Hindawi
Wireless Communications and Mobile Computing
Volume 2022, Article ID 8045535, 11 pages
https://doi.org/10.1155/2022/8045535

Research Article

Few-Shot Multihop Question Answering over Knowledge Base

Meihao Fan,[1] **Lei Zhang**,[2] **Siyao Xiao**,[1] **and Yuru Liang**[3]

[1]*School of Information Science and Engineering, Chongqing Jiaotong University, Chongqing 400074, China*
[2]*School of Mathematics and Statistics, Chongqing Jiaotong University, Chongqing 400074, China*
[3]*School of Economics and Management, Chongqing Normal University, Chongqing 400074, China*

Correspondence should be addressed to Lei Zhang; zhangleicqjtu@163.com

Received 20 December 2021; Accepted 16 March 2022; Published 6 May 2022

Academic Editor: Yingjie Wang

KBQA is a task that requires to answer questions by using semantic structured information in knowledge base. Previous work in this area has been restricted due to the lack of large semantic parsing dataset and the exponential growth of searching space with the increasing hops of relation paths. In this paper, we propose an efficient pipeline method equipped with a pretrained language model. By adopting beam search algorithm, the searching space will not be restricted in subgraph of 3 hops. Besides, we propose a data generation strategy, which enables our model to generalize well from few training samples. We evaluate our model on an open-domain complex Chinese question answering task CCKS2019 and achieve F1-score of 62.55% on the test dataset. In addition, in order to test the few-shot learning capability of our model, we randomly select 10% of the primary data to train our model, and the result shows that our model can still achieves F1-score of 58.54%, which verifies the capability of our model to process KBQA task and the advantage in few-shot learning.

1. Introduction

Due to the proliferation of artificial intelligence (AI), smart systems have made significant achievements in communication and information extraction [1–8]. Since a sophisticated smart system can bring much convenience and efficiency, the research in this field has attracted extensive attention from academic and industrial circles.

A KBQA system aims to answer questions (QA) by understanding the semantic structure and extract the answers in large knowledge base (KB).Recently, tremendous KBQA models are proposed to effectively utilize KB to answer "simple" questions. Here, "simple" refers to questions that can be answered with a single predicate or a predicate sequence in the KB. For instance, "Who directed Avatar?" is a simple question due to its answer can be obtained by a single triplet fact query (?, director_of, Avatar). To answer such questions, plenty of rule-based [9], keyword-based [10], and synonym-based methods [11–14] have been proposed. However, questions in real life are usually more complex which can only be answered correctly by a multihop query path with constraints. As is shown in Figure 1, for answering a complex question, a sequence of operations needs to be generated, including multihop query and answers combination. Recently, the use of KB to answer such complex questions (KBCQA) has attracted growing interests prodigiously [15]. Previous state-of-art KBCQA models can be categorized into a taxonomy that contains two main branches, namely, information retrieval-based (IR-based) and neural semantic parsing-based (SP-based) model. The IR-based model first recognizes topic entities in the natural language and links them to node entities in knowledge base [16–19]. Then, all nodes surrounding around the topic nodes are regarded as candidate answers, and a score function is used to model their semantic relevance and predict the final answers. Methods based on

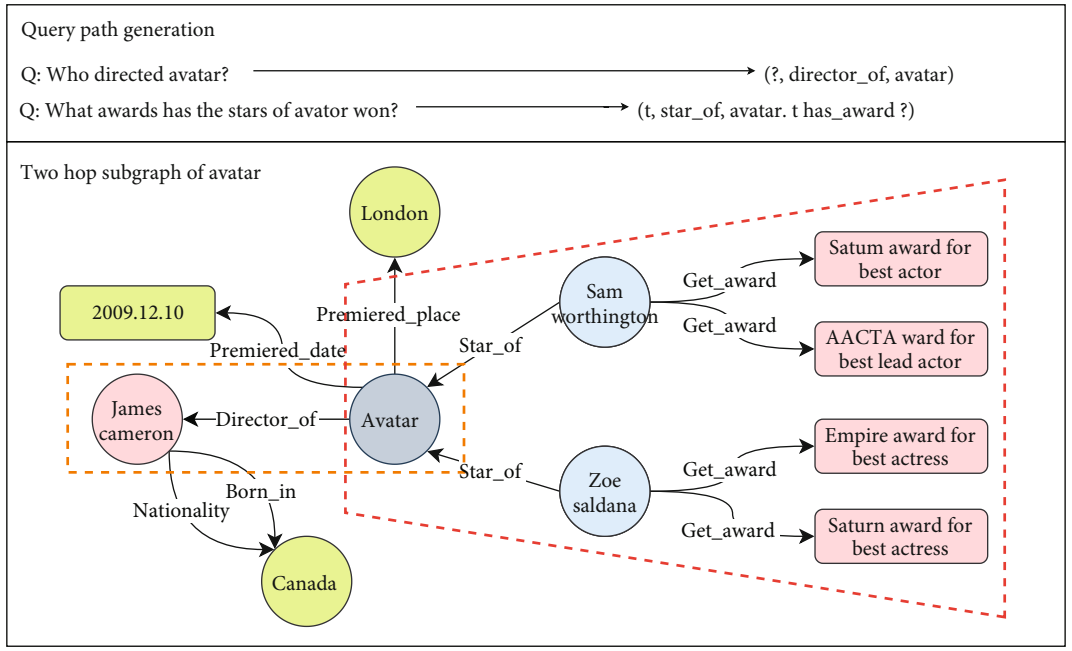

Figure 1: The subgraph of KB and two query paths.

semantic parsing usually includes a Seq2Seq module which converts natural languages into executable query languages and an executor module which executes the generated logical sequence on KB to obtain the final answers [20–24].

