Bi-LSTM models to determine the optimal embedding size and hidden units in LSTM and Bi-LSTM layers. The neural network architecture was set the embedding size and hidden units in LSTM and Bi-LSTM layer to 300 units. The number of epochs that was set in this training is 10 epochs.

We performed a grid search to find the optimal parameter for the CRF model. The parameters that were selected are algorithm and max_iterations. Table 3. shows the result of the grid search that uses the F1-score as the evaluation metric.

Table 3: *CRF model grid search results*

Algorithm	Max_iter	F1-score (mean)
ap	100	0.9833
ap	200	0.9837
ap	300	0.9839
ap	**400**	**0.9840**

pa	100	0.9825
pa	200	0.9827
pa	300	0.9828
pa	400	0.9828
lbfgs	100	0.9678
lbfgs	200	0.9763
lbfgs	300	0.9792
lbfgs	400	0.9801

According to Table 3, the Aggressive Perceptron (AP) algorithm and the number of max iterations is 400 are the optimal parameters for this model. We trained the CRF model with the optimal parameters.

Table 4: *Comparative performance of detection models.*

Method	Train				Test			
	Acc.	Prec.	Rec.	F1.	Acc.	Prec.	Rec.	F1.
LSTM	0.9800	0.9437	0.8788	0.9101	0.9801	0.9443	0.8784	0.9101
Bi-LSTM	**0.9848**	**0.9611**	0.9045	**0.9319**	**0.9848**	**0.9615**	0.9042	**0.9320**
CRF	0.9847	0.9574	**0.9073**	0.9317	0.9848	0.9577	**0.9076**	0.9320

Table 4. shows the comparative performances of the detection model using LSTM, Bi-LSTM, and CRF. The performance of all three models is quite high, and there is not much difference. Bi-LSTM outperforms with 99.11% accuracy, 96.11% precision, and 93.19% F1-score. CRF model achieved 93.32% F1-score and LSTM model achieved 91.01% F1-score.

4.2 Error Correction

In the correction phase, we presented the experimental result of all methods on the same dataset. We trained five models to compare the performance of them, including the encoder-decoder using Bi-GRU, encoder-decoder using LSTM network, the encoder-decoder using Bi-LSTM network, the encoder-decoder using Attention mechanism, and WangchanBERTa finetune. The encoder-decoder model consists of an embedding layer with a size of 300, an LSTM or Bi-LSTM encoder with 1,024 nodes, an LSTM decoder with 1,024 nodes, and an output dense projection layer with a SoftMax activation function. The optimizer is Adam with a learning rate of 0.001 on the cross-entropy loss. Another model using Attention mechanism consists of an embedding layer with a size of 300, a Bi-LSTM encoder with

512 nodes in each direction, an LSTM decoder with 1024 nodes, and the Attention layer that score was calculated by the simple dot product. The optimizer is Adam with a learning rate of 0.0003 on the cross-entropy loss.

Our proposed model has employed the tokenizer and model from wangchanberta-base-att-spm-uncased. The tokenizer has been updated with new specific tokens, and the max length is set to 50. The model has set the embedding layer size equal to the number of tokens in the tokenizer. The model was trained with a learning rate of 0.00002 and 10 epochs. Table 5. shows the comparative performances of the correction model using LSTM, Bi-LSTM, Bi-LSTM with Attention mechanism, and WangchanBERTa. The model using the Attention mechanism outperformed all of them with an 86.31% F1-score. WangchanBERTa achieved 81.36% F1-score.

In addition, WangchanBERTa outperformed the other models in terms of precision. shows that WangchanBERTa is more effective at accurately correcting with a lower rate of false positives. WangchanBERTa's superior language understanding abilities, which are optimized for specific tasks and help it to more accurately recognize relevant components and context in the data. On the other hand, the Attention model may not have the same level of contextual understanding or robustness when dealing with complex language constructs particular to spelling correction.

Table 5: *Comparative performance of correction models.*

Method	Train				Test			
	Acc.	Prec.	Rec.	F1.	Acc.	Prec.	Rec.	F1.
LSTM	0.8929	0.5261	0.7331	0.6103	0.8934	0.5271	0.7345	0.6114
Bi-LSTM	0.8979	0.6006	0.6852	0.6335	0.8979	0.6016	0.6865	0.6345
Attention	**0.9715**	0.9494	**0.7879**	**0.8611**	**0.9720**	0.9506	**0.7904**	**0.8631**
WangchanBERTa	0.9646	**0.9989**	0.6862	0.8129	0.9647	**0.999**	0.6872	0.8136

Table 6. shows the comparative performances of the correction model with Thai TC techniques. The Bi-LSTM with Attention mechanism outperformed with 4.03% WER. WangchanBERTa achieved 4.51% WER, Bi-LSTM achieved 11.26% WER, and LSTM achieved 12.49% WER.

