Question Answering over Knowledge Graphs via Machine Reading Comprehension

Weidong Han, Zhaowu Ouyang, Yifan Wang, and Weiguo Zheng(✉)

School of Data Science, Fudan University, Shanghai, China
{wdhan20,ycwou20,zhengweiguo}@fudan.edu.cn, yifan_wang21@m.fudan.edu.cn

Abstract. Due to the representation gap between unstructured natural language questions and structured knowledge graphs (KGs), it is challenging to answer questions over KGs. The existing semantic parsing-based methods struggle for building structured queries that can be executed over the KG, and thus they are difficult to cover diverse complex questions. The information retrieval-based methods suffer from poor interpretability. In this paper, we present a novel approach powered by machine reading comprehension. To transform a subgraph of the KG centered on the topic entity into text, we sketch the subgraph through a carefully designed schema tree, which facilitates the retrieval of multiple semantically-equivalent answer entities. Instead of seeking answers from all the automatically generated paragraphs, we pick out the promising paragraphs containing answers by a contrastive learning module. Finally, it is straightforward to deliver the answer entities based on the answer span that is detected by the machine reading comprehension module. The results on benchmark datasets demonstrate that our method achieves significant improvement compared with the existing methods.

Keywords: Knowledge Graphs · Question Answering · Machine Reading Comprehension · Schema Tree

1 Introduction

By modeling the complex relationships among varieties of entities, knowledge graphs (shorted as *KGs*) have been widely used in many real-world applications, such as recommendation [30], fraud detection [33], and cyber security [18]. Generally, a knowledge graph is a structured repository that contains a collection of facts in the form (subject, relation, object). Knowledge graph question answering (KGQA) is an important task that aims at answering natural language questions based on a given knowledge graph, providing an easy and intuitive way to query knowledge graphs. Currently, how to understand and answer complex questions remains a challenging problem as they are often grounded to multiple facts in the underlying KG. For example, "Who won the prize at the spin-off of

X. Wang et al. (Eds.): DASFAA 2023, LNCS 13944, pp. 577–594, 2023.
https://doi.org/10.1007/978-3-031-30672-3_39

the 1885 Wimbledon Championships-Gentlemen's Singles?" is a multi-hop question involving two facts *(the 1885 Wimbledon Championships, spin-off, ?x)* and *(?obj, win, ?x)*. Moreover, the notorious ambiguity and variability of natural language further increase the difficulty of KGQA.

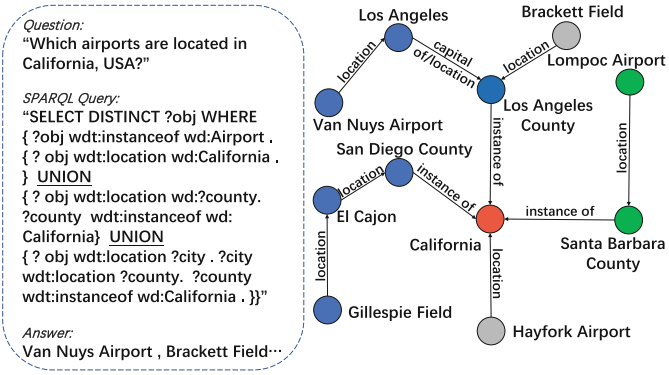

Fig. 1. An example question whose SPARQL query has multiple UNION operators. A subgraph of Wikidata is presented, where the red node denotes topic entity. (Color figure online)

The existing methods developed for KGQA can be roughly divided into three groups, i.e., rule-based/template-based (RT-based) methods, semantic parsing-based (SP-based) methods, and information retrieval-based (IR-based) methods. RT-based methods [34] exhibit superiority in precision but fail to handle the flexible and varied representation of the same semantic meanings. Existing SP-based methods try to transform natural language question q into symbolic logic form, e.g., SPARQL, which can be executed against the knowledge graph to get answers to q [6,12]. However, the existing SP-based methods are difficult to cover diverse complex queries (e.g., multi-hop reasoning, constrained relations, and numerical operations) [15]. For instance, let us consider the question in Fig. 1. To answer this question, the system should generate a complex SPARQL that is composed of three different basic graphs combined by two "UNION" operators. IR-based methods regard the process of finding answers as a classification task [21,24]. Most methods adopt deep neural networks like GNN to learn entity embeddings and rank candidate entities. One major challenge of these methods is lacking interpretability because they only take the ranking scores as objective. Moreover, the IR-based methods are not effective to deal with the multi-answer or aggregation questions as it is not known how many entities will be involved in the answers. For example, the question in Fig. 1 has 205 answers in total. It is difficult to predict the number of answers precisely.