However, although the state-of-art models have made great achievements, several challenges still exist. Firstly, the dependency of annotated data is a thorny problem for SP-based models, which is usually settled by using a breadth-first search (BFS) to produce pseudo-gold action sequences and adopting the reinforce learning (RL) algorithm [25, 26]. Yet since BFS will inevitably ignore many other plausible annotations and RL usually suffers from several challenges, such as sparse reward and data inefficiency, the research of SP-based models is immensely hindered. Secondly, both IR-based and SP-based methods suffer from the large searching space. For better performance on KBCQA task, large KBs, such as Wikidata or FreeBase, are usually needed [27, 28]. Although these KBs contain comprehensive knowledge, they also bring vast search space when searching a query path with more than 3 hops. We record the average number of relations in one-hop and multihop subgraphs of a topic entity in our training dataset. It is shown that in one-hop subgraphs, the average number is 515, while in 2-hop and 3-hop subgraphs, it grows to 1920 and 6408, respectively. This exponential growth of generated candidate tuples makes it expensive and difficult for calculation. Thirdly, most previous work requires large KBCQA datasets to train their model, such as complex web questions and QALD [29, 30]. However, these large datasets are usually in English, hindering research in more realistic settings and in languages other than English.

To solve the three problems above, we propose a template-based model consisting of question classification, named entity recognition, query path generation, and path ranking module. Our contribution can be categorized into three fields:

(1) We propose a data-efficient model equipped with a pretrained language model BERT which can achieve high performance but only use tiny amount of data. Thus, our model can be utilized to process KBQA task in some languages without large KBQA datasets

(2) By adopting beam search algorithm and using ERNIE [31] to score for each searching branch, the spatial complexity and time complexity have been greatly dropped, but the generating accuracy still remains competitive

(3) We put forward a method to construct artificial data on predefined schemas of query graphs, allowing our model to process questions with novel categories which are excluded by training set

With the utilize of pretrained language model BERT and predefined schemas of query graphs, our model can effectively extract and filter the query tuples for a complex question. Also, we adopt beam search algorithm to relieve the exponential growth with increasing hops, which make it possible to handle multihop questions.

This paper is organized as follows: In Section 2, we review works on NER and beam search, which are the basis of our experiments. In Section 3 we present the overall architecture and then introduce each key component in detail. In Section 3, we demonstrate the evaluated models and the methodology used to generate the sentence embeddings. In Section 4, we describe the experimental setup and evaluation of the proposed model. Finally, we summarize the contribution of this work in the Section 5.

2. Related Work

Recently, with the rapid development and increasing attention of deep learning, the research on natural language processing has made great process. Especially when supported by emerging word embedding technologies and pretrained language models, the effectiveness of knowledge base question answering has been greatly improved. In this section, we will introduce some previous work related to the submodules of our model including named entity recognition (NER) and beam search algorithm. Besides, some few-shot KBQA models and a template-based model will also be introduced.

Named entity recognition is a key component in NLP systems for question answering, information retrieval, and relation extraction. Early NER models are mainly based on unsupervised and bootstrapped systems [32, 33] or feature-engineering supervised task [34, 35]. Nowadays, researchers tend to use neural network for NER task. NER is often solved as a sequence labeling problem by using the conditional random field (CRF) which requires a set of predefined features. Recently, some effective neural network approaches, especially for bidirectional long short-term memory, significantly improve the performance of CRF for NER task. Huang et al. use two LSTMs to capture past features and future features in sequence tagging task [36]. Then, a CRF layer is used to efficiently grasp the sentence level tag information of the sentence. The BiLSTM CRF is usually employed as the cornerstone of many subsequent improved NER models. BERT BiLSTM CRF uses BERT to embed extract rich semantic features into vectors and sends them to the BiLSTM CRF [37]. This model has achieved state-of-art performance in many NER tasks [38].

Beam search is a common heuristic algorithm for decoding structured predictors. When generating query paths for complex multihop questions, we need to consider longer relation path in order to reach the correct answers. However, the search space grows exponentially with the length of relation paths, bringing expensiveness for calculation and storage. The core idea of beam search is to use a score function to keep Top-K candidate relations instead of considering all relations when extending a relation path. Thus, the definition of score function determines the performance of Bean Search. Chen et al. (2019) proposed to keep only the best matching relation with a path ranking module that considers features extracted from topic entities and semantic information of the generated query paths [20]. Lan et al. (2019) also keep only one candidate relation using a traditional Siamese architecture where both the question and the candidate paths are each separately encoded into a single vector before the two vectors are matched [39]. The experimental results of these two models show little performance dropped but with significant reduction in spatial complexity and time complexity.

Since the expensiveness of constructing the annotated datasets, several works have been focused on few-shot learning for KBQA task. Chada et al. (2021) proposed a simple fine-tuning framework that regards the query path generation as a text-to-text task [40]. By leveraging a pretrained sequence-to-sequence models, their method outperforms many state-of-art models with an average margin of 34.2 F1 points on various few-shot settings of multiple QA benchmarks. Hua et al.(2020) proposed a semantic parsing based method using BFS to find the pseudo-gold annotation of a question and learning a reinforcement learning (RL) policy to generate a query sequence for obtaining the final answer [41].

Our model is most inspired by a template-based Chinese KBCQA model proposed by Wang et al. [42]. They use a pipeline method including a NER module, a query path generation module, and candidate tuple ranking module and process the question step by step. In NER module, they attach the BiLSTM CRF layer with a BERT layer to better understand the semantic information in the question, which gets quite high accuracy in topic entities recognition. Then, they extend one or two relations from the topic entity to generate the query paths and adopt bridging technology to process questions with multiple entities. Finally, a candidate query path ranking module is carefully designed to select the final query path. The differences between their work and our model are that we process the one-entity and multientity questions separately with a question classification module and predefine a set of query schema to restrict the searching space. On the predefined query pattern, we use a strategy to construct artificial questions which improve the ability of the classification model for few-shot learning. Moreover, we adopt beam search algorithm when generating query paths, which helps us achieve comparable performance but only using 10% resource of calculation and storage.

3. Our Method

In this section, we will present the overall architecture (shown in Figure 2) and then introduce each key component of the proposed model in detail.