Table 6: *Evaluation result of correction models.*

Method	WER	
	Train	Test
LSTM	0.126±0.0234	0.1249±0.0234
Bi-LSTM	0.113±0.0658	0.1126±0.0672
Attention	**0.0377±0.0013**	**0.0403±0.0013**

WangchanBERTa	0.0456±0.0043	0.0451±0.0045

The result of error correction was evaluated by using WER, and the result of each model was compared with the OCR result. The Seq2seq with LSTM, and Bi-LSTM create a few correct Thai sentences and possess altered meanings. An analysis of the output indicates that the model tends to favor producing sentences or phrases that are prevalent in the training dataset. The Seq2seq model with Attention mechanism frequently generates accurate Thai sentences and outperformed which has WER of 4.03%. However, the performance of our proposed model is not significantly different from the Attention mechanism. WangchanBERTa can reduce the WER on the test set from 11.99% to 4.51%. The evaluation result is shown in Table 7.

Table 7: *Evaluation of the correction models.*

Model	WER (%)	Δ WER (%)
OCR result	11.99	-
LSTM	12.49	4.17
Bi-LSTM	11.26	-6.09
Attention	**4.03**	**-66.39**
WangchanBERTa	4.51	-62.39

In addition, we compared the execution time for the correction of each model. We ran this testing on Google Colab with GPU runtime, the GPU timing information is on an NVIDIA Tesla T4. The time required by each model to perform inference on the test set is shown in Table 8. There are 26,751 samples in the test set that were used to test the performance of error correction. Our proposed model has the fastest inference time. Although the Attention model performed the best, it took the longest time to infer.

Table 8: *Inference time on the test set.*

Model	Execution time for testing	Time per line (*ms*)	Line per second
LSTM	0:20:28	45.92	21.78
Bi-LSTM	0:21:45	48.77	20.51
Attention	6:28:01	870.29	1.15

WangchanBERTa	**0:09:16**	**20.80**	**48.08**

The performance of the Attention mechanism and WangchanBERTa is quite high and not different significantly. However, when compared to the inference time, WangchanBERTa takes 9 minutes while the Attention mechanism takes 6 hours on our test set.

5 Conclusion

In this study, we investigated the performance of various techniques in Thai spelling correction. Our dataset is based on the error text from the Thai OCR results that were collected from Thai academic writing. We performed post-processing on the results from the original OCR because they contain many errors for Thai characters. Our spelling correction model consists of two-phases: error detection phase and error correction phase.

Our investigation into error detection shows high performance for each technique, such as LSTM, Bi-LSTM, and CRF. Their performances achieve an average accuracy of 90%. In the context of sequence labeling tasks, Bi-LSTM and stand out as widely employed and effective techniques. Both methods have demonstrated considerable success in various natural language processing tasks, owing to their ability to capture complex patterns and dependencies within sequential data. Their utilization in the current study is well-founded, given their proven track record in similar applications.

In the investigation of error correction, we investigated the performance machine learning and deep learning models, such as LSTM, Bi-LSTM, Bi-LSTM with Attention mechanism, and WangchanBERTa. We evaluated the performance and inference time of each technique. The Bi-LSTM with Attention mechanism demonstrated superior performance compared to WangchanBERTa. However, WangchanBERTa is faster in terms of inference time compared to Bi-LSTM models due to its transformer architecture and inherent parallelism. Token processing in Bi-LSTM models is sequential, whereby the representation of each token depends on the preceding token, thereby resulting in slower computations. In contrast, the transformer architecture of RoBERTa facilitates token parallel processing, enabling concurrent handling of multiple tokens. Moreover, through the application of self-attention techniques, the model efficiently captures global dependencies without the need for sequential processing. Regarding the suitable application of our proposed method, WangchanBERTa demonstrates several advantageous features, including its capacity to effectively handle long-range dependencies, perform parallel processing (resulting in improved inference time), and utilize effective pre-training techniques. These attributes collectively enable WangchanBERTa to effectively process a wide range of documents with diverse text representation units.

References

[1] S. Pradha, M. N. Halgamuge, and N. Tran Quoc Vinh, "Effective text data preprocessing technique for sentiment analysis in social media data," *Proc. 2019 11th Int. Conf. Knowl. Syst. Eng. KSE 2019*, 2019, doi: 10.1109/KSE.2019.8919368.