To address the drawbacks of the existing methods, we propose a novel framework powered by machine reading comprehension (shorted as MRC) in this paper. Benefiting from the powerful pre-trained language model (PLM), MRC

is promising to find the answers from plain text without complex structured queries. Thus we need to transform the knowledge graph or its subgraph into text (KG2Text). However, generating text for triples is a time-consuming task. To facilitate KG2Text, we present an effective approach based on a newly designed structure, namely *k-hop schema tree*, which sketches the subgraph through *aggregation nodes*. Meanwhile, it is natural to support the retrieval of multiple semantically-equivalent answers. Since generating text for irrelevant triples not only leads to noisy information but also is time-wasting, it is required to rule out the triples that are not relevant to the question. To the end, we perform relation linking to reduce the search space by excluding the paths that do not contain any desired relations. If the refined schema tree is still too large, multiple paragraphs will be generated for its decomposed subtrees respectively. For the automatically generated paragraphs, contrastive learning is applied to pick the target paragraph containing the answers. Powered by MRC, knowledge graph reasoning is regarded as the natural language understanding (NLU) task to obtain the aggregation answer that can be traced back to find all the answers along the schema tree. The contributions of the paper are summarized as follows:

- We develop a novel approach to KGQA based on MRC, migrating the KG reasoning to an NLU task that benefits from PLMs.
- We propose the schema tree to sketch the subgraph through aggregation nodes, facilitating KG2Text and supporting the retrieval of multiple answers.
- Contrastive learning is invoked to pick the target paragraph from all the generated paragraphs, over which MRC is conducted to find the answers.
- Empirical studies on benchmark datasets have demonstrated the effectiveness of the proposed method.

2 Problem Definition and Preliminary

2.1 Problem Definition

Definition 1 (Knowledge Graph). *A knowledge graph (KG), denoted by* $\mathcal{G} = (\mathcal{E}, \mathcal{L}, \mathcal{R})$*, is a directed graph consisting a set of triples* (h, r, t)*, where* $\mathcal{E}$*,* $\mathcal{L}$*, and* $\mathcal{R}$ *represent the set of entities, literals, and relations (including predicates and properties), respectively. Specifically,* $h \in \mathcal{E}$*,* $t \in \mathcal{E} \cup \mathcal{L}$*, and* $r \in \mathcal{R}$ *represents the relation between two entities* h *and* t*.*

Definition 2 (k-hop Path Tree). *Given an entity* $e \in \mathcal{E}$*, the* k*-hop path tree rooted at* e*, shorted as* k*-PT*(e)*, is a tree formed by combining all* k*-hop sequence of triples starting from* e*.*

Note that the "path" in the k-$PT(e)$ ignores the direction of triples. k-$PT(e)$ can be constructed through the extended breadth-first search by ignoring the direction and allowing the reusing of the nodes.

Definition 3 (Aggregation Node). *Given a* k*-PT*(e)*, several entities are grouped as an aggregation node if they share the same father node with the identical edge labels.*

An aggregation node actually clusters the entities that act the same semantic role, making it feasible to return multiple semantically-equivalent answers. Specially, a single node in k-$PT(e)$ can be viewed as an aggregation node if it cannot be grouped with other nodes.

Definition 4 (k-hop Schema Tree). *Given an entity $e \in \mathcal{E}$, the k-hop schema tree rooted at e, shorted as k-$ST(e)$, is a tree in which the starting node of each edge is an entity/aggregation node and each ending node is an aggregation node.*

Example 1. For the subgraph presented in Fig. 2(a), the corresponding 2-hop path tree rooted at the entity "California" is shown in Fig. 2(b). Figure 2(c) depicts the 2-hop schema tree, where the aggregation node in the second layer groups entities Hayfork Airport and Lake Tahoe Airport.

