Few-shot Multi-hop Question Answering over Knowledge Base

Meihao Fan
fmhtodayisniceday@gmail.com

Abstract

Previous work on Chinese Knowledge Base Question Answering has been restricted due to the lack of complex Chinese semantic parsing dataset and the exponentially growth of searching space with the length of relation paths. This paper proposes an efficient pipeline method equipped with a pre-trained language model and a strategy to construct artificial training samples, which only needs small amount of data but performs well on open-domain complex Chinese Question Answering task. Besides, By adopting a Beam Search algorithm based on a language model marking scores for candidate query tuples, we decelerate the growing relation paths when generating multi-hop query paths. Finally, we evaluate our model on CCKS2019 Complex Question Answering via Knowledge Base task and achieves F1-score of 62.55% on the test dataset. Moreover when training with only 10% data, our model can still achieves F1-score of 58.54%. The result shows the capability of our model to process KBQA task and the advantage in few-shot learning.

1 Introduction

KBQA is a NLP task that needs to understand the semantic structure of a Question and then use Knowledge Base(KB) to search the Answer. Since KBQA can provide significant convenience and efficiency for human society, extensive attention has been attracted from academic and industrial circles all over the world.

Recently, tremendous KBQA models are proposed to effectively utilize KB to answer ‘simple’ questions. Here ‘simple’ refers the questions that can be answered with a single predicate or predicate sequence in the KB. For instance, “Who directed Avatar?” is a simple question due to its answer can be obtained by a triplet fact query (?, director of, Avatar). To answer a simple question, plenty of Rule-based(Orasan et al., 2008[11]), Keyword-based(Unger et al., 2011[13]) and Synonym-based methods(Unger et al., 2012[12]; Yahya et al., 2012[14]; Zou et al., 2014[15]; Zheng et al., 2015[16]) have been proposed. However, questions in real life are usually more complex. As is shown in Figure1 for answering a complex question, a sequence of operations need to be executed, such as multi-hop query and answer combination. Recently more and more studies are focused on Answering Complex Questions via Knowledge Base(KBCQA). Previous state-of-art KBCQA models can be categorized into a taxonomy that contains two main branches, namely Information Retrieved-based(IR-based) and Neural Semantic Parsing-based(SP-based). The IR-based model(Dong et al., 2017[1]; Hao et al., 2017[2]; Sun et al., 2018[3]; Chen et al., 2019[9];) first recognizes topic entities in the natural language and links them to Node Entities of Knowledge Base. 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 Semantic Parsing(Luo et al., 2018[5]; Maheshwari et al., 2019[6]; Zhu et al., 2020[7]; Chen et al., 2020[8]; Sun et al., 2020[10]) usually includes a Seq2Seq module which converts natural languages into executable query languages.
and a Executor Module which executes the generated logical sequence on KB to obtain the final answers.

However, although the state-of-art models have made great achievements, several challenges still exist. Firstly, the dependency in 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 or adopting a Reinforce-Learning (RL) algorithm ([20] [21]). Yet since BFS will inevitably ignores many other plausible annotations and RL usually suffers from several challenges, such as Sparse Reward and Data inefficiency, the research of SP-based models are 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 [22] or FreeBase [23] are usually required. Although these KBs contain comprehensive knowledge, searching a query path with more than 3 hops often leads to vast search space. Specifically, in SP-based methods, searching desired action sequences would consume a considerable amount of time and memory, while in IR-based methods, numerous candidate query tuples will be extracted when generating query graph for multi-hop complex questions. Thirdly, most previous work requires Large KBCQA datasets to train their model, such as ComplexWebQuestions [24] and QALD [25]. However, this Large Datasets are usually in English, hindering research in more realistic settings and in languages other than English. To solve the above three problems, we propose an Information Retrieval-based model consisting of Question Classification, Named Entity Recognition, Query Paths Generating and Path Ranking Module. Our contribution can be categorized as three fields:

1. We propose a data-efficient model equipped with a Pre-Trained Language Model BERT[^1] which can achieve high performance but only using 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 BERT to score for each searching branch, the spatial Complexity and Time Complexity has been greatly dropped but generating accuracy is still competitive.