[2] S. Hasan, C. Heger, and S. Mansour, "Spelling correction of user search queries through statistical machine translation," *Conf. Proc. - EMNLP 2015 Conf. Empir. Methods Nat. Lang. Process.*, no. September, pp. 451–460, 2015, doi: 10.18653/v1/d15-1051.

[3] O. J. ONI and F. O. ASAHIAH, "Computational modelling of an optical character recognition system for Yorùbá printed text images," *Sci. African*, vol. 9, p. e00415, 2020, doi: 10.1016/j.sciaf.2020.e00415.

[4] M. V. Christanti, Rudy, and D. S. Naga, "Fast and accurate spelling correction using trie and Damerau-levenshtein distance bigram," *Telkomnika (Telecommunication Comput. Electron. Control.*, vol. 16, no. 2, pp. 827–833, 2018, doi: 10.12928/TELKOMNIKA.v16i2.6890.

[5] S. Watcharabutsarakham, "Spell Checker for Thai Document," *TENCON 2005 - 2005 IEEE Reg. 10 Conf.*, pp. 5–8, 2005.

[6] R. K. Chaurasiya, N. D. Londhe, and S. Ghosh, "A Novel Weighted Edit Distance-Based Spelling Correction Approach for Improving the Reliability of Devanagari Script-Based P300 Speller System," *IEEE Access*, vol. 4, pp. 8184–8198, 2016, doi: 10.1109/ACCESS.2016.2614494.

[7] S. Zhang, J. Xiong, J. Hou, Q. Zhang, and X. Cheng, "HANSpeller++: A unified framework for chinese spelling correction," *Proc. 8th SIGHAN Work. Chinese Lang. Process. SIGHAN 2015 - co-located with 53rd Annu. Meet. Assoc. Comput. Linguist. 7th Int. Jt. Conf. Nat. Lang. Process. ACL IJCNLP*, vol. 20, no. 1, pp. 38–45, 2015, doi: 10.18653/v1/w15-3107.

[8] J. Guo, T. N. Sainath, and R. J. Weiss, "A Spelling Correction Model for End-to-end Speech Recognition," *ICASSP, IEEE Int. Conf. Acoust. Speech Signal Process. - Proc.*, vol. 2019-May, pp. 5651–5655, 2019, doi: 10.1109/ICASSP.2019.8683745.

[9] J. Devlin, M. W. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of deep bidirectional transformers for language understanding," *NAACL HLT 2019 - 2019 Conf. North Am. Chapter Assoc. Comput. Linguist. Hum. Lang. Technol. - Proc. Conf.*, vol. 1, no. Mlm, pp. 4171–4186, 2019.

[10] M. Tan, D. Chen, Z. Li, and P. Wang, "Spelling Error Correction with BERT based on Character-Phonetic," *2020 IEEE 6th Int. Conf. Comput. Commun. ICCC 2020*, pp. 1146–1150, 2020, doi: 10.1109/ICCC51575.2020.9345276.

[11] S. Zhang, H. Huang, J. Liu, and H. Li, "Spelling Error Correction with Soft-Masked BERT," pp. 882–890, 2020, doi: 10.18653/v1/2020.acl-main.82.

[12] B. Kruatrachue, K. Somguntar, and K. Siriboon, "Thai OCR error correction

using genetic algorithm," *Proc. - 1st Int. Symp. Cyber Worlds, CW 2002*, vol. 7, pp. 137–141, 2002, doi: 10.1109/CW.2002.1180870.

[13] A. Lertpiya, T. Chalothorn, and E. Chuangsuwanich, "Thai Spelling Correction and Word Normalization on Social Text Using a Two-Stage Pipeline with Neural Contextual Attention," *IEEE Access*, vol. 8, pp. 133403–133419, 2020, doi: 10.1109/ACCESS.2020.3010828.

[14] H. Bast and C. Korzen, "A Benchmark and Evaluation for Text Extraction from PDF," in *2017 ACM/IEEE Joint Conference on Digital Libraries (JCDL)*, 2017, vol. 2022-March, pp. 1–10, doi: 10.1109/JCDL.2017.7991564.

[15] Q. Li, W. An, A. Zhou, and L. Ma, "Recognition of offline handwritten Chinese characters using the tesseract open source OCR engine," *Proc. - 2016 8th Int. Conf. Intell. Human-Machine Syst. Cybern. IHMSC 2016*, vol. 2, pp. 452–456, 2016, doi: 10.1109/IHMSC.2016.239.

[16] H. Sidhwa, S. Kulshrestha, S. Malhotra, and S. Virmani, "Text Extraction from Bills and Invoices," *Proc. - IEEE 2018 Int. Conf. Adv. Comput. Commun. Control Networking, ICACCCN 2018*, pp. 564–568, 2018, doi: 10.1109/ICACCCN.2018.8748309.

