applied sciences

MDPI

Article

Enhance Text-to-Text Transfer Transformer with Generated Questions for Thai Question Answering

Puri Phakmongkol and Peerapon Vateekul *

Department of Computer Engineering, Faculty of Engineering, Chulalongkorn University, Bangkok 10300, Thailand; puri.pmk@gmail.com
* Correspondence: peerapon.v@chula.ac.th

Abstract: Question Answering (QA) is a natural language processing task that enables the machine to understand a given context and answer a given question. There are several QA research trials containing high resources of the English language. However, Thai is one of the languages that have low availability of labeled corpora in QA studies. According to previous studies, while the English QA models could achieve more than 90% of F1 scores, Thai QA models could obtain only 70% in our baseline. In this study, we aim to improve the performance of Thai QA models by generating more question-answer pairs with Multilingual Text-to-Text Transfer Transformer (mT5) along with data preprocessing methods for Thai. With this method, the question-answer pairs can synthesize more than 100 thousand pairs from provided Thai Wikipedia articles. Utilizing our synthesized data, many fine-tuning strategies were investigated to achieve the highest model performance. Furthermore, we have presented that the syllable-level F1 is a more suitable evaluation measure than Exact Match (EM) and the word-level F1 for Thai QA corpora. The experiment was conducted on two Thai QA corpora: Thai Wiki QA and iApp Wiki QA. The results show that our augmented model is the winner on both datasets compared to other modern transformer models: Roberta and mT5.

Keywords: natural language processing; question answering; machine reading comprehension

Citation: Phakmongkol, P.; Vateekul, P. Enhance Text-to-Text Transfer Transformer with Generated Questions for Thai Question Answering. *Appl. Sci.* **2021**, *11*, 10267. https://doi.org/10.3390/app112110267

Academic Editor: Arturo Montejo-Ráez

Received: 20 September 2021
Accepted: 27 October 2021
Published: 1 November 2021

1. Introduction

One of the Natural Language Processing (NLP) tasks that allow machines to understand the information in text format and answer given questions is Question Answering (QA). Many researchers aim to develop QA systems in many languages because QA systems have many benefits and can be used as a part of many intelligent systems such as chat bots, or answer highlighters in search engines. One of the most popular languages developed in QA tasks is English. There are many techniques and machine learning models as well as many language resources that contribute to QA system development in the English language. For example, the Text-to-Text Transfer Transformer model [1], a Transformer-based model [2] that was trained with the huge English dataset called Colossal Clean Crawled Corpus (C4) [3], achieved state-of-the-art results in SQuAD 1.1 [4] with an F1 score of 96.22%. These contributions can support English QA models to reach higher performance than other languages.

There are several research works about Thai QA, for example, using heuristic functions to extract the answer, developed by Hatsanai Decha et al. [5], and using the Bi-Directional Attention Flow (BiDAF) model [6] developed by Theerit Lapchaicharoenkit et al. [7]. However, one of the most important limitations of Thai QA is a lack of availability of training data. There are currently only two datasets of Thai QA: Thai Wiki QA [8] and iApp Wiki QA. Each sample of both datasets consists of a context, a question, and a ground truth answer. Both datasets are span extraction type such that the answer to the question is the span of text in the corresponding context. Thai Wiki QA contains 15,000 samples while iApp Wiki QA contains 7242 samples. Each dataset has a small number of samples

compared to an English span extraction dataset such as SQuAD 1.1, which contains more than 100 thousand samples. With this limitation, directly using the same techniques or models of the English language such as deep learning models with Thai corpora might not be able to utilize the capability of models to raise the performance.

In this paper, we aim to improve the Thai QA model performance by presenting an enhanced QA framework tailored for the Thai language, with low training resources. First, the limitation of data is overcome by generating synthesized data using Raul Puri et al.'s method [9]. We further investigated and improved their technique in many aspects: the synthesized data selection (all vs. filtered data) and the fine-tuning strategies (merge and sequence). Second, we employed recent transformer models, where the pretrained weights supported the Thai language. There are two chosen models in our comparison: WanchanBERTa [10] and Multilingual Text-to-Text Transfer Transformer (mT5) [11]. Third, we presented preprocessing methods for the Thai language to reduce the misspelling words as well as to improve the quality of data. Final, the metrics that are widely used in QA tasks for evaluating model performance, such as Exact Match (EM) and F1 score, are not sufficient due to inabilities of the word tokenizer and the ambiguity of the Thai language. To obtain nearer-correct scores, we proposed a Syllable-level F1 that calculates the F1 score with syllable-tokens of prediction and the ground truth instead of word-tokens. In this work, we evaluated the models with syllable-level F1 along with word-level F1. The details of each module in our framework are explained in Chapter 3. The experiment was conducted on two Thai QA corpora: Thai Wiki QA and iApp Wiki QA. The results showed that the synthesized data along with a sequence fine-tuning strategy outperformed the original Transformer based models.

In summary, our contributions are as follows:

- We present a data preprocessing method for the Thai Language.
- We demonstrate fine-tuning of two Transformer based models, WangchanBERTa and mT5, for the QA task, with synthesized data and real human-labeled corpus, and achieve higher EM and F1 scores than those when using only the real human-labeled data.
- We compare the quality of the generated question-answer pairs used in the QA models as well as training strategies.
- We propose new metrics: Syllable-level F1 to evaluate the models along with the original Word-level F1.

We organize the rest of this paper as follows. Related works are introduced in Section 2, followed by the presentation of our proposed framework in Section 3. We then explain our experiment settings in Section 4. The result and discussion are presented in Sections 5 and 6, and finally the conclusion of our work in Section 7.

2. Literature Review

In this section, we introduce related research to our work. This section is divided into five parts as follows: recent research on QA, research on Thai QA, data augmentation methods, the Text-to-Text Transfer Transformer, and the WangchanBERTa model.

2.1. Recent Research in Question Answering

Most recent research works on NLP focus on developing language models to use with many tasks including the QA task. Most language models use the Transformer model as a part of their processing because the Transformer model has proved that it can reach higher performance than older-style NLP models, such as BiDAF [6], that use Long Short-Term Memory (LSTM) [12]. BERT [13] is the first Transformer based language model that uses only the encoder part of the Transformer. There are two sizes of BERT models: $BERT_{Base}$ with 12 layers of Encoder and $BERT_{Large}$ with 24 layers of Encoder. In the experiment, BERT could achieve state-of-the-art performance in QA tasks. $BERT_{Base}$ could reach 80.8 and 88.5 EM and F1 scores, respectively, $BERT_{Large}$ could also reach

84.1 and 90.9 EM and F1 scores, respectively, when tested with SQuAD 1.1, while BiDAF could achieve only 68.0 and 77.3 EM and F1 scores, respectively.

There were several trials to develop a BERT model to better predict span, which is more appropriate with QA tasks. SpanBERT [14] is one of these. SpanBERT changed some functions of the pretraining process by using span masking instead of token masking, and adding a Span Boundary Objective to train the model to predict the masked span with adjoining words. SpanBERT used $BERT_{Large}$ architecture and applied a described method. SpanBERT was tested with the SQuAD 1.1 dataset to evaluate its performance, and achieved 88.8 and 94.6 EM and F1 scores, respectively.

However, neither of those developments was chosen to use in our work because WangchanBERTa was considered a better model than BERT, and SpanBERT must be pretrained with Thai documents before using, which is not convenient to use.

2.2. Researches in Thai Question Answering

Hatsanai Decha et al. [5] developed a QA system in Thai with a keyword extraction method by finding keywords from questions and using extracted keywords to find candidate answers from a set of contexts, then finding the best answer with a heuristic function called word order consistency, which functions in a manner that measures similarity between contexts and questions. This work does not use deep learning model, it is thus not directly related to our work.

