BERT+vnKG: Using Deep Learning and Knowledge Graph to Improve Vietnamese Question Answering System

Truong H. V Phan[1]
Faculty of Information Technology
Van Lang University
Ho Chi Minh City, Vietnam

Phuc Do[2]
University of Information Technology
Vietnam National University
Ho Chi Minh City, Vietnam

***Abstract*—A question answering (QA) system based on natural language processing and deep learning is a prominent area and is being researched widely. The Long Short-Term Memory (LSTM) model that is a variety of Recurrent Neural Network (RNN) used to be popular in machine translation, and question answering system. However, that model still has certainly limited capabilities, so a new model named Bidirectional Encoder Representation from Transformer (BERT) emerged to solve these restrictions. BERT has more advanced features than LSTM and shows state-of-the-art results in many tasks, especially in multilingual question answering system over the past few years. Nevertheless, we tried applying multilingual BERT model for a Vietnamese QA system and found that BERT model still has certainly limitation in term of time and precision to return a Vietnamese answer. The purpose of this study is to propose a method that solved above restriction of multilingual BERT and applied for question answering system about tourism in Vietnam. Our method combined BERT and knowledge graph to enhance accurately and find quickly for an answer. We experimented our crafted QA data about Vietnam tourism on three models such as LSTM, BERT fine-tuned multilingual for QA (BERT for QA), and BERT+vnKG. As a result, our model outperformed two previous models in terms of accuracy and time. This research can also be applied to other fields such as finance, e-commerce, and so on.**

Keywords*—Bidirectional Encoder Representation from Transformer (BERT); knowledge graph; Question Answering (QA); Long Short-Term Memory (LSTM); deep learning; Vietnamese tourism; natural language processing

I. Introduction

Question answering system using deep learning is a challenging field and has received a lot of attention in recent years. Question answering system is also highly applied in practice. Answering questions automatically helps investors make market price decisions, users can order products anytime, visitors can ask about tourism places, etc. The question answering system accepts a natural language input question and the returned result is a natural language answer in a specific field. Many models have been applied for this system such as LSTM, knowledge graph and BERT, which are surveyed in this section. Besides that, we also used combined model to improve the efficiency of the Vietnamese question answering system that can be applied in Vietnam tourism.

Question answering system using the LSTM model has been extensively studied because LSTM can predict next word due to permanently store the status of words based on context of previous sentences [1] [2]. Di Wang et al. proposed a method that combined Bidirectional Long-Short Term Memory (BiLSTM) and keywords matching to choose an answer sentence without using parsing syntax or any additional resources [3]. This model loaded words from questions and answers and returned their appropriate score for every answer. Firstly, the method put a tag <S> at the end of every question to discriminate answers. Each word of an input question was encoded into vector by using word2vec. Afterwards, BiLSTM read both of them according to two directions that kept previous and future context of words in questions and answers. These contexts were stored in cell memory vectors combined to generate hidden vectors. The final hidden vectors were labeled to determine whether the correct answer for input question. In addition, their method associated with keywords matching to indicate the correct answer and to avoid proper nouns not existed in vocabulary of word embedding.

G. Rohit used LSTM and memory network to build question answering system for professional fields [4]. For the LSTM model, the authors converted all exchanged content in a dataset by using word2vec from Google. The question was then added to each text that contained the answer as input for LSTM. Each word in the text was predicted according to the difference between the question and the text. This discrepancy was sent back to LSTM to reprocess until the prediction matched the question. For memory network, the authors' system had four components: (1) input featured map converted questions into vectors, (2) generalization compressed and generalized the old memory for the next step, (3) output featured map generated a new output from the new input and the current memory state, (4) the response converted the result from (3) into an appropriate readable format.

Besides LSTM model, knowledge graph is also applied to build question answering system instead of natural language understanding, recommender systems [5] [6] [7] [8]. Ben Hix et al. established the KNOWBOT system as a Q&A system about science and collected relationships between concepts in each question [9]. They collected 107 scientific questions from the 4th York New York Regents test, where each question had 4 answers that would be converted into question-answer pairs and labeled True for the correct answer and False for the wrong answer. KNOWBOT built graphs from dialog and utterance. To build utterance graphs, the system converted a user's sentence into a fully conceptual relationship after removing stop words in the sentence. To construct a graph from a dialog, the system created edges taken from utterance and calculated score for the answers in relation set. This set included questions and supporting sentences obtained from text corpus that appeared in the dialog graph. From the above calculated score, the system easily found correct answers for questions.

Xiao Huang et al. presented knowledge embedding based on question answering to look up a triple likes (head, predicate, tail) of input question in knowledge graph embedding and found relevant facts as correct answers [10]. Their model included three stages: (1) from input question, the model took a question as input parameter and trained predicate learning model to obtain a vector representing predicate. Then, head entity learning model was also trained to get head entity representing vector of input question. (2) Head Entity Detection model was executed to filter appropriate candidate entities for declining searching space. (3) The model used a function to calculate a score for each candidate tail entity as an answer.