Definition 5 (Topic Entity). *Given a question q, its topic entity is the entity $e \in \mathcal{E}$ that is linked to a mention in q.*

Problem Statement 1. *Given a question q and a knowledge graph $\mathcal{G}$, the task is to find answers to q from the knowledge graph $\mathcal{G}$.*

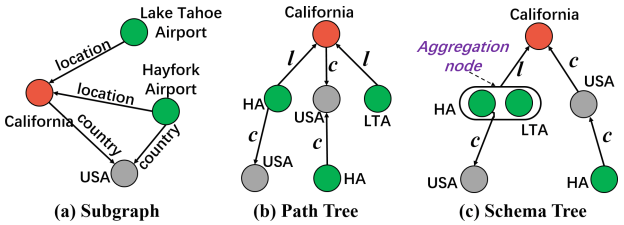

Fig. 2. An illustration of path tree and schema tree, where l denotes location, c denotes country, HA denotes Hayfork Airport, and LTA denotes Lake Tahoe Airport.

2.2 Preliminary

KG2Text. Given a subgraph g in $\mathcal{G}$, KG2Text aims at generating high-quality sentences to represent g. Recently, researchers mainly use encoder-decoder neural networks with attention [25,29] to address this problem. In this paper, we use the KG2Text tool JointGT [13] which presents three pre-training tasks on an encoder-decoder framework and a structure-aware self-attention layer.

Contrastive Learning. Contrastive Learning pulls semantically similar pair close and keeps dissimilar data away in continuous space, by using self-supervision signals. Inspired by SimCSE [8], we collect triples (q, p^+, p^-), where p^+ is called a positive instance corresponding to a paragraph containing the answers and p^- is a negative instance corresponding to a paragraph without

any answer. Thus, the model picks out the sentences which are more likely to contain the final answers to the question.

Machine Reading Comprehension. Machine reading comprehension (MRC) requires a machine to answer questions based on a given textual context. It has attracted increasing attention with the incorporation of various deep-learning techniques over the past few years. The performance of the MRC task highly depends upon the ability to understand natural language for a machine. We notice that any MRC model could be applied to our framework and we choose a simple MRC model [26] in our experiments.

3 Overview of Our Approach

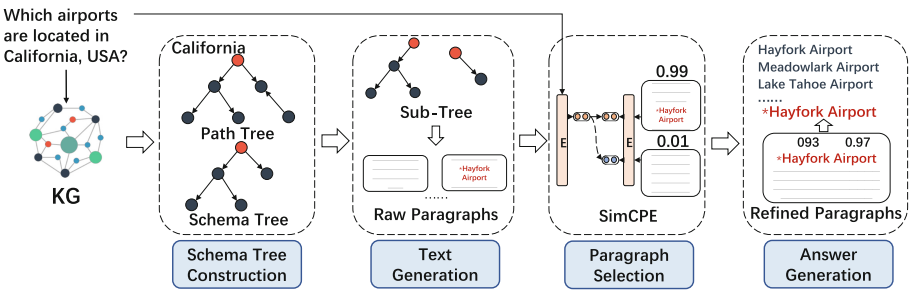

Fig. 3. Architecture of the proposed approach, consisting of four major steps. Red node indicates the topic entity. (Color figure online)

As depicted in Fig. 3, we propose a novel approach to KGQA, consisting of four components, i.e., schema tree construction, text generation, paragraph selection, and answer generation via MRC.

Schema Tree Construction. Motivated by fact that the answers to a question share the identical entity type and they appear in semantically-equivalent structures in the KG to represent the input question, we propose a novel algorithm to group and merge the semantically-equivalent nodes in a carefully designed *schema tree*. Specifically, for a question q, we extract the k-hop path tree rooted at the topic entity e_0 from the knowledge graph, aggregate the child nodes of one parent node connected by the same relation into an aggregation node, and refine the tree. In this way, we get a new subtree rooted at e_0, named schema tree, which simplifies the subgraph while preserving all necessary structure information in the knowledge graph.

Text Generation. The structure information of knowledge graphs is more difficult to understand and deal with for a computer than serialized text. It motivates us to generate texts for the built schema tree by employing KG2Text models and

retrieve answers from the generated texts. On one hand, the tree structure almost complies with text writing. On the other hand, the generated text is more concise benefiting from the aggregate nodes, removing the repeated and redundant paths. Besides, just like the article describing objects from different aspects in segments, if the schema tree $k\text{-}ST(e)$ is too large, we will decompose it into smaller subtrees and generate a raw paragraph for each of them respectively.