3.1. Method Overview. The general idea behind our method is to process the question step by step. Given a question, we first encode it with a BERT layer, and then, the representations will be passed to an entity linking module (Section 3.2) of BERT-BiLSTM-CRF layer and a question classification module (Section 3.3) trained with extra manually constructed samples (Section 3.5). With the recognized topic entities and a specific category the question belongs to, we can refer to a more precise schema (Section 3.4) to generate the query path in a narrower searching space. However, since the query graph of a complex question may involve multiple relations, such simple generating program will bring intolerable time complexity and spatial complexity and bring calculating burden to the candidate tuple ranking module. To solve this, we adopt a heuristic algorithm for graph search (Section 3.6) based on a pretrained text-match model, which greatly decreases the number of candidate query paths. Afterwards, a candidate tuple ranking module is designed to sift out the final path using the above PTM-TextMatch model. By executing the golden query tuple, we can retrieve the answer in knowledge base.

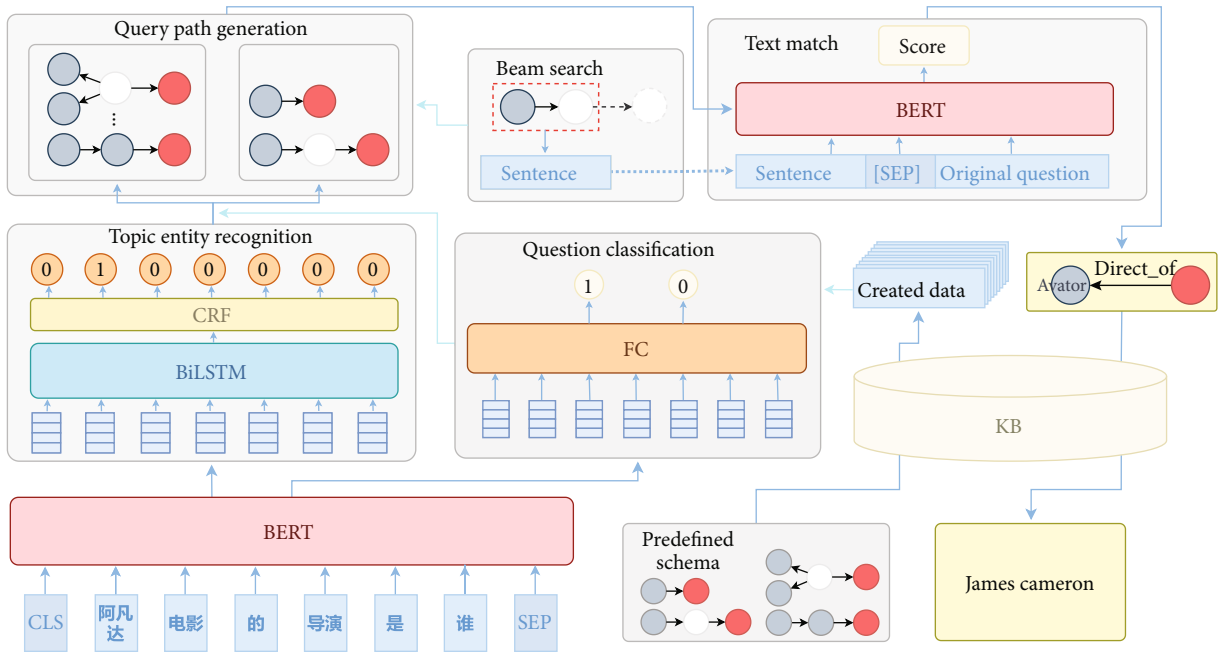

Figure 2: Basic framework of our model.

Besides, we are not to search aimlessly in KB when generating query subgraph. Instead, we refer to a set of predefined schemas of all possible query graphs in complex question answering. This policy will not only narrow the searching space significantly but also provide a semantic framework for reference when constructing artificial questions.

3.2. Node Extractor. The main goal of this module is to identify topic entities in the question. This module includes tokenization with dictionaries, named entity recognition (NER), and entity linking.

3.2.1. Tokenize. Different from English tokenize, Chinese tokenizing usually uses dictionaries as a supplementary to tokenize Chinese question text into Chinese words. In this paper, we use a dictionary provided by CCKS consisting of all subjects in KB, all entities, and their mentions in mention dictionary.

3.2.2. Named Entity Recognition. In the NER module, we encode the question with BERT layer and then pass it through a BiLSTM to capture the information of context and a CRF layer to predict label of each token. Let us use $Q = (t_1, t_1, t_1, \cdots, t_n)$ to represent a tokenized question. We put Q into a BERT layer to encode representations with semantic knowledge. Next, the representations $X_{i=1}^{|Q|}$ are passed through a BiLSTM layer and CRF layer [28].

For each input token, the context information is captured by two LSTMs, where one capture information from left to right and the other from right to left. At each time step t, a hidden vector $\overrightarrow{h_t}$ (from left to right) is computed based on the previous hidden state $\overrightarrow{h_{t-1}}$ and the input at the current step x_t. Then, the forward and backward context representations, generated by $\overrightarrow{h_t}$ and $\overleftarrow{h_t}$, are concatenated into a long vector which we represent as $h_t = [\overrightarrow{h_t} : \overleftarrow{h_t}]$. The basic LSTM function is defined as follows:

$$\begin{bmatrix} \widetilde{c_t} \\ f_t \\ o_t \\ i_t \end{bmatrix} = \begin{bmatrix} \sigma \\ \sigma \\ \sigma \\ \tanh \end{bmatrix} \left(W^T \begin{bmatrix} x_t \\ h_{t-1} \end{bmatrix} + b \right), \tag{1}$$

$$c_t = i_t \odot \widetilde{c_t} + f_t \odot c_{t-1},$$

$$h_t = o_t \odot \tanh(c_t),$$

where W^T and b are trainable parameters; $\sigma()$is the sigmoid function; i_t, o_t, and f_t indicate input, output, and forget gates, respectively; $\odot$ represents the dot product function; and x_t is the input vector of the current time step.

The output vectors of the BiLSTM contain the bidirectional relation information of the words in a question. Then, we adopt CRF to predict labels for each word, considering the dependencies of adjacent labels. The CRF is the Markov random field of Y given a random variable X condition and included an undirected graph G, where Y are connected by undirected edges indicating dependencies. Formally, given the observation variables $H = h_{i=1}^{|Q|}$, and a set of output values $y \in \{0, 1\}$, where $y = 1$ means, the corresponding token is a topic entity, and $y = 0$ is not. CRF defines potential function as

$$p(y|h) = \frac{1}{Z_h} \prod_{s \in S(y,h)} \phi_s(y_s, h_s), \tag{2}$$

where Z_h is a normalization factor overall output values, $S(y, h)$ is the set of cliques of G, and $\phi_s(y_s, h_s)$ is the clique potential on clique s.