As mentioned above, LSTM model had some certain drawbacks, so BERT (Bidirectional Encoder Representations from Transformers) has emerged as a model that is more outstanding than LSTM. Jacob Devlin et al. invented BERT to improve drawbacks of previous models that processed according to one direction [11]. BERT uses a masked language model (LM) to randomly tag tokens with masks in the input sentence, then the model predicts appropriate vocabulary for the labels based on the context of the sentence. The masked language model allows bidirectional contextual training from left to right and vice versa. The model has two steps: pre-training and fine-tuning. For the pre-training step, BERT uses two Masked LM tasks and Next Sentence Predict. In the former task, Masked LM, the model randomly selects 15% of the tokens in the sentence. For each chosen *i-th* word, it will be replaced by [MASK] token, then BERT will predict the masked word in the original dictionary for that token with cross entropy loss function. In the latter task, Next Sentence Predict, the model is applied to understand the relationship between sentences using binary next sentence prediction task from any corpus. For example, model selects two sentences S_1 and S_2 to train, if S_2 appears 50% after S_1, it will be labeled IsNext; if S_2 is a random sentence in the corpus, it will be assigned NotNext. In the latter task, fine-tuning step, BERT will encode text before using the self-attention mechanism to predict words in sentences. BERT is also applied in the question answering system that takes input parameter that is a question-context pair and the output result will process token representing vector to give the answer [12]. BERT is designed to support many natural language processing tasks such as sentence classification, tokenization, emotional classification, and so on [13] [14] [15].

Aiting Liu et al. proposed BERT-CRF model to find entities in question and BERT-softmax to solve the entity disambiguation problem [16]. From there, they proposed the BB-KBQA (BERT-based Knowledge Base Question Answering) model that combined BERT with the knowledge base to represent the semantics of questions, entities and predicates. Their system consisted of three components such as entity linking, predicate linking and answer selection. The entity linking component used BERT-CRF to perform the question mentioned detection function to detect entities in question based on the input sequence and the final state vector passed to the CRF layer to predict labels for each token. In addition, entity linking also performed disambiguation entities based on the input sequence and final states. For each final state, the softmax function calculated probability of each labeled candidate entity which appeared in the sentence. The predicate mapping component was applied for predicate set in knowledge base that was obtained from the entities of the entity linking component. Candidate predicates were scored based on semantic similarity with the question. Finally, answer selection calculated total weight of the candidate entities with the candidate predicate and selected the highest-weight entity-predicate pair as the answer.

Dongfang Li et al. used BERT to select the answer [17]. They choose a pair of candidate answers and train them to choose the one that best fit the question. The input for the encoding step of the model was a triple (Q, POS, NEG) where Q was a given question, POS was a positive answer, and NEG was a negative answer. Then, they created two pairs of Q-POS and Q-NEG as fine-tuning input step of BERT to receive embedding [CLS]. The following layer was fully connected layer that calculated the score for each pair through the sigmoid function.

Most of the above models showed certain effectiveness for choosing answers to input questions. In particular, the BERT model has yielded state-of-the-art results, so it is applicable to many different languages [18] [19]. There also were some studies that combined BERT and knowledge graph. Liang Yao proposed BERT for knowledge graph completion (KG-BERT) that used BERT to classify triplets, predict links and predict relations in a knowledge graph [20]. Weijie Liu proposed a knowledge-enabled language representation (K-BERT) that used knowledge graph to embed into input sentence as a knowledge expert [21]. This aimed to clearly explain what head and tail entities of a triplet were. However, few studies have used BERT and knowledge graphs for question answering system. In this paper, we proposed a model named BERT+vnKG that was a combination of BERT and knowledge graph applied for the Vietnamese question answering system in tourism. Our model's results were more outstanding than two previous models such as BERT for multilingual QA and LSTM about accuracy and speed for an answer. Our contributions were as follows:

- Developing a knowledge graph in Vietnam tourism from Vietnamese non-textual materials. Knowledge graph used to narrow context space based on entities of input questions.
- Using BERT model to find answers accurately and fast for Vietnamese question answering system based on knowledge graph that was narrowed and converted into text.

- Test on hand-made Vietnamese question answering dataset and compare the effectiveness on LSTM, BERT, and BERT+vnKG

We presented components to establish a question answering system using BERT+vnKG in Section 2. In Section 3, we experimented the performance of LSTM, BERT and BERT+vnKG on Vietnam tourism crafted dataset. Finally, discussions and suggestions for future research of the system will be covered in Section 4.

II. Methodology

A. System Overview

This section presents an overview of the Q&A system for tourism in Vietnam and describes in detail each component. Fig. 1 shows an overview of the BERT+vnKG system that answered questions about visiting places in Vietnam. Firstly, the system accepts a natural language question. Secondly, the Extract Subject component is responsible for drawing out entities in the question. Then, these entities become the input parameter for the Subgraph component that is responsible for extracting a subgraph obtained from knowledge graph. This subgraph contains the extracted entities. If a subgraph exists, the subgraph is converted to context. Otherwise, the dataset become context. The contextual content and the question become the input for BERT to produce the final answer.