Paragraph Selection. Since the amount of the generated texts from the knowledge graph may be still huge even if pruned by extracted relations from the question. Hence, we need to further filter out the unpromising generated texts. Motivated by *SimCSE*, we develop a paragraph selection module *SimCPE* to pick out the target paragraphs. In training stage, for each question q_i, we use the paragraph containing answers as a positive instance p_i^+. The paragraph having no answers or that is borrowed from the paragraphs of other questions will be taken as a negative instance p_i^-. Thus the triples (q_i, p_i^+, p_i^-) for contrastive learning can be collected to fine-tune the model. In test stage, we choose those paragraphs whose similarity to q_i goes beyond a predefined threshold.

Answer Generation via MRC. To find answers from the target paragraphs, we resort to the MRC model [26], because 1) the answers to questions are mostly nodes in schema tree, thus can be extracted from generated text. This process can be regraded as a sequence tagging problem, which is exactly what the MRC model elaborates. 2) the model can solve the questions which need arithmetic operations or aggregate functions, thus improving the performance. Specifically, we use this MRC model to find answer spans from the candidate texts. Once the aggregate nodes are labeled as answer spans, the final answers can be delivered straightforwardly by reporting the entities clustered in the aggregation node.

Advantages of the Proposed Framework. Compared to previous methods, our approach exhibits the following advantages:

(1) The novel schema tree has achieved lossless simplification of semantically-equivalent entities, which can alleviate the burden of text generation and improve the quality of generated texts.
(2) With the aggregation node, we can return an answer set that may contain multiple entities at once, thus overcoming the drawback of IR-based methods that they do not know how many entities will be involved in the answers.
(3) We can anchor the schema tree that generates the text containing answers, guaranteeing the interpretablity of our method.
(4) Our method utilizes the semantic correlation with the question to retrieve subgraphs from the underlying knowledge graph and finds answers in generated text, thus can handle the question with multiple answers containing different subgraph patterns respectively, which is difficult to address for both IR-based methods and SP-based methods.

Algorithm 1. Schema Tree Construction

Require: a natural language question q, a topic entity e_0, knowledge graph $\mathcal{G}$;
Ensure: a schema tree $k\text{-}ST(e_0)$.

```
1: D ←triples within k hops from e_0, Ents ←all entities in D
2: Dict ←create an empty dict, an index structure where key is entity and value is
   (relation, aggregate node) pair
3: for ent in Ents do
4:    D_ent ←triples with one-hop relation of ent from D
5:    for each relation r_i in Adjacent(ent) do
6:       agg_ent ← merge all tail entities adjacent ent
7:       add (ent, r_i, agg_ent) into Dict
8: k-ST(e_0) ←empty tree rooted at e_0
9: frontier ← Queue(e_0)
10: while not IsEmpty(frontier) do
11:    if depth < k then
12:       ent_p ←pop frontier
13:       tmp ←retrieve {(r', aggnodes')} from Dict with key ent_p
14:       for r' and aggnodes' in tmp do
15:          k-ST(e_0) ← add an edge labelled with r' from ent_j to aggnodes'
16:          for node in aggnodes' do
17:             if node is not the same with its ancestor then
18:                add node to frontier
19: return k-ST(e_0)
```

4 Text Generation

4.1 Schema Tree Construction

Given a natural language question q as input, we first obtain the topic entity e_0 by using the existing method, e.g., spciy-entity-linker[1]. Then we extract and sketch the subgraph centered on the topic entity e_0 from KG. Algorithm 1 outlines the process of constructing a schema tree.

For each entity $e \in \mathcal{G}$, it may have outgoing and incoming edges as $\mathcal{G}$ is a directed graph. For ease of presentation, they are all called the edges starting from the entity e in the following sequel. We extract the set D of triples within h hops from e_0 and merge adjacent edges of the same relation, as shown in Algorithm 1 (lines 4–7). Then we merge tail entities with the same relation r starting from each $e \in D$ via breadth-first search, as shown in Algorithm 1 (lines 10–18). The combined entities form an aggregation node, taking one of the entities as its label. A special mark "*" can be assigned to indicate the role of the aggregation node. For example, as shown in Fig. 2, for the entity California, we merge the tail entities connected by the relation "location" into an aggregation node named "*Hayfork Airport". We can merge entities to form aggregation nodes iteratively. Finally, the schema tree $k\text{-}ST(e_0)$ is constructed.