Afterwards, in the BiLSTM-CRF model, a softmax over all possible tag sequences yields a probability for the sequency. The prediction of the output sequence is computed as follows:

$$y_* = \arg \max_{y \in \{0,1\}} \sigma(H, y),$$
$$\sigma(H, y) = \sum_{i=0}^{n} A_{y_i, y_{i+1}} + \sum_{i=0}^{n} P_{i, y_i}, \tag{3}$$

where A is a matrix of transition scores, $A_{y_i, y_{i+1}}$ represents the score of a transition from the tag y_i to y_{i+1}, n is the length of a sentence, P is the matrix of scores output by the BiLSTM network, and P_{i, y_i} is the score of the y_i^{th} tag of the i^{th} word in a sentence.

3.2.3. Entity Linking. In this module, we link the recognized named entity to the entity in KB and select a set of candidate topic entities with a mention dictionary. The mention dictionary is provided by CCKS sponsors describing mapping relations from mentions to node entities. After obtaining mentions of entities in a question, we correspond them to relevant node entities. Then, we need to extract helpful features from the mentions and entities to select the potential candidate entities. In this work, we extract six features as follows: the length of entity mention (f_1), the TF value of entity mention (f_2), the distance between the entity mention and interrogative word (f_3), word overlap between question and triplet paths (f_4), and popularity of candidate entities (f_5). The popularity is calculated as $\sqrt{k}$, where k represents the number of relation path the candidate entity has within 2-hop graph. We assume that an entity with larger f_1, f_2, f_4, and f_5 and smaller f_3 is more likely to be a topic entity.

These six features will be calculated and put into a linear weighing layer to output relative scores. Entities with Topk score build the candidate entities set.

The score is calculated using the following function:

$$s = w_1 \cdot f_1 + w_2 \cdot f_2 + w_3 \cdot f_3 + w_4 \cdot f_4 + w_5 \cdot f_5 \tag{4}$$

where f_i represents the i^{th} feature and w_i represents the corresponding weight.

3.3. Question Classification. In order to improve the efficiency of our model, we use a pretrained language model BERT to classify the complex questions into two categories, one topic entity question and multientity question, and process each of them separately. In one entity question, predicted paths usually extend from the topic entity with one relation or a sequence of relation hops. While in multientity questions, correct answers can only be obtained accurately by executing the query paths extended from several topic entities in the question. For instance, the question "Whose husband is the director of Avatar?" is one-entity question because its query paths (?, wife_of, t. t, director_of, Avatar) can be extracted from the "Avatar" through the relations "director_of" and "wife_of" and the transitional entity t. Meanwhile, "Which actors in Avatar born in British?" is a complex question because the correct query paths can only be generated from the entity "Avatar" and "British," respectively, through the relations "actor_of" and "born_in". In addition, we generate artificial questions in a semantic structured form to improve the performance of our classification model. The detailed implementation will be represented in Subsection 3.5.

Given a question, we encode it with words encoding, position encoding, and segment encoding and attach a special token [CLS] at the beginning of a question to separate different sentences. Then, the semantic information will be captured with a multihead attention system, and a dense layer will be attached to obtain the prediction.

3.4. Predefine the Query Schema. The golden key to solving the KBCQA task is to map entities of a question into a specific query graph. A semantic parsing-based model transfers the KBQA task into a Seq2Seq task. By feeding the model with numerous annotated data, SP-based model can understand the semantic framework of a question and refine corresponding query graph. An information retrieval-based model adopts a different method that searches all query graphs surrounding the extracted topic entities and then uses a candidate tuple ranking module to sift the final query graphs. However, with limited data, it is challenging to learn the query structure of questions, let alone changing it to an executable action sequence. In this work, we relieve this problem by predefining the schema of query graph and adopt beam search to pruning the searching space of multihop query paths.

Inspired by Aqqu [43], we propose an inverse solution that we first take a deep insight into numerous Chinese multihop questions and propose eight searching schemas for complex questions as shown in Figure 3. By predefining the schema of query graph, our model can benefit from three aspects:

(a) Predefining the schema introduces prior knowledge, which stipulates the semantic structure of the queried question and greatly prunes the search space

(b) Since the patterns of query tuples are specified, we can easily turn each query tuples into its semantic form and calculate the similarity between the artificial question and real question with a pretrained language model, which we define as the score of the query path we generate

(c) Extra data can be constructed on the enumerated query schema to train the classification model, which allows the model to learn the basic semantic knowledge of classifying questions

We assume that the diversity of candidate tuples will lead to poor performance of candidate query path ranking module. Thus, we divide the query schema into two modules according to number of topic entities the query pattern has.

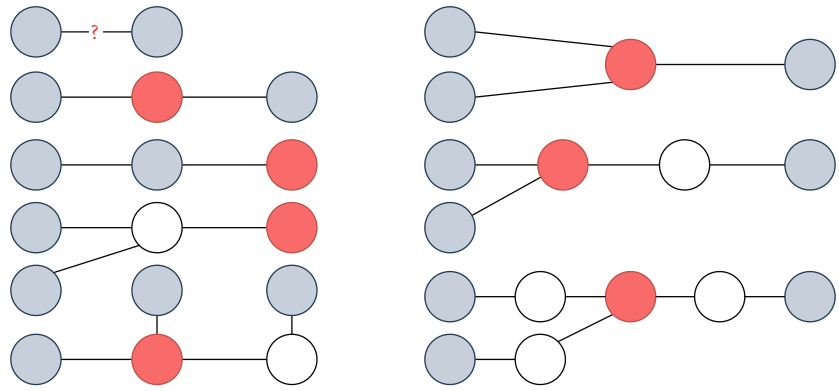

Figure 3: Predefined schema of query path.