3. We put forward a method to construct artificial data on pre-defined schema of query graphs, which allows our model to process questions with novel categories which is excluded by training set.

This paper is organized as follows: In Section 2 we review work on NER and Beam Search, which are the basis of our experiments. In Section 3 we presents the overall architecture, and then introduces each key component of the proposed model in detail. In Section 4 we describe 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.

[^1]: For better performance, We select ERNIE[^30] as our pre-trained language model.
2 Related Work

Recently, with the rapid development and increasing attention of deep learning, the research on natural language processing has made great progress. 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 sub-modules of our model including Named Entity Recognition (NER) and Beam Search algorithm. Besides, some few-shot KBQA model and an Information Retrieval-based models will also be introduced.

Named Entity Recognition is a key component in NLP systems for question answering, information retrieval, relation extraction, etc. Early NER models are mainly based on unsupervised and bootstrapped system(32 Collins and Singer et al., 1999; 33 Etzioni et al., 2005) or Feature-engineering supervised task. (34 Zhou and Su et al., 2002; 35 Malouf et al., 2002). Nowadays, people 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 require a set of pre-defined features. Recently, some effective neural network approaches, especially for Bi-directional Long Short-Term Memory, significantly improves the performance of CRF for NER task. 25 Huang et al. using two LSTMs to capture past features and future features in sequence tagging task. Then a CRF layer is used to efficiently grasp the sentence level tag information of the sentence. The BiLSTM-CRF was 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 send them to the BiLSTM CRF. This model has achieve state-of-art performance in many NER tasks.

Beam Search is a common heuristic algorithm for decoding structured predictors. When generating query paths for complex multi-hop question, 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 determine the performance of Beam Search. Chen et al. (2019) 28 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. 29 also keep only one candidate relation using a traditional Siamese architecture where both the question and the candidate path are each separately encoded into a single vector before the two vectors are matched. This experiment results of this two model show little performance dropped but with significance reduction in spatial Complexity and Time Complexity.

Since the expensive of constructing the annotated datasets, several works have been focused on few-shot learning for KBQA task. Chada et al.(2021) 37 proposed a simple fine-tuning framework that regards the query path generation as a text-to-text task. By leveraging a pre-trained 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) 38 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.

Our model is most inspired by a Information Retrived-based Chinese CKBQA model proposed by Cao et al. 30. They use a pipeline method including a NER module, a query paths generating module and candidate tuples ranking module and process the question step by step. In NER module, they attach the BiLSTM CRF layer with a BERT layer to better understanding the semantic information in the question, which arrives quite high accuracy in topic entities recognition. Then, they extend one or two relations from the topic entity to generating the query paths and adopting Bridging technology to process question with multiple entities. Finally, a candidate query paths ranking module is carefully designed to select the final query path and execute it on Knowledge Base. The most differences between their work and our model are that we process the one-entity and multi-entity 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 a Beam Search algorithm when generating query paths, which helps us achieve comparable performance but only using only 10% resource of calculation and storage. Most importantly, as our
model is a framework which facilitates the use of different models, we can expect the performance to remain competitive over time.

3 Our Method

In this section, we will present the overall architecture (shown in Figure 2), and then introduces 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 pass to a Entity Linking module (Sec. 3.2) and a Question Classification Module (Sec. 3.3) which is trained with extra manually constructed samples (Sec. 3.5). With the recognized topic entities and a specific category the question belongs to, we can refer to a more precise schema (Sec. 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 pass calculating burden to the Candidate Tuples Ranking module. To solve this, we adopt a Heuristic algorithm for graph search based on a Pre-trained TextMatch model (Sec. 3.6), which greatly decrease the number of query paths. Afterwards, a Candidate Tuples Ranking module is designed to sift out the final path using the above PTM-TextMatch model. By execute the query tuple, we can retrieve the answer in Knowledge Base.

Besides, we are not to search aimlessly in KB when generating query subgraph, instead we refer to a set of pre-defined schema 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 NE is to identify the topic entity in the question. This module includes Tokenize with dictionary, Named Entity Recognition (NER) and Entity Linking.