[1] https://github.com/egerber/spaCy-entity-linker.

Algorithm 2. Schema Tree Refinement

Require: a schema tree k-$ST(e)$, a question q;
Ensure: a refined schema tree k-$ST(e)$.
1: $R_l \leftarrow$ invoke relation linking methods for q, $R \leftarrow \{\}$
2: **for** each path P_j in k-$ST(e)$ **do**
3: $R_p \leftarrow$ relations in path P_j
4: **if** $\exists rel \in R_p, rel \in R_l$ **then**
5: $R \leftarrow R \cup R_p$
6: k-$ST(e) \leftarrow$prune k-$ST(e)$ and maintain relations in R
7: **return** k-$ST(e)$

To prevent introducing too many noisy triples, we perform relation linking-based path refinement as shown in Algorithm 2. Any relation linking methods, e.g., Falcon2.0 [23] and GenRL [22], can be applied to the question. From the topic entity e_0, we iteratively examine each path of the schema tree k-$ST(e)$ and maintain the path if and only if it contains a relation in the relation list R_t (line 2–7). We update the schema tree and remove the relation out of list R (line 9). In other words, we discard the path or sub-path which does not contain any relation in the linked relations. Though list R_l may do not contain golden relation r_{gold}, the schema tree could include r_{gold} via the other relation in R_l in the same path. In the end, we obtain a refined schema tree k-$ST(e)$.

4.2 Text Generation

In this process, we transform a schema tree into a set of paragraphs. JointGT is employed in our task. Since the number of triples in the whole schema tree k-$ST(e_0)$ is beyond model capacity, we decompose k-$ST(e_0)$ into subtrees $\{\mathcal{T}_i\}_{i=1}^m$ based on relations $\{r_i\}_{i=1}^m$, $r_i \in \mathcal{R}_0 = \{r|(e_0, r, e_{1r}) \in k\text{-}ST(e_0)\}$:

$$\mathcal{T}_i = (e_0, r_i, e_{1r_i}) \cup \mathcal{T}'(e_{1r_i}),$$

where $\mathcal{T}'(e_{1r_i})$ is the subtree rooted at e_{1r_i}. Then, we linearize $\mathcal{T}_i$ into a sequence $\mathcal{T}_{linear} = \{\omega_1, \omega_2, \ldots, \omega_n\}$ consisting of n tokens. For instance, "$(A, r_1, B), (B, r_2, C)$" is transformed into "$< H > A < R > r_1 < T > B < H > B < R > r_2 < T > C$". We put $\mathcal{T}_{linear}$ into the encoder-decoder framework with a structure-aware self-attention layer proposed by JointGT framework: $x_{\mathcal{T}_i} = \text{Encoder-Decoder}(\mathcal{T}_{linear})$, where $x_{\mathcal{T}_i}$ is the representation of subtree $\mathcal{T}_i$. Then, we use $x_{\mathcal{T}_i}$ to generate paragraph corresponding to subtree $\mathcal{T}_i$.

5 Answer Selection

To retrieve answers from the text for the question, we first determine the target paragraphs in Sect. 5.1, based on which the final answers are generated powered by machine reading comprehension models in Sect. 5.2.

5.1 Paragraph Selection

Although we have refined the schema tree in the previous step, it is inevitable to generate irrelevant paragraphs that correspond to some subtrees $\mathcal{T}_i$. Thus, we propose a Simple Contrastive Learning of Paragraph Embeddings, shorted as **SimCPE**, to pick out the most promising paragraphs that contain the answers.

As shown in Fig. 4, for each question q_i we use the paragraph which contains answers as positive instance p_i^+ and the paragraph which does not contain answers or the paragraph for other questions as negative instance p_i^-, thus obtaining triples (q_i, p_i^+, p_i^-). We apply RoBERTa$_{large}$ framework as a basic model and try to narrow the distance between q_i and p_i^+ and keep q_i and p_i^- away in continuous space. We minimize the objective $\mathcal{L}_i$ loss function:

$$-\log \frac{e^{\mathrm{sim}(q_i,p_i^+)/\tau}}{\sum_{j=1}^{N}(e^{\mathrm{sim}(q_i,p_j^+)/\tau} + e^{\mathrm{sim}(q_i,p_j^-)/\tau})},$$

where the similarity measure $\mathrm{sim}(x_1, x_2)$ is cosine similarity $\frac{x_1^T x_2}{||x_1||||x_2||}$, N is mini-batch size and τ is a temperature hyperparameter. In the test phase, we input previously generated paragraphs, output the probability of containing answers for each paragraph, and deliver the k paragraphs with the highest probability.