Input: KB, question q, topic entity set E, number of hops T
Output: P^T
Initialize: $P^0 \longleftarrow \{e_0\} \in E$
for t=1,2,...,T **do**
  $\tilde{P}^{(t)} \longleftarrow \phi$
  $\tilde{S}^{(t)} \longleftarrow \phi$
  for each $p \in P^{(t-1)}$ **do**
    $e_{t-1} \longleftarrow tail(p)$
    for each $(e_{t-1}, r, e_t) \in KB$ **do**
      if $e_t \in E$ **then**
        $p' \longleftarrow p \oplus (r, e_t)$
      else
        $p' \longleftarrow p \oplus (r)$
      end if
      $\tilde{P}^{(t)} \longleftarrow \tilde{P}^{(t)} \cup \{p'\}$
      $\tilde{S}^{(t)} \longleftarrow \tilde{S}^{(t)} \cup \{Sentence(p')\}$
    end for
  end for
  score all elements in $\tilde{S}^{(t)}$ and rank all corresponding elements in $\tilde{P}^{(t)}$
end for

Algorithm 1: Multihop relation extraction. For each query schema, we generate a set of candidate query paths P^T, where T represents the hop number of the schema.

When generating query paths, we use two separate modules to generate candidate query paths. For one-entity question, we simply search the subgraph of the topic entity within two relation hops. While for questions of multiple entities, we generate query paths on the searching schemas shown in Figure 3. The gray ones represent topic entities we already know. The white one represents transitional entity we need not record, and the red one represents the answer we query. Let n represents the number of candidate topic entities, and m represents the number of true topic entities in a given question. Since combinatorial number C_n^m grows too large when m is greater than 3, we only consider questions containing three or fewer topic entities.

3.5. Artificial Data Construction. For better predicting which class a question belonging to and alleviating the need of labeled training data, we generate substantial artificial questions on the predefined query schemas. In our method, we randomly select a node entity in KB and extend a query path from the entity. When generating a query path, we are not to consider all branches in a random searching schema. Instead, we conduct the algorithm on the predefined query schema which has been introduced in Subsection 3.4. For instance, as for the above question "Whose husband is the director of Avatar?," the corresponding query schema is $(x, r_1, t.t, r_2, e.)$, where x represents the answer and r_1 and r_2 represent any relations in two-hop query path extended from the topic entity e through an intermediate entity t. We generate the artificial question by replacing mentions of topic entities (in this example is "Avatar") and relations ("wife_of", "director_of") with mentions of randomly selected node entities and correlated relations. In addition,

if the query schema is excluded in training samples, we only need to manually construct a fake question corresponding to the query schema and then execute the above steps.

Since our predefined query schema contains semantic structure for both one-entity and multientity questions, our constructed samples can lead the pretrained language model to converge in a direction which is more compatible with our specific classification task. Besides, the ratio of questions of different query patterns should be carefully controlled in order to improve the generalization of created data.

Although our constructed questions have some differences from the real questions in semantic expression, our model can still learn extra semantic structure of questions in two classes. In our experiment, we constructed 5 k artificial questions and use them to train our classification model. With the help of pretrained language model, our model can handle some questions that have never shown in training set. As the results in Section 4 shown, given only 10% of training data, our model can achieve good performance in classifying the questions.

3.6. Beam Search. It is worth to note that when extending multihop relations of the two type questions above, query path generation module often suffers from the vast searching space. To solve this, we adopt a heuristic algorithm beam search algorithm equipped with a pretrained language model BERT to score for each breach of relations; thus, we avoid exhaustive search on irrelevant relations. When extending a new relation path at n-step, we try to add the relation r_n to the previous generated query path R_{n-1} and use the strategy introduced in Artificial Data Construction to transfer the graph into a semantic form S_n. Then, S_n and original question Q are tokenized and concatenated with a special token [SEP] as

$$\text{input} = [\text{CLS}]S_n[\text{SEP}]Q. \tag{5}$$

This two sentences are fed into a pretrained language model of downstream task to calculate the semantic similarity which represents the score for r_n given a subquery path R_{n-1}. The formulation is defined as

$$Sco(r_n|R_{n-1}) = \text{BERTLayer}(\text{input}). \tag{6}$$

At each extending step, we only consider relations with Topk score for further search, which significantly excluded some irrelevant query branches. The result in Section 4.3.1 shows that by adopting the beam search algorithm, the accuracy of query path generation remains competitive, but the number of candidate paths decreases above 80%. The detailed description is seen in Algorithm 1.

4. Experiments

In this section, we study the performance our model achieves on complex question answering with limited training data. We take an insight into each module and conduct ablation experiments to better understand our model.

Table 1: Number of triples, entity type, and entity linking in PKU-Base.

Type	Triples	Entity type	Entity linking
Number of data	61,006,527	25,182,627	13,930,117

Table 2: Results of ablation experiments in entity linking module.

Type	One entity	Multientity
Baseline	0.848	0.726
w/o f_1	0.841	0.733
w/o f_2	0.848	0.721
w/o f_3	0.843	*0.744*
w/o f_4	0.838	0.706
w/o f_5	*0.849*	0.637

Table 3: We evaluate our model on primary training datasets, where created samples are excluded.

Data	Train	Valid	Test
10%	82.90	84.31	80.13
10% + created data	*87.51*	*89.54*	*82.75*
50%	94.95	*93.99*	88.50
50% + created data	*95.12*	93.72	*89.41*
100%	97.39	95.42	88.76
100% + created data	*99.09*	*95.45*	*91.11*

4.1. KB and Datasets. Our model uses an open-domain KB PKU-Base, which adopts resource description framework (RDF) as their data format and contains billions of SPO (subject, predicate, and object) triples [30], as shown in Table 1. We train and evaluate our model on CCKS datasets, which contain 2298, 766, and 766 pairs of questions.

4.2. Entity Linking. In entity linking module, we remove each feature of candidate entities to observe the influence on the performance of entity linking models. The left column is disassembled model, and the right is its recall of recognizing topic entities.

As is shown in Table 2, without f_3, the recall of multientity questions surprisingly increased while accompanied with a sacrifice of accuracy for one-entity questions. Similarly, without f_5, the topic entity extracting accuracy for questions of one topic entity increases, but the accuracy for multientity question drops. Moreover, excluding any of other features, the performance of entity linking model drops, which verifies their contribution for this module. Based on the results, we can modify the entity linking module by discarding feature f_5 in one-entity question's entity linking stage while only considering f_1, f_2, f_4, and f_5 when processing multientity questions. This will be included in our further study.