Theerit Lapchaicharoenkit et al. [7] modified the BiDAF model to support two types of questions, span extraction type and yes-no question type, by adding a question type classifier to the model. The model also used contextualized word embedding from the BERT model that was pretrained with only Thai documents. The model was tested in a competition called the National Software Contest organized in Thailand in 2018–2019. This competition dataset consisted of 15,000 samples of span extraction tasks and 2000 samples of yes-no question tasks.

Nevertheless, we did not use both above-mentioned works in our research because the first method was not related to our work, and Transformer based models have proved that they could achieve better performance than BiDAF in QA tasks.

2.3. Data Augmentation Methods

There are several research works in data augmentation for improving the performance in QA tasks. Bhuwan Dhingra et al. [15] presented a cloze-style question generation method by extracting questions and answers using the document structure of English articles that mostly provides the summary of articles in the introduction. They used the BiDAF model as a QA model. This method was able to raise the EM and F1 evaluation scores by 0.32% and 0.11%, respectively.

Raul Puri et al. [9] introduced a Question Generation pipeline with three Transformer based models inside. There are three steps of the pipeline including (1) answer generation, (2) question generation, and (3) question filtration. Answer generation is performed by a BERT model trained to select the candidate answer from a given context. Question generation is performed by a GPT-2 model [16] trained to create a proper question to a given context and answer. The last step, Question filtration, is performed by a BERT model trained with question answering objectives with human-labeled data. The researchers used this model to predict an answer from the generated question and context. If the answer from this model was equivalent to the answer from the answer generation step, they considered the generated question-answer pair to be an admissible sample. With this pipeline, they were able to generate more than 19 million question-answer pairs from Wikipedia articles, and used them to train the BERT model. The result achieved more than their baseline EM and F1 scores by 1.7% and 1.2%, respectively.

To conclude, the first method cannot be used with Thai articles because the Thai article structure is more ambiguous than English. It cannot simply extract the answers and

questions by using heuristic rules. Given this limitation, using a deep learning model to extract answers and questions is a more appropriate method to synthesize the data.

2.4. The Text-to-Text Transfer Transformer Model

The Text-to-Text Transfer Transformer (T5) [1] is one of the Transformer based models that uses the same architecture of Transformer as shown in Figure 1. The objective of T5 models is to support every NLP task by treating every text processing problem as a "text-to-text" task, by taking the given text as input and producing new text as output. With this method, many tasks could be used with this model, for example, Question Answering, document summarization, or sentiment classification. There is research work that uses a set of documents containing 101 languages, including Thai, to pretrain T5 models called Multilingual Text-to-Text Transfer Transformer (mT5) [11].

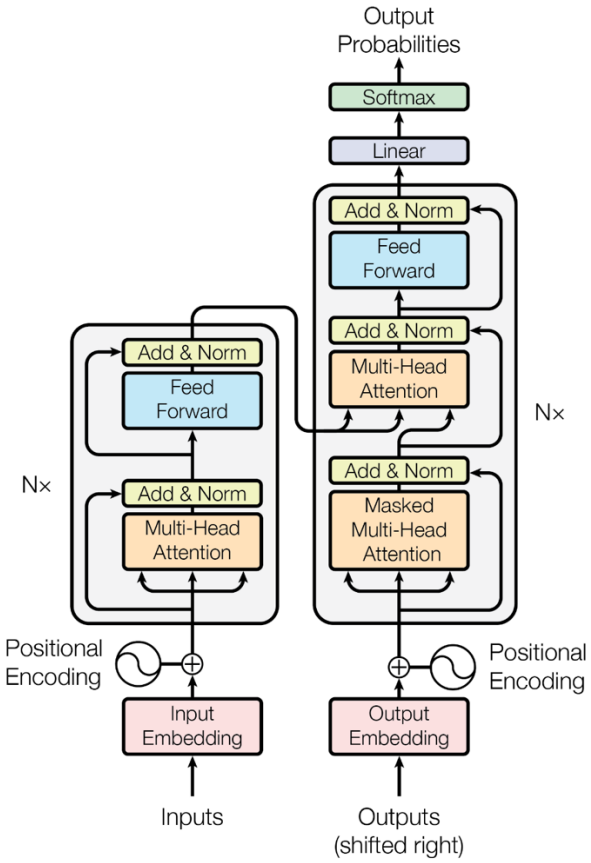

Figure 1. Model architecture of Transformer [2] that was used in the mT5 model.

Due to the model's ability to be used with various tasks, this model was used in our research in both the question generation and question answering parts.

2.5. The WangchanBERTa Model

WangchanBERTa [10] is a pretrained language model based on the Roberta [17] configuration. The architecture of Roberta is the same as that of BERT in terms of using only the Encoder part of the Transformer model as shown in Figure 2. WangchanBERTa was pretrained on a large set of Thai documents including social media texts, news, and public articles. In addition, the appropriate methods were applied to the texts before training. The result showed that this model beat other Thai supported Transformer based models, such as Multilingual BERT, on many downstream tasks. We used this model in question answering part to compare with the mT5 model.

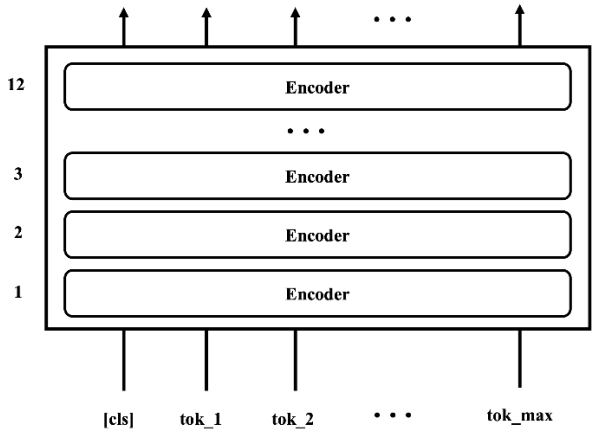

Figure 2. Model architecture of BERT that uses the Encoder part of the Transformer model. (Note: This architecture is also used in the Roberta model.)

3. Proposed Method

This section explains the components of the proposed QA framework. For example, preprocessing methods for Thai texts, question-answer pairs generation, training strategies of QA models and model evaluation. The components of the framework are illustrated in Figure 3.

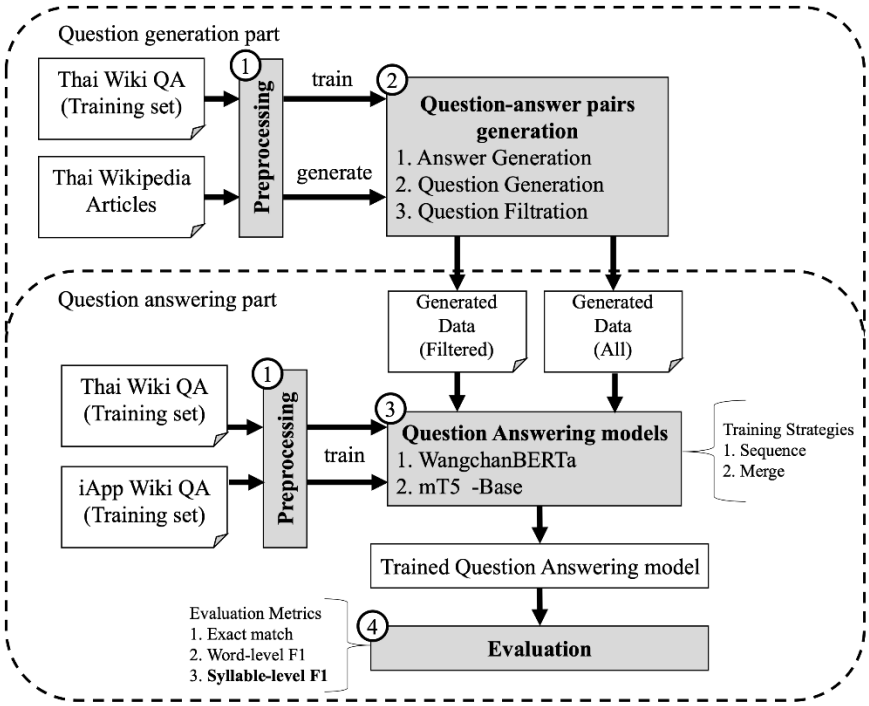

Figure 3. Illustration of the proposed overall QA framework.