1) Tokenize: Different from English Tokenize, Chinese Tokenizing usually use dictionary as a supplementary or use Statistical Language Model. In this paper, we use dictionary as supplementary to tokenize Chinese question text into Chinese words. The dictionary is provided by CCKS consisting of all subjects of KB, all entities and its mentions in Mention Dictionary.
2) Named Entity Recognition: In the NER module, we encode the question with BERT layer, and then pass it through a BiLSTM and CRF layer to predict label of each tokens. Let use $Q = (t_1, t_2, t_3, ..., t_n)$ to represent a tokenized question. We put $Q$ into a BERT to encode representations with semantic knowledge. Next, the representations $Q = X_{i=t-1}^{|Q|}$ are passed through a BiLSTM layer and CRF layer[23].

BiLSTM uses two LSTMs to learn each token of the sequence based on both the past and the future context of the token. One LSTM processes the question from left to right, the other one from right to left. Each LSTM[23] has an input gate, an output gate and a cell activation vector $c_t$. At each time step $t$, a hidden vector $h_t$ (from left to right) is computed based on the previous hidden state $h_{t-1}$ and the input at the current step $x_t$. Then The forward and backward context representations, generated by $\overrightarrow{h_t}$ and $\overleftarrow{h_t}$ respectively, 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 follow:

$$\begin{bmatrix}
\tilde{c}_t \\
f_t \\
o_t \\
i_t \\
h_t
\end{bmatrix} =
\begin{bmatrix}
\sigma \\
\sigma \\
\sigma \\
tanh
\end{bmatrix}
\begin{bmatrix}
W^T \\
[ x_t \\
h_{t-1} ]
\end{bmatrix} + b$$

(1)

$$c_t = i_t \odot \tilde{c}_t + f_t \odot c_{t-1}$$

(2)

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

(3)

where $W^T$ and $b$ are trainable parameters; $\sigma()$ is the sigmoid fuction; $i_t, o_t, f_t$ indicating input, output and forget gates respectively; $\odot$ represents the dot product function; $x_t$ 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 $Y$ to given a random variable $X$ condition and includes a undirected graph $G$ where $Y$ are connected by undirected edges indicating dependencies. Formally, given observations 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 a potential functions as below:

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

where $Z_h$ is a normalization factor overall output values, $S(y, h)$ is the set of cliques of $G$. $\phi_s(y_s, h_s)$ 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 sequence $y$. The prediction of the output sequence is computed as follows:

$$y_\ast = \text{argmax}_{y \in \{0, 1\}} \sigma(H, y)$$

$$\sigma(H, y) = \sum_{i=0}^n A_{y_i, y_{i+1}} + \sum_{i=0}^n P_{y_i, y_{i+1}}$$

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. $P_{y_i, y_{i+1}}$ is the score of the $i$th word in a sentence.

3) Entity Linking: In this module, we link the recognized named entity into the entity in KB and select a set of candidate topic entities with a Mention Dictionary. The Mention Dict is a dictionary provided by CCKS Sponsors describing mapping relations from mention to node entities. After obtaining mentions 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 protential candidate entities. In this work, we extract six features as below: 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 $k$ represent the number of relation path the candidate entity has within 2 hop graph. We assume that a entity with larger $f_1, f_2, f_4, f_5$ and smaller $f_3$, are more likely to be a topic entity.
This six features will be caculated and be put into a linear weighing layer to output relative scores. We select entities with Topk score into the candidate entities set. The score is caculated using the following fuction:

\[ s = w_1 \cdot f_1 + w_2 \cdot f_2 + w_3 \cdot f_3 + w_4 \cdot f_4 + w_5 \cdot f_5 \]

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 pre-trained language model BERT to classify the complex question into two categories: one topic entity question and multi-entity question, and then process them separately. In one entity question, predicted paths usually extend from one topic entity with one relation or a sequence of relation hops. While in multi-entity 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 Avator?” is one-entity question because its query paths (?, wife of, t, t, director of, Avator) can be extracted from the “Avator” through the relations “director of” and “wife of” and the transitional entity t. Meanwhile, “Which actors in Avator born in British?” is a complex question because the correct query paths can only be generated from the entity “Avator” 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 begin of a question to separate different sentences. Then the semantic information will captured with a multi-head attention system and a Dense Layer will be attached to obtain the predicts.