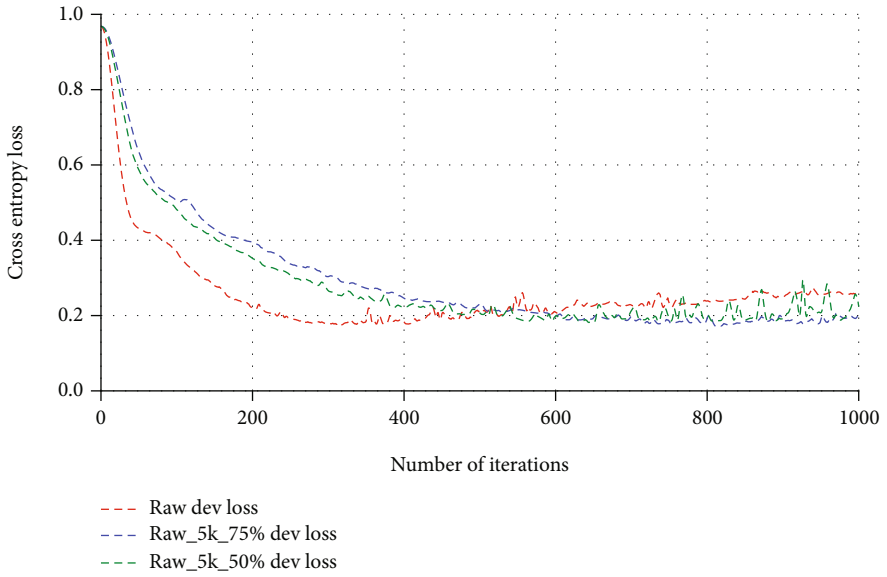

Figure 4: Loss of classification model.

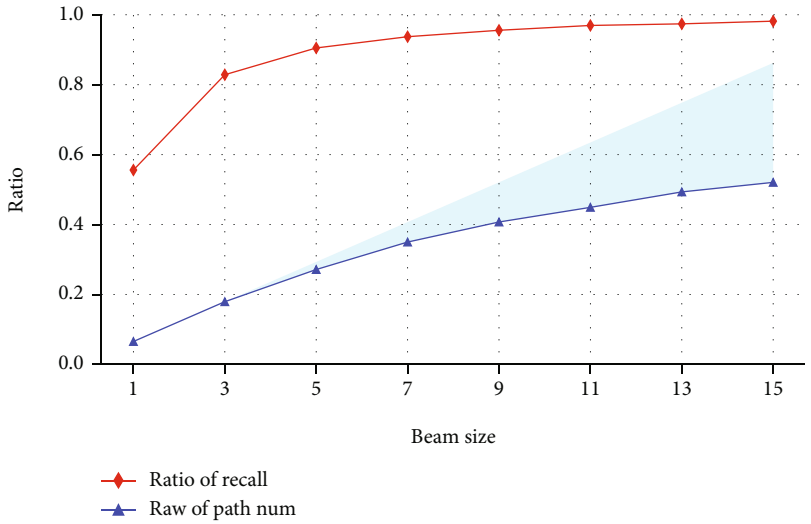

Figure 5: Ratio of recall and path numbers.

4.3. Question Classification. In this module, we construct 5 k artificial data based on the predefined query graphs and attached them to the training datasets. In order to evaluate the learning capability of our model on small amount of data, we train our model on 10%, 50%, and 100% randomly selected samples of primary training datasets and compare their performance with those additionally attached with certain number of created training samples.

Notably, when adding the constructed samples, we should carefully control the quantity according to the number of primary training samples. For one thing, negligible improvement of the learning ability can be brought, if the quantities of the added samples are too small. For the other, adding too many constructed data will bring knowledge noise, which leads the model to learn a distribution far away from the primary datasets. In our experiment, for 10%, 50%, and 100% primary training data, we add 0.05 k, 0.5 k, and 3.75 k manually constructed samples, respectively. The result is illustrated in Table 3.

From the above table, we find that when attached with manually constructed samples, our model's performance has improved on both partial and whole primary data. Our strategy can bring more significant improvement especially when given a small amount of training data. Moreover, we can see an obvious improvement of the prediction on training datasets, which indicates that appropriate number of created samples can make the model better fit the distribution of training data.

We owe the model's out performance to the introduction of prior knowledge. Due to the diversity of the samples in datasets, the test set may contain questions whose semantic structures have not appeared in training set. In this zero-

shot or few-shot situation, the model may have difficulty predicting the correct class. However, with additional created samples, our model can learn the predefined semantic structures. If these structures appear in test sets while not included by training set, the performance of our model will be improved. Thus, our model may need more steps to converge.

To verify the idea, we record the loss of each iteration when training with total primary data attached with 0%, 50%, and 75% created data, as shown in Figure 4.

We find that when training primary data attached with 0%, 50%, and 75% created data, our model converges at about 280, 550, and 760 steps, respectively, which indicates that with more created data, the model needs more iterations to converge.

4.4. Beam Search. For better exhibiting the effect of beam search (BS), we select 653 questions whose query path containing 2 hops of relations to test our methods. In the experiment, we design the benchmark by enumerating all the query paths within two-hop relations of the topic entity and recording the average number of query paths N. Notably, we only use BS algorithm at first hop, while searching for the second hop, we only extend from the reserved Top-K subquery path filtered by the BS algorithm and keep all the two hops query paths. By setting different beam size, we can observe the influence on the recall and number of generated query paths.

Figure 5 shows that a larger beam size will bring an increase in both recall and number of candidate query paths. Through further observation, we notice that the growth of both indexes slow down. The retarded growth of recall is intelligible. Due to the existence of upper bound, if the beam size is large enough, the recall will approach to and finally reach 1.0. However, the retarded increasing speed of the number of candidate tuples can illustrate something. When designing the score function for BS, we use a PTLM model to calculate the similarity of generated query paths and primary questions. Thus, the remaining one-hop relations are usually more relevant to the semantic information in the primary question. As the Figure 5 shows, extended from a relation with lower semantic score, the second hop tends to generate fewer query paths. Since the relation whose tail has more triples may have more probability to be the component of golden query path, we assume that the language model can be interpreted as a probabilistic model not only in the dimension of words but also in the dimension of query paths.