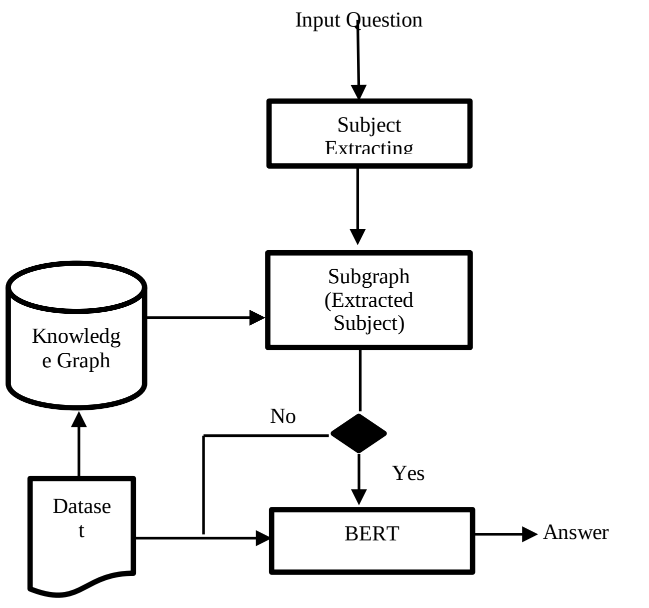

Fig 1. A Vietnamese Tourism Question Answering System using BERT.

B. Dataset

We create manual data by browsing contents of travel websites in Vietnam. We then generate a fact consisting of a pair of question and answer, which are separated by a label or symbol. For example, “Which bridge does Da Nang have? <BOA>Da Nang has Golden Bridge” is a fact, where <BOA> means “Begin of Answer” which is the label separating two sentences. We collect 300-paired sentences that included questions and answers about Vietnam tourist places and create more than 4600 relationships between entities from these sentence. This dataset is both to create the knowledge graph and to become context for BERT to find answers.

C. Knowledge Graph Module

A knowledge graph (KG) consists of a set of entities $E = \{e_1, e_2, ..., en\}$ that are nodes representing famous places and dishes in Vietnam and a set of relation R = {r1, r2, ..., rn} are predicates that link between two entities. For example, the sentence “Da Nang has Golden Bridge” is presented by knowledge graph as $Da\ Nang \xrightarrow{has} Gold\ Bridge$. Fig. 2 illustrated how we constructed the knowledge graph from the original corpus D.

From the travel data set, we use underthesea and VnCoreNLP library, which are open source natural language processing toolkits, to analyze sentences [22] [23]. For each sentence, we separate word tokens to draw a triple (e1, r, e2), where e1 and e2 are entities that appear in the sentence, r is a predicate of the sentence. From the triples, we create relationships and add them to the knowledge graph. This process is described in algorithms 1 and 2.

The pos_tag (s) function of the underthesea library is applied to assign label to each word in the sentence. A set E contains nouns or noun phrases in the sentence and a set V contains verbs in the sentence. For example, the sentence "Da Nang has Golden Bridge" was labeled as {("Da Nang", "Np"), ("has", "V"), ("Gold", "N"), ("Bridge" "," N "), E = {" Da Nang "," Gold Bridge "}, V = {" has "}. Finally, we create a relationship for the triple (e_i, v_i, e_{i+1}). Besides that, we also experienced another library named VnCoreNLP [23]. VnCoreNLP is also an open source that can be downloaded from [23]. The difference with underthesea is that VnCoreNLP has dependency parsing feature that can analyze relationships between parts of speech in a Vietnamese sentence. For example, a sentence “Da Nang has Gold Bridge” was parsed as [(“Da Nang”, “sub”, 2, 1), (“has”, “root”, 0, 2), (“Gold”, “nmod”, 4, 3), (“Bridge”, “dob”, 3, 4), (“.”, “punct”, 2, 5)]. From that, we have a triple t = (head, predicate, tail) as (“Da Nang”, “has”, “Gold Bridge”). But, we only create one-directional relationship for the triple because of semantic sentence. Consequence, the result of triple extracting using VnCoreNLP was more correct and semantic than the result using underthesea library. Therefore, we choose VnCoreNLP library as a major component of our system.

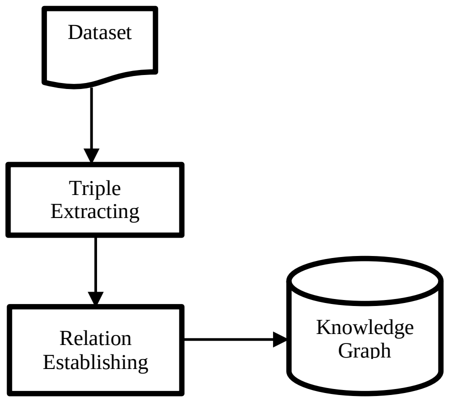

Fig 2. The Process Converted Each Sentence of Dataset into Knowledge Graph.

Algorithm 1 Pseudo code for extracting a triple of a sentence

```
Input    A sentence s
Output   A triple T = ($e_1$, r, $e_2$) represents for a sentence
 1:  Function ExtractTriple(s)
 2:  parse_sentence = Call $vncorenlp.dep\_parse(s)$ to execute dependency parsing for sentence s
 3:  for $i \in parsen\_tence$ do
 4:      if $i$ = "sub" then
 5:          $h \leftarrow h \cup i$
 6:      Endif
 7:  Endfor
 8:  for $i \in parse\_sentence$ do
 9:      if $i$ = "root" then
10:          $p \leftarrow p \cup i$
11:          if $i \notin \{sub, punct, dob, pob\}$ then
12:              $p = p \cup i$
13:          Endif
14:      Endif
15:  Endfor
16:  for $i \in parse\_sentence$ do
17:      if $i \notin \{sub, root, punct\}$ then
18:          $t = t \cup i$
19:      Endif
20:  Endfor
21:  $T = \emptyset$
22:  if each $t_i \in t$ then
23:      $T \leftarrow T \cup (h, p, t_i)$
24:  endif
25:  Return $T$
26:  End Function
```

Algorithm 2 creates a knowledge graph with more than 4600 triples describing the relationships between the two entities in the sentence. We save this graph to a text file for supporting BERT model to find the answers.