3.1. Preprocessing Methods for Thai Texts

All Thai texts must be preprocessed with appropriate methods before being used in training and testing with the models. The first step is applying lowercase characters to the text in case there are English characters in the text. The second step is normalizing the text into the correct and standard form by removing duplicate characters, and changing the order of word typing to the correct one. With this step, we could reduce misspelled words in the datasets, which enables the model to work more accurately. We used the implementation of PyThaiNLP's normalization function [18] for normalizing texts as described.

3.2. Question-Answer Pairs Generation

The method for generating question-answer pairs is based on Raul Puri et al.'s method [9], which consists of three steps: (1) Answer Generation, (2) Question Generation, and (3) Question Filtration. This method is able to generate a set of triplets which include Context $\boldsymbol{c}$, Question $\boldsymbol{q}$ and Answer $\boldsymbol{a}$ by using a given set of Articles $\boldsymbol{A}$, pursuant to Probability $\boldsymbol{p}(\boldsymbol{q}, \boldsymbol{a}|\boldsymbol{c})$.

The difference of implementation between Raul Puri et al.'s work and our work is the selection of the base models in the Question Generation pipeline. In our work, we used the same type of models corresponding to the original work, WangchanBERTa for BERT and mT5 for GPT-2, but our models support the Thai language. The summaries of the different models are shown in Table 1. However, we used the mT5 model instead of WangchanBERTa in the Answer Generation step because we found that the mT5 model could generate more appropriate answers than WangchanBERTa.

Table 1. Difference of implementation between Raul Puri et al.'s work and ours in the Question Generation pipeline.

Implementation	Answer Generation	Question Generation	Question Filtration
Raul Puri et al.	BERT	GPT-2	BERT
Our	mT5-Large	mT5-Large	WangchanBERTa

3.2.1. Step 1: Answer Generation

Due to the difficulty and ambiguity of the Thai Language, extracting answer candidates from heuristic rules is not sufficient to select the high-quality answers because natural language processing tools for Thai do not perform correctly in every word or sentence; sometimes word features from the given text are extracted incorrectly.

To overcome this limitation, the Answer Generation Model—$\boldsymbol{p}(\boldsymbol{a}|\boldsymbol{c})$ was used to select an appropriate word to be an Answer $\hat{\boldsymbol{a}}$ of a sample. Unlike Raul Puri et al.'s implementation, we fine-tuned the mT5-Large model by using Context $\boldsymbol{c}$ as an input of the model to learn the answer distribution of the dataset as shown in Figure 4. The answer was selected by the highest probability score.

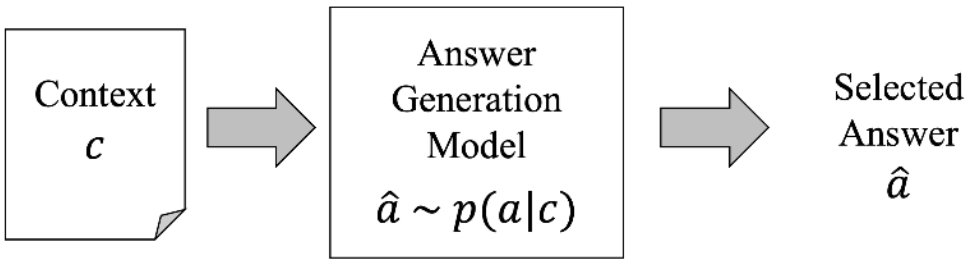

Figure 4. The input and the output of the Answer Generation Model.

3.2.2. Step 2: Question Generation

In this step, the Question Generation Model—$\boldsymbol{p}(\boldsymbol{q}|\hat{\boldsymbol{a}}, \boldsymbol{c})$ was trained to a generated question in accordance with a given context and answer. We fine-tuned the mT5-Large model by using Context $\boldsymbol{c}$ and selected Answer $\hat{\boldsymbol{a}}$ from Answer Generation Model as inputs as shown in Figure 5.

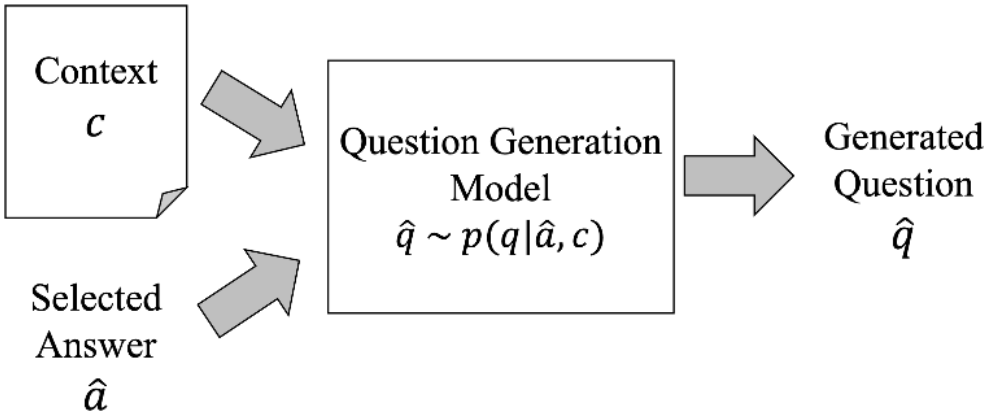

Figure 5. The inputs and the output of the Question Generation Model.

3.2.3. Step 3: Question Filtration

After obtaining a Generated Question $\hat{q}$ from Question Generation Model and an Answer $\hat{a}$ from Answer Generation Model, we already obtained a triplet of generated data $(c, \hat{q}, \hat{a})$. Before using a sample from the generated data, we must verify if this triplet is admissible. To achieve this, we trained a Question Filtration model in the question answering task with labeled training data. After that, we applied the generated Question $\hat{q}$ and Context c to the Question Filtration model for predicting the Answer $\tilde{a}$ as shown in Figure 6. We then compared the Answer $\tilde{a}$ from the model with Answer $\hat{a}$ from the triplet. If these two answers are equivalent, then this triplet is considered an admissible and high-quality sample. Thus, the process of generating a question-answer pair is illustrated in Figure 7.