### 3.4 Predefine the Query Schema

The golden key to solve the KBCQA task is to map entities of a question into a specific query graph. A Semantic Parsing-based model transfer the KBQA task into a Seq2Seq task. By feeding the model with plenty of annotated data, SP-based model can understand the semantic framework of a question and refine corresponding query graph. An Information Retrived-based model adopt a different method that it searching all query graphs surrounding the extracted topic entities and then use a Candidate Tuple Ranking Module to sift the final query graphs. However, with limited data, it is challenging to learn to understand the query structure of question let alone changing it to an executable action sequence. In this work, we relieve the problem by predefining the schema of query graph and adopt Beam Search to prune the searching space of multi-hop query paths.

Inspired by Aqqu[31], we propose a inverse solution that we first take a deep insight into numerous Chinese multi-hop questions and propose five searching schema for complex questions as shown in Figure 3. By predefining the schema of query graph, our model can benefit from several advantages:

a) Predefining the schema introduces prior knowledge stipulating the semantic structure of the queried question, which greatly prunes the search space.

b) Since the schema of query tuple are specified, we can construct a artificial question for each generated tuple. Then we caculate the similarity between the artificial question and real question with a pre-trained language model, which we define as the score of the query path we generate.

c) Extra data can be constructed on the enumerated query graph to train the classification model, which allows the model to learn the basic semantic knowledge.

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. When generating query paths, we use two separate modules to generate candidate query paths. For one-entity question, we simply search the sub graph of the topic entity within two relation hops. While for questions of multiple entities, we generating query paths on the searching schema shown in Figure 3.
3.5 Artificial Data construction

For better predicting which class a question belonging to and alleviating the need of labeled training data, we generating substantial artificial questions by using a breadth-first-search (BFS) algorithm. In BFS, we randomly select a node entity in KB and extend a query path from the entity. When generating a 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 introduce in subsection 3.5. For instance, as for the above question, the corresponding query schema is \((x, r_1, t, t, r_2, e)\) where \(x\) represents the answer, \(r_1\) and \(r_2\) represent any relations in two hop query path extended from the topic entity \(t\) through a intermediate entity \(t\). We generate the artificial question by replace mentions of topic entities and relations with mentions of random selected node entities and correlated relations.

Since our predefined query schema contains semantic structure for both one-entity and multi-entity questions, our constructed samples can lead the pre-trained language model to converge at a direction which is more compatible with our specific classification task. Besides, the ration of questions based on 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 learn extra semantic structure of questions in two classes. In our experiment, we constructed 5k artificial questions and use them to train our classification model. With the help of pre-trained language model, our model can handle some questions that has never shown in training set. As the results in Sec.4 shown, with only 10% of training data, our model can achieve good performance in classifying the questions.

3.6 Beam Search

It is worth to notice that when extending multi-hop relations in the process of above two type questions, 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 (Sec.3.5) 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 below:

\[
\text{input} = [CLS]S_n[SEP]Q
\]

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 sub query path \(R_{n-1}\). The formulation is defined as:

\[
Sco(r_n|R_{n-1}) = BERTLayer(input)
\]

At each extending step, we only consider relations with \(Top_k\) score for further search, which significantly excluded some irrelevant query branches. The result in Sec.4.3.1 shows that, by adopting
the Beam Search algorithm, the accuracy of query paths generating remains competitive but the number of candidate paths decrease above 80%. The detailed description is seen in Algorithm 1.

**Algorithm 1** Multihop relation extraction. For each query scheme, we generate a set of candidate query paths $P^{(T)}$, where $T$ represents the hop number of the schema.