Algorithm 2 Pseudo code for building a knowledge graph

```
Input:    Vietnamese tourism dataset D
Output:   Knowledge graph G
 1:  Function BuildKnowledgeGraph(D)
 2:  $G \leftarrow \emptyset$
 3:  for $d \in D$ do
 4:      K ← ExtractTriple(d)
 5:      for $k \in K$ do
 6:          $G \leftarrow G \cup k$
 7:      endfor
 8:  endfor
 9:  Return $G$
10:  End Function
```

D. Subject Extracting Module

This component extracts places or dishes that were mentioned in the question. We continued using the natural language processing functions of VnCoreNLP libraries to extract entities in question. We run dependency parsing function on the question to obtain subjects and predicates. For example, the sentence "Where is King Garden" is analyzed as R={"is"}, E = {"King Garden"}. This process is showed in algorithm 3.

Algorithm 3 Pseudo code for extracting entities from a question

```
Input:    A question q
Output:   Entity set E = {$e_1$, $e_2$, …} of question q
 1:  Function ExtractEntityFromQuestion (q)
 2:  parse_sentence = Call $vncorenlp.dep\_parse(q)$ to run dependency parsing on question q
 3:  for $i \in parse\_sentence$ do
 4:      if i = "sub" then
 5:          $E \leftarrow E \cup i$
 6:      endif
 7:  endfor
 8:  for $i \in parse\_sentence$ do
 9:      if $i \notin \{sub, root, punct\}$ then
10:          $E \leftarrow E \cup i$
11:      endif
12:  endfor
13:  Return $E$
     End Function
14:
```

E. Subgraph for Extracted Subjects Module

This component extracts subgraphs for the entities taken from input question. For each entity, we search the knowledge graph created from the dataset to collect entities related to entities in question. For example, "King Garden located in Thanh Thuy district, Phu Tho province" is analyzed for entity E = {"King Garden", "Thanh Thuy district", "Phu Tho province"} and episode R = {"located in"}. With the question "Where is King Garden?" We have the subgraphs related to the entity "King Garden" as follows:

$$King\ Garden \xrightarrow{located\ in} Thanh\ Thuy\ district$$

$$King\ Garden \xrightarrow{located\ in} Phu\ Tho\ province$$

Fig. 3 describes the process that extracted subgraphs for each entity in the question. Firstly, we use Subject Extracting module to obtain entities. Then, we apply Subgraph for Extracted Subjects module associated with the knowledge graph to collect subgraph of each entity. In Fig. 4, we show an example to demonstrate that process.

Algorithm 4 Pseudo code for finding subgraphs of entities

```
Input:   Knowledge graph G, entity set $E_q$ in question q
Output:  A set of subgraph SG for $E_q$
1:   Function FindSubgraphForEntity (G, $E_q$)
2:   for $e_i \in E_q$ do
3:       for triple t = ($h_m$ ,r, $t_n$) $\in G$ do
4:           if $e_i \in t$ then
5:               $SG \leftarrow SG \cup t$
6:           endif
7:       endfor
8:   endfor
9:   Return $SG$
10:  End Function
```

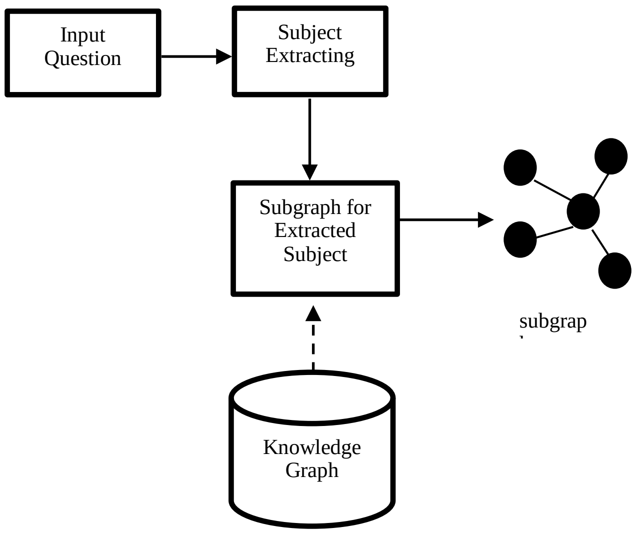

Fig 3. Illustration of Process that Extracted Subgraph from Knowledge Graph.