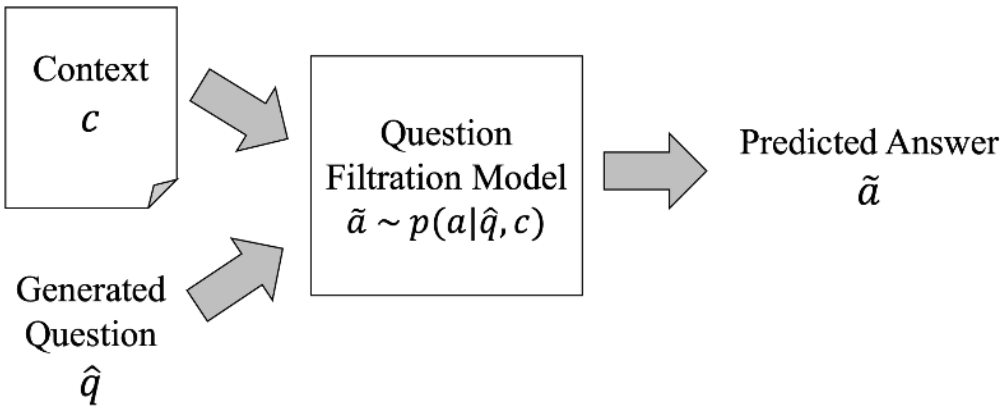

Figure 6. The inputs and the output of the Question Filtration model.

In this part, we selected to use WangchanBERTa as a base model because this model is similar to Raul Puri et al.'s work that used BERT as a base model. Moreover, the WangchanBERTa model is more proper to use in Thai because it was pretrained with Thai documents. Before using it, we fine-tuned the question answering task to this model with Thai QA datasets as we describe in Section 4.1.

In conclusion, we compared two types of generated data: (1) filtered generated data, and (2) all generated data in the experiment. The 'filtered generated data' is the set of samples $(c, \hat{q}, \hat{a})$ that passes the Question Filtration step while the 'all generated data' is the set of triplets $(c, \hat{q}, \hat{a})$ after passing Question Generation step, whether it passes the Question Filtration step or not.

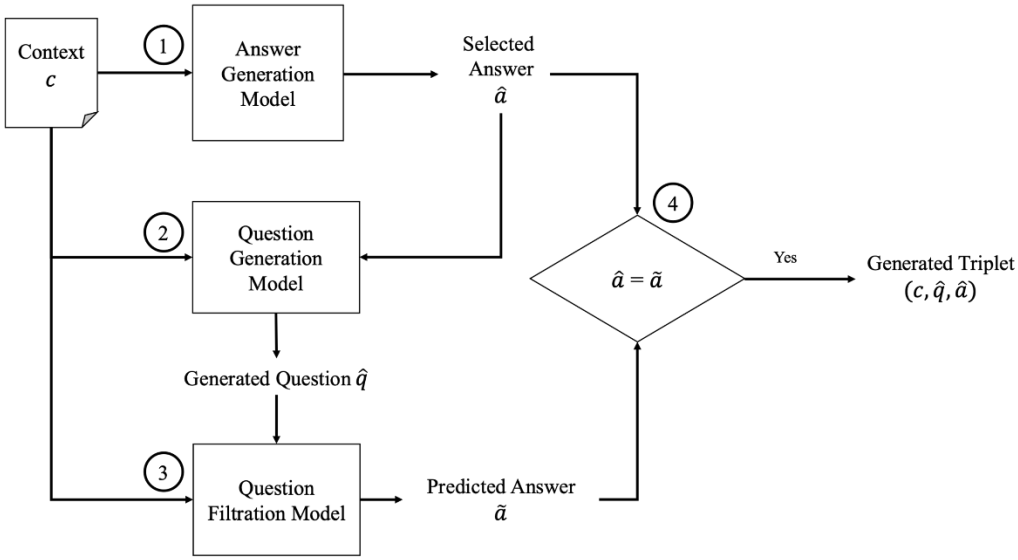

Figure 7. Illustration of the Question Generation pipeline.

3.3. Question Answering Models Training

In QA model training, we selected two Transformer based models as a baseline QA model: WangchanBERTa and mT5. In addition, we compared two training strategies for fine-tuning QA models with generated data and real human-labeled data.

The first training strategy is the Sequence Strategy, which involves sequentially fine-tuning the generated data, followed by the real human-labeled training data. The Sequence Strategy process is illustrated in Figure 8.

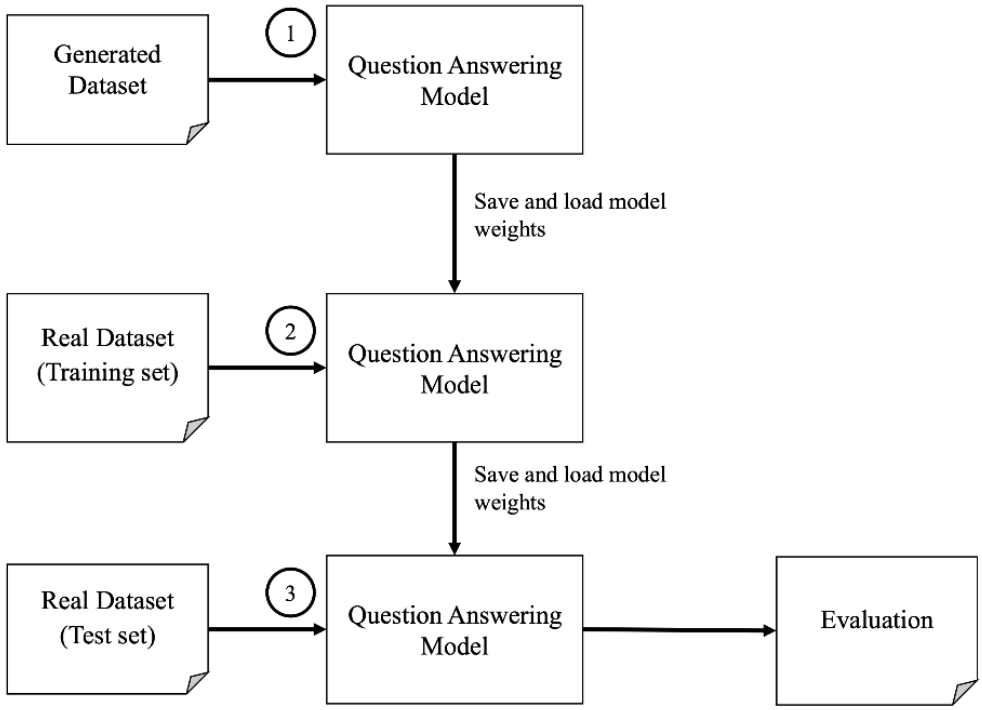

Figure 8. Illustration of the training flow of the Sequence Strategy.

The other training strategy is Merge Strategy, which merges the generated data and the real training data, and fine-tunes at the same time, as illustrated in Figure 9.

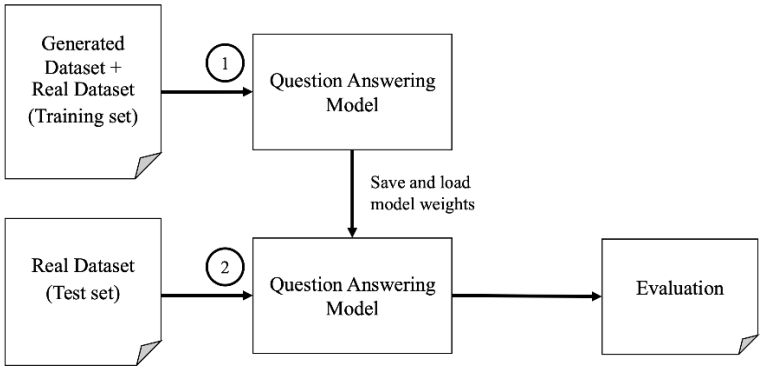

Figure 9. Illustration of the training flow of the Merge Strategy.