**Input:** $KB$, question $q$, topic entity set $E$, number of hops $T$

**Output:** $P^{(T)}$

1: Initialize: $P^{(0)} \leftarrow \{e_0 \in E\}$
2: for $t = 1, 2, \ldots, T$ do
3:     $\tilde{P}^{(t)} \leftarrow \phi$
4:     $\tilde{S}^{(t)} \leftarrow \phi$
5:     for each $p \in P^{(t-1)}$ do
6:         $e_{t-1} \leftarrow \text{tail}(p)$
7:         for each $(e_{t-1}, r, e_t) \in KB$ do
8:             if $e_t \in E$ then
9:                 $p' \leftarrow p \oplus (r, e_t)$
10:            else
11:                 $p' \leftarrow p \oplus (r)$
12:            end if
13:         $\tilde{P}^{(t)} \leftarrow \tilde{P}^{(t)} \cup \{p'\}$
14:         $\tilde{S}^{(t)} \leftarrow \tilde{S}^{(t)} \cup \{\text{Sentence}(p')\}$
15:     end for
16: end for
17: score all elements in $\tilde{S}^{(t)}$ and rank all corresponding elements in $\tilde{P}^{(t)}$
18: end for

4 Experiments

In this section, we analysis the performance of 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.

4.1 KB and Datasets

Our model use a open-domain KB PKUBase, which adopts Resource Description Framework (RDF) as their data format and contains billions of SPO (subject, predicate, object) triples.\cite{25}, as is shown in table 1. We train and evaluate our model on CCKS datasets, which contain 2298, 766, 766 pairs of questions.

| type           | triples | entity type | entity linking |
|----------------|---------|-------------|----------------|
| number of data | 61,006,527 | 25,182,627  | 13,930,117     |

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 topic entity.

As is shown in table 2, without $f_3$, the recall of multi-entity question surprisingly increased while accompanied with a sacrifice of accuracy for one-entity question. Samely, without $f_5$, the topic entity extracting accuracy for question of one topic entity increases, but the accuracy for multi-entity question drops. Moreover, excluding any of other features, the performance of Entity Linking model drops, which verifies their contribution for this module.
Table 2: Results of ablation experiments in Entity Linking Module.

| type     | one-entity | multi-entity |
|----------|------------|--------------|
| 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        |

4.3 Question Classification

In this module, we construct 5k artificial data based on the predefined query graph, 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 lead 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 0.05k, 0.5k and 3.75k manually constructed samples respectively. The result is illustrated in table 3.

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

| data          | train | valid | test |
|---------------|-------|-------|------|
| 10%+raw data  | 82.90 | 84.31 | 80.13|
| 10%+created data | 87.51 | 89.54 | 82.75|
| 50%+raw data  | 94.95 | 93.99 | 88.50|
| 50%+created data | 93.86 | 83.86 | 89.41|
| 100%+raw data | 97.39 | 95.42 | 88.76|
| 100%+created data | 99.09 | 95.45 | 91.11|

From the above table, we finds 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 a obvious improvement of the prediction on training datasets, which indicates 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, shown as figure 4.

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

4.3.1 Beam Search

For better presenting 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 baseline 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 sub query path filtered by the BS algorithm and keep all the two hop query paths. By setting different beam size, we can observe the influence on the recall and number of generated query paths.

![Figure 5: ratio of recall and path numbers when using different beam size.](image)

Figure 5 shows that a larger beam size will bring an increase on both recall and number of candidate query paths. Through further observation, we notice that the growth of the both indexes slow down. The retarded growth of recall is intelligible. Due to the existence of upper bound, as the beam size is large enough, the recall will infinitely 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 question. Thus, the remaining one-hop relations are usually more relevant to the semantic information in the primary question. As the figure shows, extended from a relation with lower semantic score, the second hop tend to generate fewer query paths. Since the relation whose tail has more triples may has 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.

### 4.3.2 Final Result

We evaluate our model in the CCKS2019 datasets, and compare our performance with a start-of-art model proposed by Cao et al. 2019.

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, Cao et al. use a PTLM model to calculate scores for generated query paths. Notably, their model introduce negative samples when training the Semantic Match model. Besides, introducing Bridging technology may harm the predicting performance of one-entity question, they adopt a Literal Match technology to rerank the generated query path. The table below shows the comparison of our models.