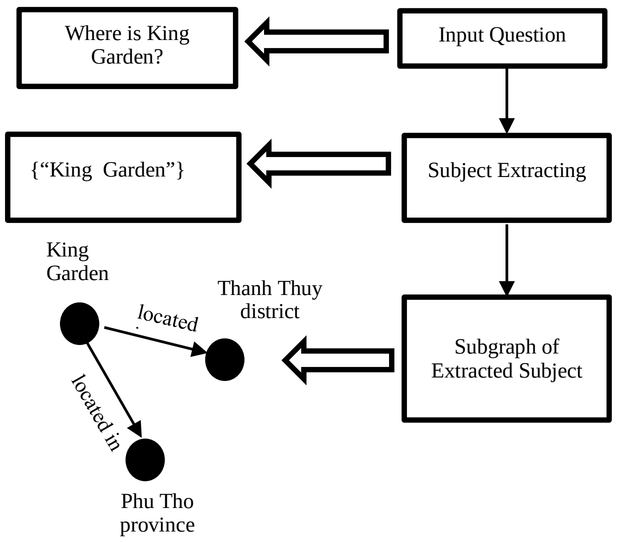

Fig 4. An Example about Extracting a Subgraph for Sentence "King Garden Located in Thanh Thuy District, Phu Tho Province"" was applied by Subgraph for Extracted Subject Module.

F. BERT Module

This component uses the BERT model to predict the answer. BERT is a masked language model with two main functions: pre-training and fine-tuning. BERT has many varieties such as BERT-Tiny, BERT-Mini, BERT-Small, BERT-Medium and BERT-Large. BERT-Base currently supports 104 languages including Vietnamese. BERT is used in many different applications, especially in the question answering system [24] [25] [26]. We use the fine-tuned BERT model for multilingual Q&A created by Manuel Romero because this model has dataset that is compatible with original dataset SQuAD v1.1 but also extends more eleven languages including Vietnamese language [27]. The input question, subgraph of entities relate to the question and dataset are fed as input parameters to BERT model to generate the answer. If the subgraph does not exist, BERT loads the question and original dataset to find the answer. In this case, time to respond an answer is very slow because BERT has to load entire content into memory. Otherwise, if the subgraph for entities existed, we converted the subgraph into textual context, then BERT used this context to find the answer. As a result, BERT determines the answer quickly because the value range for the candidate answers were narrowed.

Algorithm 5 Pseudo code for converting subgraph into context

```
Input:   A subgraph SG
Output:  Context c
1:   Function SubgraphToContext (SG)
2:   for each triple $s \in SG$ do
3:       sentence ← $s.h \cup s.r \cup s.t$
4:       c ← c ∪ sentence
5:   endfor
6:   Return $c$
7:   End Function
```

Algorithm 5 shows how to convert subgraphs to text as one of input parameters for BERT model. For each triple (h, r, t) in the subgraph, we join each component into sentences.

Algorithm 6 FindAnswerWithBERT (q, SG, D)

```
Input:   Question q, subgraph SG, dataset D
Output:  Answer a
1:   Function FindAnswerWithBERT (q, SG, D)
2:   if $SG \neq \emptyset$ then
3:       c ← SubgraphToContext(SG)
4:       a ← BERT(q, c)
5:   else
6:       c ← D
7:       a ← BERT(q, c)
8:   endif
9:   Return $a$
10:  End Function
```

Algorithm 6 shows how BERT predicts an answer. The best case happens when each entity in the question is able to derive subgraph from the knowledge graph to limit search space. The worst case occurs when entities in the question are not exist any subgraphs respectively, then BERT loads entire dataset to find the answer. Consequently, time to return an answer is very slow.

III. Experiment

A. Training and Testing Dataset

We collect over 300 questions and answers about famous places and dishes in Vietnam. Afterwards, we create three files as follows: facts.txt, questions.txt and answers.txt. The facts.txt file contains 300 question-answer pairs in the following format: *"question<BOS>answer"*. For example, we have a fact likes *"Where is King Garden? <BOS> King Garden is located in Thanh Thuy district, Phu Tho province"*. The *questions.txt* file contains only 300 questions that are separated from facts.txt file, the answers.txt file contains only 300 answers that are separated from facts.txt. A knowledgegraph.txt file contains 300 triples extracted from answers.txt to establish a knowledge graph. We use 80% of data in each file for training and 20% of data to evaluate the learning of models.

B. Metrics and Comparisions

We use the LSTM model with batch size = 32, latent dim = 256, epochs = 1000 parameters to experiment on facts.txt file. We then run BERT fine-tuned for QA model on the answers.txt file. Finally, we test our proposed model BERT+vnKG on answers.txt file. Three models are tested on computer with Intel Core i5-6200U CPU configuration 2.30 GHz 2.40 GHz, 16 GB RAM.

For LSTM model, each learning step loads all facts n = 300 pairs of questions and answers. This process repeats 1000 epochs, so the complexity is O(n^2). For BERT multilingual for QA, for each question in the questions.txt file, the model loads entire answers n = 300 from the answers.txt file as a context to find answers, so the complexity is O(n^2). We observe the run time of BERT for QA and calculate the following results: every 50 questions took three hours; the total time that BERT for QA answers 300 questions is T(n) = 300 * 3/50 = 18 hours. For BERT+vnKG model, because each question has subgraphs for its entities, so the domain for finding answers is narrowed. As a result, the complexity of this model is O(nlogn).

TABLE I. A Comparison about Time and Complexity among three Models as LSTM, BERT for multilingual QA and BERT+vnKG

Model	Dataset QA sentences	Approximate Time	Complexity
LSTM	300	4 hours	O(n^2)
BERT for multilingual QA	300	18 hours	O(n^2)
BERT+vnKG	**300**	**3 hours**	**O(nlogn)**

Table I shows the results that are experimented on 300 QA sentences and we observe the time that finishes answering prediction of three models. As a result, our model achieves more effectively than both LSTM and BERT for multilingual QA in term of time and complexity.