3.4. Model Evaluation

We used the F1 score and Exact Match (EM), which are widely used in span extraction Question Answering tasks [19] to evaluate the performance of models. The Exact Match measures how much the model is able to retrieve the exact ground truth span correctly in the whole dataset. The F1 score is a harmonic mean of precision and recall of prediction compared to the ground truth. Originally, to measure the precision and the recall, we count the number of words found in both the prediction and the ground truth.

To calculate the F1 score, the equations below were used. *TP* refers to 'True Positive', that counts the tokens appearing in both the prediction and the ground truth. *FP* refers to 'False Positive' that counts the tokens that appear only in the prediction. *FN* refers to 'False Negative', which means the number of the tokens that appear only in the ground truth. The F1 score of the dataset is an average of the F1 score of every sample.

$$Precision = \frac{TP}{TP + FP} \tag{1}$$

$$Recall = \frac{TP}{TP + FN} \tag{2}$$

$$F1 = \frac{2 \times Precision \times Recall}{Precision + Recall} \tag{3}$$

$$F1 = \frac{\sum_{i=1}^{N} F1_i}{N} \tag{4}$$

Due to the imperfections of the Thai word tokenizer, measuring at the word-level might not be sufficient. In English, there are space separators between words that make English easier to be tokenized into words. On the other hand, the Thai language is more ambiguous as there is no space between words. Thus, the Word-level F1 depends on the quality of the tokenizer used. To overcome this, we also calculated the F1 score at the syllable-level.

The Syllable-level F1 score, the F1 score that calculates based on syllable tokens, is a more appropriate metric than the Word-level F1 score for a language ambiguous to segment because of the following reasons. First, due to the quality of word tokenizers, using different word tokenizers may result in different F1 scores and cause the score to be unable to be compared with other works. Secondly, because of the imperfection of word tokenizers, there are still mistakes when segmenting some similar words. Lastly, due to the ambiguity of the Thai language, some Thai words can be tokenized in many ways, especially the proper nouns. In contrast, using syllables to calculate scores is less ambiguous because there is only a way to segment a word into syllables that maintains a unit of pronunciation.

In this experiment, we used the 'newmm' tokenizer [18] that is currently one of the fastest and the most reliable word tokenizers for the Thai language. However, as shown inthe example in Table 2, the 'newmm' tokenizer could not tokenize the word into a

proper form. To address this problem, we evaluated the predictions with Syllable-level F1 along with Word-level F1. Using syllable tokens to calculate the F1 score could obtain a more accurate score to the linguistic word segmentation than word tokens because the syllable tokenizer can extract overlapping words into pieces of syllables while the word tokenizer cannot.

Table 2. Example of tokenization used in this work.

	Original Word	Human-Tokenized	Use Word Tokenizer (newmm)	Syllable Tokenizer
Ground truth	*มลายู* (Malay language in short term)	*มลายู*	*มลายู*	*ม / ลา / ยู*
Prediction	*ภาษามลายู* (Malay language)	*ภาษา / มลายู*	*ภาษามลายู*	*ภา / ษา / ม / ลา / ยู*
F1 Score		**66.6**	**0.0**	**77.49**

Similar to the English language, Thai words can have one or more syllables. Tokenizing a word into syllables means dividing the word by a unit of pronunciation that has one vowel sound. For example, in Table 2, the word "*มลายู*" (Malay language in the short term) can be pronounced as /mala:ju:/ which has three syllables as "*ม / ลา / ยู*"; each piece can be pronounced as /ma/, /la:/ and /ju:/ respectively. Another example is the word "*ภาษา*" (language), which can be pronounced as /pa:sa:/. This word has two syllables as "*ภา / ษา*"; each piece can be pronounced as /pa:/ and /sa:/ sequentially.

4. Experiment Setup

In this section, we describe the datasets used in the experiments, tools and parameter setup as follows.

4.1. Datasets

There are two Thai QA corpora used in our experiments: Thai Wiki QA and iApp Wiki QA. The dataset statistics of both datasets are shown in Table 3.

Table 3. Datasets splitting for experiments.

Dataset	No. of Training Set	No. of Validation Set	No. of Test Set
Thai Wiki QA	9045	1005	4950
iApp Wiki QA	5761	742	439

Thai Wiki QA [8] is a SQuAD-like dataset in the Thai language. It was used as a QA competition dataset in Thailand National Software Contest (NSC), during 2018–2019. This dataset consists of 15,000 question-answer pairs with contexts from Thai Wikipedia and annotated by 15 native Thai speakers with many kinds of expertise and education levels. The publisher of Thai Wiki QA also published 125,302 Thai Wikipedia articles to support this dataset as an open domain QA task. In this study, we also used the published articles for generating more question answering samples.

iApp Wiki QA (https://github.com/iapp-technology/iapp-wiki-qa-dataset (accessed on 10 September 2021)) is a SQuAD-like dataset published by iApp Technology Company Limited. This dataset includes 7242 question-answer pairs made with Thai Wikipedia articles. However, the publisher of this dataset does not provide information about the data annotation method.

4.2. Tools and Parameter Setup

For all implementations of models including Question Generation and Question Answering parts, we used HuggingFace's Transformers [20] for model developments and training, including model architecture, model configuration and model weights. HuggingFace also provided model training tools. All models in our research used default training arguments provided by HuggingFace, except the learning rate, batch size, weight decay, and number of epochs for training. We changed the value of the learning rate to 10^{-6}, the weight decay to 0.01, and the number of epochs to 25. We also changed the value of batch size to 12 if the trained model was WangchanBERTa, and to 4 if the trained model was mT5-Base and mT5-Large.

We selected the number of batch size configurations based on technical reasons. Our system, DGX A100 with NVIDIA A100 GPU, could use only a batch size of 4 for training mT5-Base and mT5-Large models due to its enormous trainable parameters; 580 M for mT5-Base and 1.2 B for mT5-Large, while the WangchanBERTa model has only 110 M of parameters. Thus, we selected a batch size of 12 for training the WangchanBERTa model to decrease disparities and make them comparable.

The other apparatus used in this research was PyThaiNLP, a Thai natural language processing toolkit. We used this tool for applying text preprocessing before applying the text to the models. In addition, this tool provides the Thai syllable tokenizer and text tokenizers used in this research, such as the 'newmm' tokenizer, which is a fast and reliable Thai word tokenizer.

5. Results

In this section, we report the results of the experiments in several aspects. First, we explain the overall results by comparing every combination of QA models, training strategies, and generated data. The overall results correspond to Tables 4 and 5, which are the main results, and Table 6, which shows the dataset statistics of the augmented data. Secondly, we explain the performance related to training strategies. Thirdly, we describe the comparison of the generated data used. Fourthly, we present the comparison of base QA models. Lastly, we explain the results of the Syllable-level F1 score and provide some samples from the test set calculated Syllable-level F1 score.

Table 4. The experiment result of our method compared to baseline models of Thai Wiki QA. (EM, W-F1, and S-F1 refer to Exact Match, word-level F1 and syllable-level F1 respectively. Boldface refers to the winner.).