[17] B. V. Kakani, Di. Gandhi, and S. Jani, "Improved OCR based automatic vehicle number plate recognition using features trained neural network," *8th Int. Conf. Comput. Commun. Netw. Technol. ICCCNT 2017*, 2017, doi: 10.1109/ICCCNT.2017.8203916.

[18] R. R. Palekar, S. U. Parab, D. P. Parikh, and V. N. Kamble, "Real time license plate detection using openCV and tesseract," *Proc. 2017 IEEE Int. Conf. Commun. Signal Process. ICCSP 2017*, vol. 2018-Janua, pp. 2111–2115, 2017, doi: 10.1109/ICCSP.2017.8286778.

[19] H. T. Ha, M. Medved', Z. Nevěřilová, and A. Horák, "Recognition of OCR invoice metadata block types," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 11107 LNAI, pp. 304–312, 2018, doi: 10.1007/978-3-030-00794-2_33.

[20] V. N. Sai Rakesh Kamisetty, B. Sohan Chidvilas, S. Revathy, P. Jeyanthi, V. M. Anu, and L. Mary Gladence, "Digitization of Data from Invoice using OCR," *Proc. - 6th Int. Conf. Comput. Methodol. Commun. ICCMC 2022*, 2022, doi: 10.1109/ICCMC53470.2022.9754117.

[21] C. Irimia, F. Harbuzariu, I. Hazi, and A. Iftene, "Official Document Identification and Data Extraction using Templates and OCR," *Procedia Comput. Sci.*, vol. 207, no. Kes, pp. 1571–1580, 2022, doi: 10.1016/j.procs.2022.09.214.

[22] H. D. Liem *et al.*, "FVI: An End-to-end Vietnamese Identification Card Detection and Recognition in Images," *NICS 2018 - Proc. 2018 5th NAFOSTED Conf. Inf. Comput. Sci.*, pp. 338–340, 2019, doi: 10.1109/NICS.2018.8606831.

[23] M. Koistinen, K. Kettunen, and J. Kervinen, "How to Improve Optical Character Recognition of Historical Finnish Newspapers Using Open Source Tesseract OCR Engine – Final Notes on Development and Evaluation," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol.

12598 LNAI, no. November, pp. 17–30, 2020, doi: 10.1007/978-3-030-66527-2_2.

[24] H. Lu, B. Guo, J. Liu, and X. Yan, "A shadow removal method for tesseract text recognition," *Proc. - 2017 10th Int. Congr. Image Signal Process. Biomed. Eng. Informatics, CISP-BMEI 2017*, vol. 2018-Janua, pp. 1–5, 2017, doi: 10.1109/CISP-BMEI.2017.8301946.

[25] A. F. de S. Neto, B. L. D. Bezerra, and A. H. Toselli, "Towards the natural language processing as spelling correction for offline handwritten text recognition systems," *Appl. Sci.*, vol. 10, no. 21, pp. 1–29, 2020, doi: 10.3390/app10217711.

[26] R. Smith, "History of the Tesseract OCR engine: what worked and what didn't," *Doc. Recognit. Retr. XX*, vol. 8658, no. February 2013, p. 865802, 2013, doi: 10.1117/12.2010051.

[27] M. M. Alamri and W. J. Teahan, "Automatic correction of arabic dyslexic text," *Computers*, vol. 8, no. 1, pp. 1–19, 2019, doi: 10.3390/computers8010019.

[28] Y. Hu, X. Jing, Y. Ko, and J. T. Rayz, "Misspelling Correction with Pre-trained Contextual Language Model," pp. 144–149, 2021, doi: 10.1109/iccicc50026.2020.9450253.

[29] P. Gupta, "A context-sensitive real-time spell checker with language adaptability," *Proc. - 14th IEEE Int. Conf. Semant. Comput. ICSC 2020*, pp. 116–122, 2020, doi: 10.1109/ICSC.2020.00023.

[30] R. Grundkiewicz and M. Junczys-Dowmunt, "Near human-level performance in grammatical error correction with hybrid machine translation," *NAACL HLT 2018 - 2018 Conf. North Am. Chapter Assoc. Comput. Linguist. Hum. Lang. Technol. - Proc. Conf.*, vol. 2, pp. 284–290, 2018, doi: 10.18653/v1/n18-2046.

[31] M. T. Luong, H. Pham, and C. D. Manning, "Effective approaches to attention-based neural machine translation," *Conf. Proc. - EMNLP 2015 Conf. Empir. Methods Nat. Lang. Process.*, pp. 1412–1421, 2015, doi: 10.18653/v1/d15-1166.