The result shows that our method is both 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.
Table 4: Comparative results between our best model with other model.

| Method                          | negative sample | Avg $F_1$  |
|---------------------------------|-----------------|-----------|
| Cao (baseline)                  | 3               | 56.70%    |
| Cao (Bridging)                  | 3               | 58.60%    |
| Cao (Bridging+literal match)    | 3               | 61.50%    |
| Cao (Bridging+literal match)    | 1               | 61.10%    |
|                               | 5               | 59.40%    |
| our model (with 10% data)       |                 | 58.54%    |
| our model (with 100% data)      |                 | 62.55%    |

5 Conclusion

This paper proposes a KBQA system equipped with pre-trained language model to handle multi-hop questions. We have shown that our model has the capability of answering multi-hop questions only 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.

References

[1] Dong L, Wei F, Zhou M, et al. Question answering over freebase with multi-column convolutional neural networks[C]//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). 2015: 260-269.

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

[3] Sun H, Dhingra B, Zaheer M, et al. Open Domain Question Answering Using Early Fusion of Knowledge Bases and Text[C]//Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing. 2018: 4231-4242.

[4] Sun H, Bedrax-Weiss T, Cohen W. PullNet: Open Domain Question Answering with Iterative Retrieval on Knowledge Bases and Text[C]//Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing: 2380-2390.

[5] Luo K, Lin F, Luo X, et al. Knowledge base question answering via encoding of complex query graphs[C]//Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing. 2018: 2185-2194.

[6] Gaurav Maheshwari, Priyansh Trivedi, Denis Lukovnikov, Nilesh Chakraborty, Asja Fischer, and Jens Lehmann. 2019. Learning to rank query graphs for complex question answering over knowledge graphs. In International Semantic Web Conference. Springer, 487504.

[7] Zhu S, Cheng X, Su S. Knowledge-based question answering by tree-to-sequence learning[J]. Neurocomputing, 2020, 372: 64-72.

[8] Yuncheng Hua, Yuan-Fang Li, Guilin Qi, Wei Wu, Jingyao Zhang, Daiqing Qi. Less is More: Data-Efficient Complex Question Answering over Knowledge Bases. arXiv preprint arXiv:2010.15881.

[9] Zi-Yuan Chen, Chih-Hung Chang, YiPei Chen, Jijnasa Nayak, and Lun-Wei Ku. UHop: An unrestricted-hop relation extraction framework for knowledge-based question answering. In NAACL, 2019.

[10] Yawei Sun, Lingling Zhang, Gong Cheng, and Yuzhong Qu. SPARQA: skeleton-based semantic parsing for complex questions over knowledge bases. In AAAI, 2020.

[11] S. Ou, C. Orasan, D. Mekhaldi, and L. Hasler. Automatic question pattern generation for ontology-based question answering. In FLAIRS, pages 183–188, 2008.

[12] C. Unger and P. Cimiano. Pythia: Compositional meaning construction for ontology-based question answering on the semantic web. In NLDB, pages 153–160. 2011
[13] C. Unger, L. Bühmann, J. Lehmann, A.-C. Ngonga Ngomo, D. Gerber, and P. Cimiano. Template-based question answering over rdf data. In WWW, pages 639–648, 2012.

[14] M. Yahya, K. Berberich, S. Elbassuoni, M. Ramanath, V. Tresp, and G. Weikum. Natural language questions for the web of data. In EMNLP, pages 379–390, 2012.

[15] L. Zou, R. Huang, H. Wang, J. X. Yu, W. He, and D. Zhao. Natural language question answering over rdf: a graph data driven approach. In SIGMOD, pages 313–324, 2014.

[16] 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 SIGMOD, pages 1809–1824, 2015.

[17] L. Zou, R. Huang, H. Wang, J. X. Yu, W. He, and D. Zhao. Natural language question answering over rdf: a graph data driven approach. In SIGMOD, pages 313–324, 2014.