Thai Wiki QA	Baseline			+ Filtered Generated Pairs (FLT)			+ All Generated Pairs (ALL)		
	EM	**W-F1**	**S-F1**	**EM**	**W-F1**	**S-F1**	**EM**	**W-F1**	**S-F1**
WangchanBERTa (WBT)									
Sequence Strategy (SEQ)	43.92	70.73	74.71	45.90	73.35	77.55	46.48	74.40	78.60
Merge Strategy (MRG)				44.63	71.11	75.15	43.35	71.09	75.26
mT5-Base (mT5)									
Sequence Strategy (SEQ)	64.14	78.24	80.35	**70.42**	**83.64**	**85.29**	69.03	83.02	84.74
Merge Strategy (MRG)				69.01	82.88	84.68	63.66	79.30	81.17

Table 5. The experiment result of our method compared to baseline models of iApp Wiki QA. (EM, W-F1, and S-F1 refer to Exact Match, word-level F1 and syllable-level F1 respectively. Boldface refers to the winner.).

iApp Wiki QA	Baseline			+ Filtered Generated Pairs (FLT)			+ All Generated Pairs (ALL)		
	EM	**W-F1**	**S-F1**	**EM**	**W-F1**	**S-F1**	**EM**	**W-F1**	**S-F1**
WangchanBERTa (WBT)									
Sequence Strategy (SEQ)	30.58	69.68	71.81	32.88	71.81	74.42	33.15	72.98	75.14
Merge Strategy (MRG)				29.77	69.59	71.97	31.66	70.37	72.82
mT5-Base (mT5)									
Sequence Strategy (SEQ)	29.36	63.46	63.71	56.02	81.31	82.58	**58.05**	**81.97**	**83.15**
Merge Strategy (MRG)				57.10	81.42	82.57	53.45	79.53	80.81

Table 6. Dataset statistics after combining with generated data.

Datasets	No. Training Set	No. Training Set + Filtered Generated Pairs	No. Training Set + All Generated Pairs
Thai Wiki QA	9045	62,610 (+592.2%)	119,813 (+1224.6%)
iApp Wiki QA	5761	59,326 (+929.8%)	116,529 (+1922.7%)

5.1. Overall Results

The experiment was conducted based on two Thai QA datasets: Thai Wiki QA and iApp Wiki QA. We compared the results in three aspects: quality of the synthesized question-answer pairs, training strategies, and baseline models used in this experiment—WangchanBERTa and mT5-Base. Furthermore, we also evaluated the results in exact match (EM), word-level F1 (W-F1) and Syllable-level F1 (S-F1). The results are summarized in Table 4 for Thai Wiki QA and Table 5 for iApp Wiki QA. The training set statistics, including the generated dataset, are illustrated in Table 6.

In the result description, we created the combination name for readily referring to the tested model. The combination name consists of three parts: the QA models, training strategies, and augmented data used. The QA models have two possible types: WangchanBERTa (WBT) and mT5-Base (mT5). Training strategies have two possible strategies: Sequence (SEQ) and Merge (MRG). Lastly, the generated data used in the experiments have two types: Filtered (FLT) and ALL. All three parts connect together with the dash symbol (-). For example, mT5-SEQ-FLT refers to using the mT5 model fine-tuned with filtered generated data and the Sequence strategy.

We compared the results with the baseline models which are the question answering models that were trained with real human-labeled data only. In most cases, using generated data could improve the performance of every metric. In the Thai Wiki QA dataset, using the mT5-SEQ-FLT provided the best performance combination that beat the result of the baseline of mT5-Base by 6.28%, 5.14%, and 4.94% for EM, word-level F1, and syllable-level F1, respectively. In the iApp Wiki QA dataset, the best performance combination used the mT5-SEQ-ALL, which beat the baseline result of mT5-Base by 28.69%, 18.51%, and 19.44% EM, word-level F1, and syllable-level F1, respectively. Using all generated data with iApp Wiki QA could slightly improve performance compared to using the filtered generated data because the human-labeled training set of iApp Wiki QA has a smaller number of samples, compared to those of Thai Wiki QA, and it needs more data to fine-tune. However, the results between using the filtered generated pairs and all the generated pairs are not much different. We can conclude that using mT5-SEQ-FLT is the best combination that outperforms both datasets.

5.2. Comparison of the Training Strategies

The difference between the two training strategies is the fine-tuning steps. Sequence Strategy is sequentially fine-tuned on the generated data before the real data while Merge Strategy fine-tunes the combination of the generated data and the real data at the same time. As a result, in most combinations, using Sequence Strategy explicitly outperforms Merge Strategy, as shown in Tables 4 and 5 in the Sequence Strategy rows.

5.3. Comparison of Using Different Qualities of the Generated Question-Answer Pairs

In our trial, fine-tuning with the filtered generated pairs versus doing so with all generated pairs has no difference between the numbers of the outperforming cases. However, if we consider only the Sequence Strategy cases, the number of the outperforming cases of fine-tuning with the filtered generated pairs is greater than that of fine-tuning with all generated pairs. We can therefore conclude that using the filtered generated pairs is preferrable compared to using all generated pairs.

However, according to Table 5, which shows the result of the mT5-Base model with the Sequence Strategy, using all generated data (mT5-SEQ-ALL) is slightly better than using

the filtered generated data (mT5-SEQ-FLT), but the number of training samples is around twice as much, which consumes longer training time. In this case, we can summarize that using the filtered generated data is better than using all generated data in terms of the training duration.

5.4. Comparison of the Baseline Models

In the baseline experiments, the tables show that the results of outperforming models are different depending on the datasets. However, when we apply our method, as listed in Tables 4 and 5 in the columns "+ Filtered Generated Pairs" and "+ All Generated Pairs," compared to the column "Baseline," the mT5-Base model outperforms WanchanBERTa in all cases.

5.5. Comparison of the Syllable-Level F1

From Tables 4 and 5, the syllable-level F1 scores of all combinations are greater than the word-level F1 score. The mT5-SEQ-FLT in Thai Wiki QA has a syllable-level F1 score greater than the word-level F1 score by 1.65%, and the mT5-SEQ-ALL in iApp Wiki QA has a syllable-level F1 score greater than the word-level F1 score by 1.18%. The reason is that there are some predicted answers and/or ground truth answers that the word tokenizer used in the F1 score calculation does not perform correctly due to inability of tokenizer itself. This results in some overlapping words not counted to calculate the F1 score. However, the syllable tokenizer can tokenize those overlapping words into syllables and is able to calculate a nearer-correct F1 score. We provide some examples of the predicted answer and ground truth answer in Table 7, showing that the syllable F1 score gives a more accurate result than the word-level F1 score.

Table 7. Examples of the predicted answer and the ground truth answer that show the syllable-level F1; all results get 0 of the word-level F1.

Predicted Answer	Ground Truth Answer	Syllable-Level F1
มหาวิทยาลัยรามคำแหง (Ramkhamhaeng University)	*รามคำแหง* (Ramkhamhaeng)	66.67
คณะนิติศาสตร์ (Faculty of Law)	*นิติศาสตร์* (Law)	85.71
คมนาคม (Transportation)	*กระทรวงคมนาคม* (Ministry of Transportation)	75.00
ไม้ล้มลุกขนาดเล็ก (Small biennial plant)	*ล้มลุก* (Biennial)	57.14

6. Discussion

In this section we further analyze the improvement of the models in detail. First, we explain the analysis of model improvements by using the distribution of the word-level F1 score of both datasets. Secondly, we report the model performance by question types. Questions are classified by keyword extraction from Table 8. The results of each question type are illustrated in Tables 9 and 10. Lastly, we present the comparison of word tokenizers in terms of word-level F1 score calculation in Tables 11 and 12.

Table 8. Example of question word categories in Thai.

Who	What	Where	Year	Date	Number
ใคร (Who)	*อะไร* (What)	*ที่ไหน* (Where)	*ปีใด* (What year)	*วันที่เท่าใด* (Which date)	*เท่าไร* (How many/much)
คนใด (Which one)	*...ใด* (What/Which …)	*ที่ใด* (Where)	*ปีไหน* (What year)	*วันใด* (Which date)	*เท่าใด* (How many/much)
คนไหน (Which one)	*...ไหน* (What/Which …)	*ประเทศใด* (Which country)	*พ.ศ ใด* (What B.E. year)	*เมื่อใด* (When)	*กี่..* (How many/much of)
		จังหวัดใด (Which province)	*ค.ศ ใด* (What C.E. year)		

Table 9. Model performance evaluated on the Thai Wiki QA dataset classified by question types.