4.5. Final Result. We evaluate our model in the CCKS2019 datasets and compare our performance with a start-of-art model proposed by Wang et al. [42]. Their model first generated all query paths within 2 hops and adopted bridging technologies to handle questions with multiple topic entities. In candidate tuple ranking module, Lan [5] uses a PTLM model to calculate scores for generated query paths. Notably, their model introduces negative samples when training the semantic match model. Besides, since introducing bridging technology may harm the predicting performance of one-entity questions, they adopt a literal match technology to rerank the generated query path.

Table 4: Comparative results between our best model with other models.

Method	Negative sample	Avg F_1
Wang (baseline)	3	56.70
Wang (bridging)	3	58.60
Wang (bridging+literal match)	3	61.50
Wang (bridging+literal match)	1	61.10
Wang (bridging+literal match)	5	59.40
Our model (with 10% data)		58.54
Our model (with 100% data)		*62.55*

We implement their model and run it on a RTX 2080. It must to be pointed out that due to the difference of experimental equipment and subtle distinction of our datasets, the performance we obtain has some discrepancy with Wang proposed. However, since both our model and theirs are trained in the same experiment environment, the comparison is still persuasive in Table 4.

The result shows that our method is data-efficient and high-performed. Only using 10% data, our model can achieve competitive result. Moreover, when using 100% data, our model outperforms at over 1.0 point.

5. Conclusion

This paper proposes a KBQA system equipped with pretrained language model to handle multihop questions. We have shown that our model has the capability of answering multihop questions given small amount of data. Besides, experiments have been conducted to demonstrate that, by adopting beam search algorithm, we can achieve competitive results with much smaller cost of calculation and storage, which shows the superiority of our model for few-shot KBCQA task.

Data Availability

The dataset we use to train our model is CCKS2019 dataset, which can be accessed by the URL "https://github.com/pkumod/CKBQA." The knowledge base we used is also available by the URL "https://github.com/pkumod/gAnswer/tree/pkubase."

Conflicts of Interest

No competing interests exist within this work.

Acknowledgments

This work was partially supported by the Group Building Scientific Innovation Project for Universities in Chongqing (CXQT21021), the Innovation and Entrepreneurship Training Program for College Students (202110618001), and the Joint Training Base Construction Project for Graduate Students in Chongqing (JDLHPYJD2021016). We are also

grateful to Ren Li for providing us experimental equipment. Besides, it is worth noting that this work is preprinted at http://arxiv.org [44].

References

[1] Z. Sun, Y. Wang, Z. Cai, T. Liu, X. Tong, and N. Jiang, "A two-stage privacy protection mechanism based on blockchain in mobile crowdsourcing," *International Journal of Intelligent Systems*, vol. 36, no. 5, pp. 2058–2080, 2021.

[2] Y. Wang, Z. Cai, Z. H. Zhan, B. Zhao, X. Tong, and L. Qi, "Walrasian equilibrium-based multiobjective optimization for task allocation in mobile crowdsourcing," *IEEE Transactions on Computational Social Systems*, vol. 7, no. 4, pp. 1033–1046, 2020.

[3] Z. P. Cai and Z. B. He, "Trading private range counting over big IoT data," in *2019 IEEE 39th International Conference on Distributed Computing Systems(ICDCS)*, pp. 144–153, Dallas, TX, USA, 2019.

[4] Y. Wang, Y. Gao, Y. Li, and X. Tong, "A worker-selection incentive mechanism for optimizing platform-centric mobile crowdsourcing systems," *Computer Networks*, vol. 171, article 107144, 2020.

[5] Y. Lan and J. Jiang, "Query graph generation for answering multi-hop complex questions from knowledge bases," in *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, p. 969974, 2020.

[6] Z. Lu, Y. Wang, Y. Li, X. Tong, C. Mu, and C. Yu, "Data-driven many-objective crowd user selection for mobile crowdsourcing in industrial IoT," *IEEE Transactions on Industrial Informatics*, p. 1, 2021.

[7] C. Liang, J. Berant, Q. Le, K. D. Forbus, and N. Lao, "Neural symbolic machines: Learning semantic parsers on freebase with weak supervision," 2016, https://arxiv.org/abs/1611.00020.

[8] J. S. Sharath and R. Banafsheh, "Question answering over knowledge base using language model embeddings," in *2020 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–8, Glasgow, United Kingdom, 2020.

[9] S. Ou, C. Orasan, D. Mekhaldi, and L. Hasler, "Automatic question pattern generation for ontology-based question answering," in *Flairs Conference*, pp. 183–188, Menlo Park, 2008.

[10] C. Unger and P. Cimiano, "Pythia: compositional meaning construction for ontologybased question answering on the semantic web," in *International conference on application of natural language to information systems*, pp. 153–160, Springer, Berlin, Heidelberg, 2011.

[11] C. Unger, L. Bühmann, J. Lehmann, A. C. Ngonga Ngomo, D. Gerber, and P. Cimiano, "Template-based question answering over RDF data," in *Proceedings of the 21st international conference on World Wide Web*, pp. 639–648, Lyon, France, 2012.

[12] M. Yahya, K. Berberich, S. Elbassuoni, M. Ramanath, V. Tresp, and G. Weikum, "Natural language questions for the web of data," in *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pp. 379–390, Jeju Island, Korea, 2012.

[13] C. Unger and C. P. Pythia, "Natural language question answering over RDF: a graph data driven approach," in *Proceedings of the 2014 ACM SIGMOD international conference on Management of data*, pp. 313–324, New York, United States, 2014.

[14] W. Zheng, L. Zou, X. Lian, J. X. Yu, S. Song, and D. Zhao, "How to build templates for rdf question/answering: an uncertain graph similarity join approach," in *Proceedings of the 2015 ACM SIGMOD international conference on management of data*, pp. 1809–1824, New York, United States, 2015.