We use F1 to evaluate the accuracy of the three models [28]. Call TP as the sentences with real classes and the forecast is true positive, FP is the sentences with really positive class but the forecast is negative, TN is the sentences with real class but negative is forecast, positive and FN is Sentences with real class and forecast are negative. From here, we calculate the coefficient of F1 with the following formula

$$Precision = \frac{TP}{TP+FP} \quad (1)$$

$$Recall = \frac{TP}{TP+FN} \quad (2)$$

$$F1 = 2 \times \frac{Precision*Recall}{Precision+Recall} \quad (3)$$

Besides that, we also use Matthews Correlation Coefficient (MCC) to measure the quality classification of three models because it uses additional true negative part that F1 do not use [29]. The formula for calculating MCC is as follows:

$$MCC = \frac{TP*TN-FP*FN}{\sqrt{(TP+FP)(TP+FN)(TN+FP)(TN+FN)}} \quad (4)$$

Table II shows the predicted results of three models. We split our dataset into two parts such as 80% dataset for training and 20% dataset for testing. Besides that, we name correctly predicted sentences as positive and incorrectly predicted sentences as negative. These support to calculate F-measure and MCC score later.

TABLE II. A Comparison of Sentence Predicting among three Models: LSTM, BERT for Multilingual QA and BERT+vnKG

Model	Trained Data (80%) 240 QA sentences		Test Data (20%) 60 QA sentences	
	Positive	**Negative**	**Positive**	**Negative**
LSTM	202	38	52	8
BERT for multilingual QA	72	168	18	42
BERT+vnKG	**217**	**23**	**55**	**5**

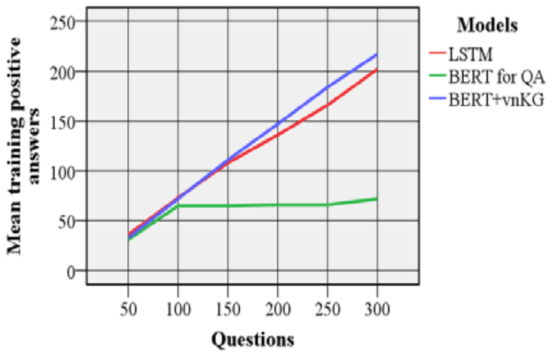

Fig 5. A Comparison of Three Models like LSTM, BERT for QA and BERT+vnKG about Predicting Positive Answers.

In Fig. 5, we evaluate the correct predicting based on means of answers that are classified into positive class. The values of vertical axis show positive answers' means of three models and the values of horizontal axis show questions that are tested. From results of Fig. 5, BERT+vnKG outperforms LSTM and BERT for multilingual QA.

Table III shows values of true positive (TP), false positive (FP), false negative (FN) that join above equations to calculate precise (P), recall (R) and F1 scores. Consequently, F-measure score of BERT+vnKG is higher than LSTM and BERT for multilingual QA.

TABLE III. A COMPARISON OF F1-SCORE ABOUT ACCURATE CLASSIFICATION AMONG THREE MODELS AS LSTM, BERT FOR MULTILINGUAL QA AND BERT+VNKG

Model	**TP**	**FP**	**FN**	**P**	**R**	**F1**
LSTM	202	38	8	0.84	0.96	0.90
BERT for multilingual QA	72	168	42	0.30	0.75	0.43
BERT+vnKG	**217**	**23**	**5**	**0.9**	**0.97**	**0.94**

In Fig. 6, we use F1 scores to compare classification of the models. The values in vertical axis show the F1 score's means of the models and the values in horizontal axis are the number of questions that are tested. According to this score, our proposed model originally predicts less correctly than LSTM at first 50 sentences, but the more sentences experiment, BERT+vnKG gives better result than LSTM and BERT for QA.

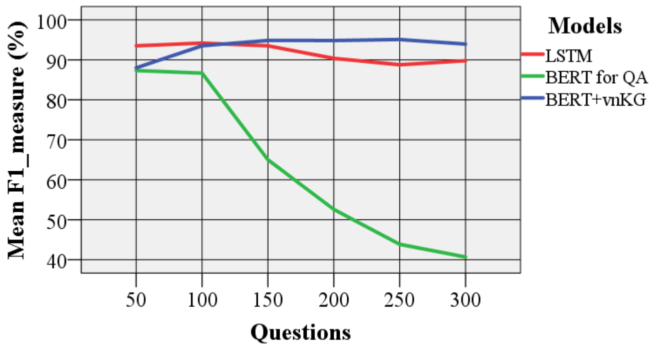

Fig 6. A Comparison of F1-Score about Binary Classification among LSTM, BERT for QA and BERT+vnKG.