[18] Talmor A, Berant J. The Web as a Knowledge-Base for Answering Complex Questions[C]//Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers). 2018: 641-651.

[19] Xu K, Lai Y, Feng Y, et al. Enhancing Key-Value Memory Neural Networks for Knowledge Based Question Answering[C]//Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers). 2019: 2937-2947.

[20] C. Liang, J. Berant, Q. Le, K. D. Forbus, N. Lao, Neural Symbolic Machines: Learning Semantic Parsers on Freebase with Weak Supervision, in: Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), vol. 1, 23–33, 2017.

[21] G. A. Ansari, A. Saha, V. Kumar, M. Bhambhani, K. Sankaranarayanan, S. Chakrabarti, Neural program induction for KBQA without gold programs or query annotations, in: Proceedings of the 28th International Joint Conference on Artificial Intelligence, AAAI Press, 4890–4896, 2019.

[22] D. Vrandeˇci´c, M. Krötzsch, Wikidata: a free collaborative knowledgebase, Communications of the ACM 57 (10) (2014) 78–5.

[23] K. Bollacker, C. Evans, P. Paritosh, T. Sturge, 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, 1247–1250, 2008.

[24] Talmor A, Berant J. The Web as a Knowledge-Base for Answering Complex Questions[C]//Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers). 2018: 641-651.

[25] Zhiheng Huang, Wei Xu, and Kai Yu. Bidirectional lstm-crf models for sequence tagging. arXiv preprint arXiv:1508.01991, 2015.

[26] Named Entity Recognition Using BERT BiLSTM CRF for Chinese Electronic Health Records

[27] Lex-BERT: Enhancing BERT based NER with lexicons

[28] Zi-Yuan Chen, Chih-Hung Chang, Yi-Pei Chen, Jijnasa Nayak, and Lun-Wei Ku. 2019. UHop: An unrestricted-hop relation extraction framework for knowledge-based question answering. In Proceedings of the NAACL-HLT, pages 345–356.

[29] Yunshi Lan, Shuohang Wang, and Jing Jiang. 2019b. Multi-hop knowledge base question answering with an iterative sequence matching model. In Proceedings of the ICDM, pages 359–368.

[30] https://conference.bj.bcebos.com/ccks2019/eval/webpage/pdfs/eval_paper_6_4.pdf

[31] Bast H, Haussmann E. More accurate question answering on free-base[C]//Proceedings of the 24th ACM International on Conference on Information and Knowledge Management. 2015: 1431-1440.

[32] Oren Etzioni, Michael Cafarella, Doug Downey, Ana-Maria Popescu, Tal Shaked, Stephen Soderland, Daniel S Weld, and Alexander Yates. 2005. Unsupervised named-entity extraction from the web: An experimental study. Artificial intelligence, 165(1):91–134.
[33] Michael Collins and Yoram Singer. 1999. Unsupervised models for named entity classification. In 1999 Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora.

[34] GuoDong Zhou and Jian Su. 2002. Named entity recognition using an hmm-based chunk tagger. In proceedings of the 40th Annual Meeting on Association for Computational Linguistics, pages 473–480. Association for Computational Linguistics.

[35] Robert Malouf. 2002. Markov models for language-independent named entity recognition, proceedings of the 6th conference on natural language learning. August, 31:1–4.

[36] Yu Sun, Shuohuan Wang, Yukun Li, Shikun Feng, Xuyi Chen, Han Zhang, Xin Tian, Danxiang Zhu, Hao Tian, and Hua Wu. 2019. Ernie: Enhanced representation through knowledge integration. arXiv preprint arXiv:1904.09223

[37] Rakesh Chada, Pradeep Natarajan. 2021. FewshotQA: A simple framework for few-shot learning of question answering tasks using pre-trained text-to-text models. arXiv preprint arXiv:2109.01951

[38] Yuncheng Hua, Yuan-Fang Li, Gholamreza Haffari, Guilin Qi, Tongtong Wu. 2020. Few-Shot Complex Knowledge Base Question Answering via Meta Reinforcement Learning. arXiv preprint arXiv:2010.15877