Question Type	mT5 Baseline			mT5-SEQ-FLT			% of Improvement		
	EM	W-F1	S-F1	EM	W-F1	S-F1	EM	W-F1	S-F1
Overall Performance	64.14	78.24	80.35	70.42	83.64	85.29	9.79	6.90	6.15
Who	69.16	82.03	81.40	73.05	85.87	85.30	5.62	4.68	4.79
What	62.90	77.45	80.48	68.82	82.89	85.24	9.41	7.02	5.91
Where	67.54	83.23	83.02	76.61	88.82	88.93	13.42	6.72	7.12
Year	76.04	83.06	85.03	83.83	88.75	89.53	10.24	6.85	5.29
Date	58.88	79.53	79.21	72.08	90.13	89.43	22.42	13.33	12.90
Number	61.99	74.14	76.67	68.19	78.94	81.32	10.00	6.47	6.07

Table 10. Model performance evaluated on the iApp Wiki QA dataset classified by question types.

Question Type	mT5 Baseline			mT5-SEQ-ALL			% of Improvement		
	EM	W-F1	S-F1	EM	W-F1	S-F1	EM	W-F1	S-F1
Overall Performance	29.36	63.46	63.71	58.05	81.97	83.15	97.72	29.17	30.51
Who	48.57	67.03	66.30	65.71	86.10	87.77	35.29	28.45	32.38
What	32.18	65.00	66.56	56.02	81.03	82.64	74.08	24.66	24.16
Where	33.33	68.22	71.85	55.56	79.72	85.17	66.69	16.85	18.54
Year	62.50	84.17	79.17	62.50	82.08	81.25	0.00	−2.48	2.63
Date	9.68	63.91	58.22	62.90	87.97	86.43	549.79	37.64	48.45
Number	13.83	55.49	53.70	60.64	80.90	80.76	338.47	45.79	50.39

Table 11. The result of Word-level F1 with different types of tokenizers of the Thai Wiki QA dataset.

Model	'Newmm' (Word-Level F1)	AttaCut (Word-Level F1)	Syllable Tokenizer (Syllable-Level F1)
WangchanBERTa Baseline	70.73	64.46	74.71
mT5 Baseline	78.24	73.58	80.35
mT5-SEQ-FLT	83.64	78.98	85.29

Table 12. The result of Word-level F1 with different types of tokenizers of the iApp Wiki QA dataset.

Model	'Newmm' (Word-Level F1)	AttaCut (Word-Level F1)	Syllable Tokenizer (Syllable-Level F1)
WangchanBERTa Baseline	69.68	65.42	71.81
mT5 Baseline	63.46	59.75	63.71
mT5-SEQ-ALL	81.97	78.87	83.15

6.1. Analysis of Model Improvement

To investigate what the improvement to the result is, we use the charts of the F1 score distribution to visualize how the score increases. We report only the result of the mT5-Base models because the results from Tables 4 and 5 show that the mT5-Base model outperforms the WangchanBERTa model. Based on the Thai Wiki QA dataset illustrated in Figure 10, the result of the baseline of the mT5-Base model (a) shows that there are more than 500 erroneous samples of F1 = 0, which means that those predicted answers are not overlapping with the ground truth answers. After using the combination mT5-SEQ-FLT (b), the result shows that the number of the perfect answers of F1 = 100 increases while the number of samples of F1 = 0 decreases. This is a proof that this method can increase the QA model performance.

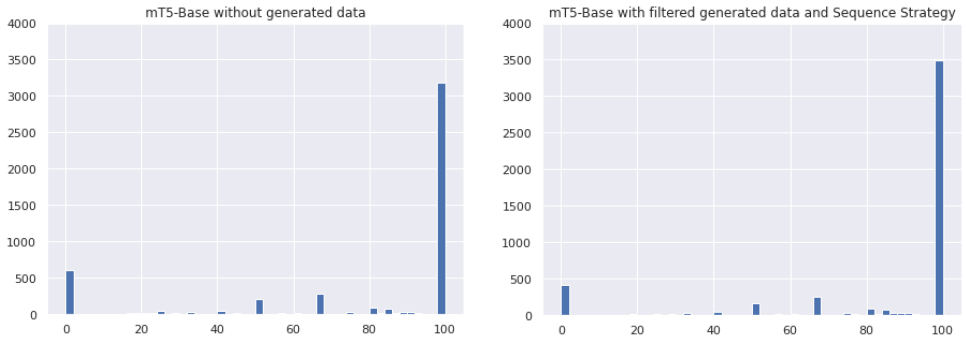

Figure 10. F1 distribution of the Thai Wiki QA dataset: (**a**) the result from the mT5-Base baseline model, and (**b**) the result from the combination mT5-SEQ-FLT of the Thai Wiki QA dataset.

The F1 distribution of the iApp Wiki QA dataset is illustrated in Figure 11. It represents a similar result to that of Thai Wiki QA; after applying the generated data and the Sequence strategy with mT5-Base model (mT5-SEQ-ALL), the number of perfect answers of F1 = 100 increases while the number of F1 = 0 cases decreases. This is also a proof that this method can increase the QA model performance even though the dataset is changed.

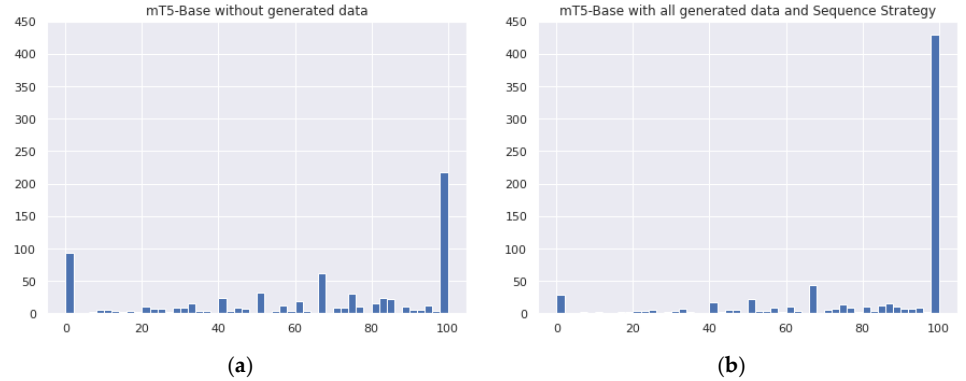

Figure 11. F1 distribution of the iApp Wiki QA dataset: (**a**) the result from the mT5-Base baseline model, and (**b**) the result from the combination mT5-SEQ-ALL of the iApp Wiki QA dataset.

6.2. Model Performance Analysis by Question Types

From the questions in the datasets, they can be classified into six groups based on the types of answers as follows.

- Who: a group of questions that requires an answer as a person name
- What: a group of questions that requires an answer as a thing or a name of things
- Where: a group of questions that requires an answer as a name of places, for instance, countries, provinces, or states
- Year: a group of questions that requires an answer as a year, either Common Era (C.E.) or Buddhist Era (B.E.)
- Date: a group of questions that requires an answer as a date
- Number: a group of questions that requires an answer as a number

We classified questions in a test set by keyword detection. The keywords that were used to categorize the questions are listed in Table 8. We next evaluated the model performance of each question type. The results from both datasets are listed in Tables 9 and 10. The results show that the best combination of both datasets can raise the performance above the baseline in most question types, except the question type 'Year' of iApp Wiki QA, which has the Word-level F1 score slightly lower than the baseline. After investigating, we found that there were only eight samples in the group 'Year' of iApp Wiki QA, which made this group of samples sensitive to the change of F1 score. However, the Syllable-level F1 score of this group improved after applying our method. This indicates that the best combination of iApp-Wiki-QA (mT5-SEQ-ALL) could predict more accurate answers, compared to the baseline model. This is further evidence that our method can increase the QA model performance.