TABLE IV. THE RESULT OF COMPARISONS ABOUT MCC SCORES FOR MEASURING THE QUALITY OF QUESTION ANSWERING CLASSIFICATIONS BASED ON THREE MODELS LIKE LSTM, BERT FOR MULTILINGUAL QA AND BERT+VNKG

Model	**TP**	**FP**	**FN**	**TN**	**MCC**
LSTM	202	38	8	46	0.59
BERT for multilingual QA	72	168	23	210	0.25
BERT+vnKG	**217**	**23**	**5**	**28**	**0.63**

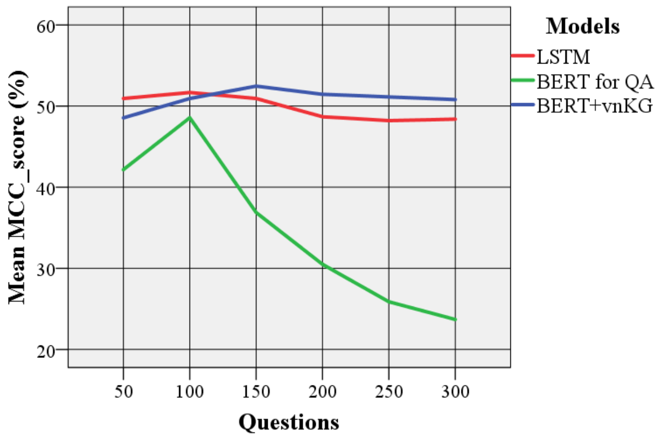

Fig 7. A Comparison of MCC Scores about Binary Classification among LSTM, BERT for QA and BERT+vnKG.

Table IV shows values of true positive (TP), false positive (FP), false negative (FN) and true negative (TN) that use to calculate MCC scores of three models. As a result, the MCC score of BERT+vnKG is higher than two remaining models.

In Fig. 7, we use MCC scores to evaluate correct classification of three models. This score is calculated from four variables such as True Positive, False Positive, False Negative and True Negative; meanwhile, F1-score only concentrate on first three variables. The values in vertical axis show the MCC score's means of the models and the values in horizontal axis are the number of questions that are experimented.

From the above tables, the coefficient F1=0.94 and MCC=0.63 show that our proposed model returns answers better than the other two models like LSTM and BERT.

IV. CONCLUSION

BERT model has brought about effectively state-of-the-art results in natural language processing, especially in machine reading comprehension. Moreover, BERT supports multiple languages including Vietnamese, so we have used this model to build a QA system about Vietnam tourism. However, using only BERT model does not return high accuracy for Vietnamese answers. Therefore, we have combined BERT and knowledge graph to improve the precision of Vietnamese question answering system. Using knowledge graph has limited search space for BERT to find answers. As a result, our system achieved both high accuracy and time improvement through experimenting on three models such as LSTM, BERT and BERT+vnKG with dataset which collected visiting places and special dishes in Vietnam.

V. FUTURE WORK

Although our suggested model outperformed, the system also had some limitations such as our model returned incomplete answers because BERT used attention mechanism that only output key phrases in an answer, the dataset is relatively small with 300 pairs of questions and answers that did not prove the true power of BERT+vnKG model, and the time to test the models was quite long due to evaluating on computer that had low resources. Future work will experiment on various datasets having a certain benchmark and large scale. Plus, we will evaluate our system by using rich-resource services such as GPUs or TPUs to improve the processing speed for big data.

Acknowledgment

This research is funded by Vietnam National University Ho Chi Minh City (VNU-HCMC) under the grant number DS2020-26-01

References

[1] Marco Antonio Calijorne Soares, Wladmir Cardoso Brandao, Fernando Silva Parreiras, "A Neural Question Answering System for Supporting Software Engineering Students," in 2018 XIII Latin American Conference on Learning Technologies (LACLO), 2018.

[2] Yashvardhan Sharma, Sahil Gupta, "Deep Learning Approaches for Question Answering System," in International Conference on Computational Intelligence and Data Science (ICCIDS 2018), 785-794, 2018.

[3] Di Wang, Eric Nyberg, "A Long Short-Term Memory Model for Answer Sentence Selection in Question Answering," in Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics, Beijing, China, 2015.

[4] G. Rohit, Ekta Gautam Dharamshi, Natarajan Subramanyam, "Approaches to Question Answering Using LSTM and Memory Networks," Soft Computing for Problem Solving, vol. 1, no. 15, pp. 199-209, 2019.

[5] Junyi Chai, Yonggang Deng, Maochen Guan, Yujie He, Bing Li, and Rui Yan, "Querying Enterprise Knowledge Graph With Natural Language," in Proceedings of the ISWC 2019 Satellite Tracks, 2019.

[6] Vincent Lully, Philippe Laublet, Milan Stankovic, Filip Radulovic, "Enhancing explanations in recommender systems with knowledge graphs," in Procedia Computer Science, Volume 137, 2018, pp 211 - 222, 2018.

[7] Alessandro Moschitti, Kateryna Tymoshenko, Panos Alexopoulos, Andrew Walker, Massimo Nicosia, Guido Vetere, Alessandro Faraotti, Marco Monti, Jeff Z. Pan, Honghan Wu, Yuting Zhao, "Question Answering and Knowledge Graphs," Exploiting Linked Data and Knowledge Graphs in Large Organisations, pp. 181-212, Springer, Cham, 2017.

[8] Honghan Wu, Ronald Denaux, Panos Alexopoulos, Yuan Ren, Jeff Z. Pan, "Understanding Knowledge Graphs," Exploiting Linked Data and Knowledge Graphs in Large Organisations, pp. 147-180, Springer, Cham, 2017.

[9] Ben Hixon, Peter Clark, Hannaneh Hajishirzi, "Learning Knowledge Graphs for Question Answering through Conversational Dialog," in Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Denver, Colorado, 2015.