[15] B. Fu, Y. Qiu, C. Tang, Y. Li, H. Yu, and J. Sun, "A survey on complex question answering over knowledge base: recent advances and challenges," 2020, https://arxiv.org/abs/2007.13069.

[16] L. Dong, F. Wei, M. Zhou, and K. Xu, "Question answering over freebase with multi-column convolutional neural networks," in *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pp. 260–269, Beijing, China, 2015.

[17] Y. Hao, Y. Zhang, K. Liu et al., "An end-to-end model for question answering over knowledge base with cross-attention combining global knowledge," in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 221–231, Vancouver, Canada, 2017.

[18] H. Sun, B. Dhingra, M. Zaheer, K. Mazaitis, R. Salakhutdinov, and W. W. Cohen, "Open domain question answering using early fusion of knowledge bases and text," in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 4231–4242, Brussels, Belgium, 2018.

[19] H. Sun, T. Bedrax-Weiss, and W. W. Cohen, "Pullnet: open domain question answering with iterative retrieval on knowledge bases and text," 2019, https://arxiv.org/abs/1904.09537.

[20] Z. Y. Chen, C. H. Chang, Y. P. Chen, J. Nayak, and L. W. Ku, "UHop: An unrestricted-hop relation extraction framework for knowledge-based question answering," 2019, https://arxiv.org/abs/1904.01246.

[21] K. Luo, F. Lin, X. Luo, and K. Zhu, "Knowledge base question answering via encoding of complex query graphs," in *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pp. 2185–2194, Brussels, Belgium, 2018.

[22] G. Maheshwari, P. Trivedi, D. Lukovnikov, N. Chakraborty, A. Fischer, and J. Lehmann, "Learning to rank query graphs for complex question answering over knowledge graphs," in *International semantic web conference*, pp. 487–504, Auckland, New Zealand, 2019.

[23] S. Zhu, X. Cheng, and S. Su, "Knowledge-based question answering by tree-to-sequence learning," *Neurocomputing*, vol. 372, pp. 64–72, 2020.

[24] Y. Sun, L. Zhang, G. Cheng, and Y. Qu, "SPARQA: skeleton-based semantic parsing for complex questions over knowledge bases," in *Proceedings of the AAAI Conference on Artificial Intelligence*, pp. 8952–8959, 2020.

[25] Y. Hua, Y. F. Li, G. Qi, W. Wu, J. Zhang, and D. Qi, "Less is more: data-efficient complex question answering over knowledge bases," 2020, https://arxiv.org/abs/2010.15881.

[26] G. A. Ansari, A. Saha, V. Kumar, M. Bhambhani, K. Sankaranarayanan, and S. Chakrabarti, "Neural program induction for KBQA without gold programs or query annotations," *IJCAI*, pp. 4890–4896, Macao, China, 2019.

[27] D. Vrandečić and M. Krötzsch, "Wikidata: a free collaborative knowledgebase," *Communications of the ACM*, vol. 27, pp. 78–85, 2014.

[28] K. Bollacker, C. Evans, P. Paritosh, T. Sturge, and J. Taylor, "Freebase: a collaboratively created graph database for structuring human knowledge," in *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, Vancouver, Canada, 2008.

[29] A. Talmor and J. Berant, "TheWeb as a knowledge-base for answering complex questions," 2018, https://arxiv.org/abs/1803.06643.

[30] C. Unger, C. Forascu, V. Lopez et al., "Question answering over linked data (QALD-4)," *Working Notes for CLEF 2015-Conference and Labs of the Evaluation forum*, Toulouse France, 2014.

[31] Y. Sun, S. Wang, Y. Li et al., "Ernie: enhanced representation through knowledge integration," 2019, https://arxiv.org/abs/1904.09223.

[32] O. Etzioni, M. Cafarella, D. Downey et al., "Unsupervised named-entity extraction from the web: an experimental study," *Artificial Intelligence*, vol. 165, no. 1, pp. 91–134, 2005.

[33] M. Collins and Y. Singer, "Unsupervised models for named entity classification," in *1999 Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora*, 1999.

[34] G. D. Zhou and J. Su, "Named entity recognition using an HMM-based chunk tagger," in *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, pp. 473–480, Philadelphia, Pennsylvania, United State, 2002.

[35] R. Malouf, "Markov models for language-independent named entity recognition," in *COLING-02: The 6th Conference on Natural Language Learning 2002 (CoNLL-2002)*, 2002.

[36] Z. Huang, W. Xu, and K. Yu, "Bidirectional LSTM-CRF models for sequence tagging," 2015, https://arxiv.org/abs/1508.01991.

[37] Z. Dai, X. Wang, P. Ni, Y. Li, G. Li, and X. Bai, "Named entity recognition using BERT BiLSTM CRF for Chinese electronic health records," in *2019 12th international congress on image and signal processing, biomedical engineering and informatics (cisp-bmei)*, pp. 1–5, Suzhou, China, 2019.

[38] W. Liu, X. Fu, Y. Zhang, and W. Xiao, "Lexicon enhanced Chinese sequence labelling using BERT adapter," 2021, https://arxiv.org/abs/2105.07148.

[39] Y. Lan, S. Wang, and J. Jiang, "Multi-hop knowledge base question answering with an iterative sequence matching model," in *IEEE International Conference on Data Mining (ICDM)*, pp. 359–368, Beijing, China, 2019.

[40] R. Chada and P. Natarajan, "FewshotQA: a simple framework for few-shot learning of question answering tasks using pre-trained text-to-text models," 2021, https://arxiv.org/abs/2109.01951.

[41] Y. Hua, Y. F. Li, G. Haffari, G. Qi, and T. Wu, "Few-shot complex knowledge base question answering via meta reinforcement learning," in *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 5827–5837, 2002.

[42] X. L. Wang, S. C. Li, Z. H. Yang et al., "A Chinese KBQA system based on pre-trained language model," *Journal of Shanxi University(Natural Science Edition)*., vol. 43, pp. 955–962, 2020, (in Chinese).

[43] H. Bast and E. Haussmann, "More accurate question answering on freebase," in *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*, pp. 1431–1440, New York, United States, 2015.

[44] F. Meihao, "Few-shot multi-hop question answering over knowledge base," 2021, https://arxiv.org/abs/2112.11909.