6.3. Word Tokenizer Choices

In this work, we used 'newmm' for calculating the Word-level F1 score. However, there are several choices of Thai word tokenizers that can be used. We conducted experiments to compare two Thai word tokenizers; we selected AttaCut [21], a deep learning-based word tokenizer for Thai, to compare with 'newmm'. The results are shown in Tables 11 and 12.

From the results, the Word-level F1 scores from 'newmm' are higher than the Word-level F1 scores from AttaCut in all cases. This means that the AttaCut tokenizer has more tokenization mistakes on ambiguous words than 'newmm', which caused the drop of F1 scores. This proves that changing tokenizers may result in different Word-level F1 scores. In contrast, using Syllable-level F1 can address this problem by tokenizing a word into syllables, which is less ambiguous.

7. Conclusions

In this paper, we propose to employ transformer-based models for Thai QA, which aims to improve the performance of the Thai question answering model. The limitation of the low resource Thai QA corpora can be overcome by using a data generation composed of three steps: answer generation, question generation, and question filtration. To utilize the generated question-answer pairs, different fine-tuning strategies were investigated. Apart from the model improvement, all challenges in Thai were addressed in data preprocessing. We also propose a new evaluation metric at the syllable-level, which is more suitable for the Thai language because there is no ambiguity in syllable tokenization. We conducted experiments on two Thai question answering datasets, the Thai Wiki QA and the iApp Wiki QA. The results showed that the generated data can explicitly enhance the model performance from 78.24 to 83.64 in Thai Wiki QA and 63.46 to 81.97 in iApp Wiki QA, in terms of the Word-level F1 score.

However, a limitation of our work is that our data generation technique is appropriate with a span-extraction question answering task; the answer of the given question is part of the given context only.

Author Contributions: Conceptualization, P.P.; methodology, P.P.; software, P.P.; validation, P.V.; data curation, P.P.; writing—original draft preparation, P.P.; writing—review and editing, P.V. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Raffel, C.; Shazeer, N.; Roberts, A.; Lee, K.; Narang, S.; Matena, M.; Zhou, Y.; Li, W.; Liu, P.J. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv* **2019**, arXiv:1910.10683.
2. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, Ł.; Polosukhin, I. Attention is all you need. In Proceedings of the Advances in Neural Information Processing Systems, Long Beach, CA, USA, 4–9 December 2017; pp. 5998–6008.
3. Dodge, J.; Sap, M.; Marasovic, A.; Agnew, W.; Ilharco, G.; Groeneveld, D.; Gardner, M. Documenting the english colossal clean crawled corpus. *arXiv* **2021**, arXiv:2104.08758.
4. Rajpurkar, P.; Zhang, J.; Lopyrev, K.; Liang, P. Squad: 100,000+ questions for machine comprehension of text. *arXiv* **2016**, arXiv:1606.05250.
5. Decha, H.; Patanukhom, K. Development of thai question answering system. In Proceedings of the 3rd International Conference on Communication and Information Processing, Tokyo, Japan, 24–26 November 2017; pp. 124–128.
6. Seo, M.; Kembhavi, A.; Farhadi, A.; Hajishirzi, H. Bidirectional attention flow for machine comprehension. *arXiv* **2016**, arXiv:1611.01603.
7. Lapchaicharoenkit, T.; Vateekul, P. Machine Reading Comprehension on Multiclass Questions Using Bidirectional Attention Flow Models with Contextual Embeddings and Transfer Learning in Thai Corpus. In Proceedings of the 8th International Conference on Computer and Communications Management, Singapore, 17–19 July 2020; pp. 3–8.
8. Trakultaweekoon, K.; Thaiprayoon, S.; Palingoon, P.; Rugchatjaroen, A. The first wikipedia questions and factoid answers corpus in the thai language. In Proceedings of the 2019 14th International Joint Symposium on Artificial Intelligence and Natural Language Processing (iSAI-NLP), Chiang Mai, Thailand, 7–9 November 2019; pp. 1–4.
9. Puri, R.; Spring, R.; Patwary, M.; Shoeybi, M.; Catanzaro, B. Training question answering models from synthetic data. *arXiv* **2020**, arXiv:2002.09599.
10. Lowphansirikul, L.; Polpanumas, C.; Jantrakulchai, N.; Nutanong, S. WangchanBERTa: Pretraining transformer-based Thai Language Models. *arXiv* **2021**, arXiv:2101.09635.
11. Xue, L.; Constant, N.; Roberts, A.; Kale, M.; Al-Rfou, R.; Siddhant, A.; Barua, A.; Raffel, C. mt5: A massively multilingual pre-trained text-to-text transformer. *arXiv* **2020**, arXiv:2010.11934.
12. Hochreiter, S.; Schmidhuber, J. Long short-term memory. *Neural Comput.* **1997**, *9*, 1735–1780. [CrossRef] [PubMed]
13. Devlin, J.; Chang, M.-W.; Lee, K.; Toutanova, K. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv* **2018**, arXiv:1810.04805.
14. Joshi, M.; Chen, D.; Liu, Y.; Weld, D.S.; Zettlemoyer, L.; Levy, O. Spanbert: Improving pre-training by representing and predicting spans. *Trans. Assoc. Comput. Linguist.* **2020**, *8*, 64–77. [CrossRef]
15. Dhingra, B.; Pruthi, D.; Rajagopal, D. Simple and effective semi-supervised question answering. *arXiv* **2018**, arXiv:1804.00720.
16. Radford, A.; Wu, J.; Child, R.; Luan, D.; Amodei, D.; Sutskever, I. Language models are unsupervised multitask learners. *OpenAI Blog* **2019**, *1*, 9.
17. Liu, Y.; Ott, M.; Goyal, N.; Du, J.; Joshi, M.; Chen, D.; Levy, O.; Lewis, M.; Zettlemoyer, L.; Stoyanov, V. Roberta: A robustly optimized bert pretraining approach. *arXiv* **2019**, arXiv:1907.11692.
18. Phatthiyaphaibun, W.; Suriyawongkul, A.; Chormai, P.; Lowphansirikul, L.; Siwatammarat, P.; Tanruangporn, P.P.; Charoenchainetr, P.; Udomcharoenchaikit, C.; Janthong, A.; Chaovavanichet, K.; et al. PyThaiNLP/pythainlp: PyThaiNLP v2.3.2 Release! *Zenodo* **2021**. [CrossRef]
19. Zeng, C.; Li, S.; Li, Q.; Hu, J.; Hu, J. A Survey on Machine Reading Comprehension—Tasks, Evaluation Metrics and Benchmark Datasets. *Appl. Sci.* **2020**, *10*, 7640. [CrossRef]
20. Wolf, T.; Chaumond, J.; Debut, L.; Sanh, V.; Delangue, C.; Moi, A.; Cistac, P.; Funtowicz, M.; Davison, J.; Shleifer, S. Transformers: State-of-the-art natural language processing. In Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations, Online, 16–20 November 2020; pp. 38–45.
21. Chormai, P.; Prasertsom, P.; Rutherford, A. AttaCut: A Fast and Accurate Neural Thai Word Segmenter. *arXiv* **2019**, arXiv:1911.07056.