[10] Xiao Huang, Jingyuan Zhang, Dingcheng Li, Ping Li, "Knowledge Graph Embedding Based Qestion Answering," in In The Twelfth ACM International, Melbourne, VIC, Australia, 2019.

[11] Jacob Devlin, Ming-Wei Chang, Kenton Lee, Kristina Toutanov, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," in Proceedings of NAACL-HLT 2019, pages 4171–4186, Minneapolis, Minnesota, 2019.

[12] Santanu Bhattacharjee, Rejwanul Haque, Gideon Maillette de Buy Wenniger, Andy Way, "Investigating Query Expansion and Coreference Resolution in Question Answering on BERT," Natural Language Processing and Information Systems. NLDB 2020. Lecture Notes in Computer Science, vol 12089. Springer, Cham, pp. 47-59, 2020.

[13] Chi Sun,Xipeng Qiu,Yige Xu,Xuanjing Huang, "How to Fine-Tune BERT for Text Classification?," in Chinese Computational Linguistics. CCL 2019. Lecture Notes in Computer Science, vol 11856. Springer, Cham, Kunming, China, 2019.

[14] Benjamin Muller, Benoıt Sagot, Djame Seddah, "Enhancing BERT for Lexical Normalization," in Proceedings of the 2019 EMNLP Workshop W-NUT: The 5th Workshop on Noisy User-generated Text, pages 297–306, Hong Kong, 2019.

[15] Manish Munikar, Sushil Shakya, Aakash Shrestha, "Fine-grained Sentiment Classification using BERT," in 2019 Artificial Intelligence for Transforming Business and Society (AITB), Kathmandu, Nepal, Nepal, 2019.

[16] Aiting Liu, Ziqi Huang, Hengtong Lu, Xiaojie Wang, Caixia Yuan, "BB-KBQA: BERT-Based Knowledge Base Question Answering," in CCL 2019: Chinese Computational Linguistics, pp 81-92, Kunming, China, 2019.

[17] Dongfang Li, Yifei Yu, Qingcai Chen, Xinyu Li, "BERTSel: Answer Selection with Pre-trained Models," in CoRR abs/1905.07588, arXiv, 2019.

[18] Danilo Croce, Giorgio Brandi, Roberto Basili, "Deep Bidirectional Transformers for Italian Question Answering," in Proceedings of the Sixth Italian Conference on Computational Linguistics (CLiC-it 2019), November 13-15, Bari, Italy, 2019.

[19] Hussein Mozannar, Karl El Hajal, Elie Maamary, Hazem Hajj, "Neural Arabic Question Answering," in Proceedings of the Fourth Arabic Natural Language Processing Workshop, pages 108–118, Florence, Italy, 2019.

[20] Liang Yao, Chengsheng Mao, Yuan Luo, "KG-BERT: BERT for Knowledge Graph Completion," arXiv:1909.03193v2, 2019.

[21] Weijie Liu, Peng Zhou, Zhe Zhao, Zhiruo Wang, Qi Ju, Haotang Deng, Ping Wang, "K-BERT: Enabling Language Representation with Knowledge Graph," arXiv:1909.07606v1, 2019.

[22] Pham Quang Nhat Minh, "A Feature-Rich Vietnamese Named-Entity," arXiv:1803.04375v1, 29 08 2018.

[23] Thanh Vu, Dat Quoc Nguyen, Dai Quoc Nguyen, Mark Dras, Mark Johnson, "VnCoreNLP: A Vietnamese Natural Language Processing Toolkit," in Proceedings of NAACL-HLT 2018: Demonstrations, pages 56–60, New Orleans, Louisiana, June 2 - 4,, 2018.

[24] Zhiguo Wang, Patrick Ng, Xiaofei Ma, Ramesh Nallapati, Bing Xiang, "Multi-passage BERT: A Globally Normalized BERT Model for Open-domain Question Answering," in Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing, pages 5878–5882, Hong Kong, China, 2019.

[25] Wataru Sakata, Tomohide Shibata, Ribeka Tanaka, and Sadao Kurohashi, "FAQ Retrieval using Qery-Qestion Similarity and BERT-Based Qery-Answer Relevance," in In Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval (SI-GIR '19), July 21–25, Paris, France, 2019.

[26] Chen Qu, Liu Yang, Minghui Qiu, W. Bruce Croft, Yongfeng Zhang, and Mohit Iyyer, "BERT with History Answer Embedding for Conversational Question Answering," in In Proceedings of the 42nd Int'l ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'19), July 21–25,, Paris, France, 2019.

[27] M. Romero, "BERT (base-multilingual-uncased) fine-tuned for multilingual Q&A," Hugging Face, 11 3 2020. [Online]. Available: https://huggingface.co/mrm8488/bert-multi-uncased-finetuned-xquadv1. [Accessed 8 6 2020].

[28] Zachary C. Lipton, Charles Elkan, Balakrishnan Naryanaswamy, "Thresholding Classifiers to Maximize F1 Score," arXiv:1402.1892v2, 15 3 2014.

[29] Davide Chicco, Giuseppe Jurman, "The advantages of the Matthews correlation coefficient (MCC) over F1 score and accuracy in binary classification evaluation," BMC Genomics 21, 22 11 2020.