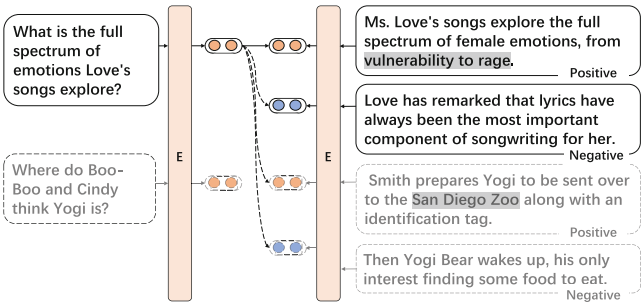

Fig. 4. Examples for contrastive learning. Orange circles denote representations of the question and ground truth text, and blue denotes noisy text. **E** is the encoder.

5.2 Answer Generation via MRC

Since an answer span may appear multiple times in the automatically generated paragraphs, we present a multi-span framework to tackle multi-span questions inspired by [26]. The paragraph containing at least one answer is assigned the possible-correct tagging. At training time, we maximize the marginal probability of possible-correct taggings:

$$\log p(\mathcal{S}|h) = \log \sum_{S \in \mathcal{S}} (\prod_{i=1}^{m} p_i[S_i]),$$

$$p_i = \mathrm{softmax}(f(h_i))$$

where h is the representation of question and text generated by KG2Text module via BERT [4], f is a parameterized function, $\mathcal{S}$ is the set of possible-correct taggings and m is length of text. At test time, we output the tagging with the max probability: $\hat{S} = \arg\max_{S \in \mathcal{V}} \prod_{i=1}^{m} p_i[S_i]$, where $\mathcal{V}$ includes all possible taggings as for `IO` tags. We predict for each token whether it is part of the answer individually following the paper [4]. Then, we collect the predicted spans as answers. If the span contains the special mark "*" indicating an aggregation node, we can collect all the semantically-equivalent answers in the corresponding aggregation node. Meanwhile, the chain starting from e_0 to the detected aggregation node in the schema tree can be taken as the inferential chain of evidencing answers.

6 Experimental Evaluation

In this section, we evaluate the proposed method through extensive experiments and report the results.

6.1 Experimental Settings

In our experiments, we made the next settings. (1) **JointGT**: We choose JointGT(Bart), and fine-tune on WebNLG following [13]. (2) **MRC**: We choose TASE$_{IO}$+SSE as a base model, train on **Drop** following [26], and fine-tune on our datasets. (3) **Depth** k: Considering the complexity of datasets, k is set to 2.

6.2 Benchmark Datasets

Table 1. Statistics of datasets.

Datasets	Train	Dev	Test
LC-QuAD2.0	17,413	4,353	6,046
WebQSP	2,848	250	1,639

Table 1 lists the statistics of the used datasets.

WebQuestionsSP (WebQSP). [32] contains 4737 question-answer pairs with Freebase as the KG. We use the same splits and pre-processing as [24].

LC-QuAD 2.0 [7] is a larger dataset based on Wikidata. The paraphrases and SPARQL queries are compatible with both Wikidata and DBpedia-2018.

6.3 Baselines

We choose the following methods as the baselines:

QAnswer [5]: QAnswer is a system that relies on the simple template-based SPARQL queries that are then ranked and executed to get the answer.

Platypus [19]: Platypus is designed as a QA system driven by NLU, using grammar rules and template-based techniques for complex questions.

GRAFT-Net [28]: Graphs of Relations Among Facts and Text Networks (GRAFT-Net) classifies nodes in subgraphs containing KG entities and text.

PullNet [27]: PullNet uses self-learning to extract the subgraph related to the question. It combines the heterogeneous information, updates the subgraph, and finally uses GCN to perform representation learning on nodes.

UNIQORN [20]: UniQORN copes with questions by advanced graph algorithms for Group Steiner Trees, identifying the best answer candidates.

EmbedKGQA [24]: EmbedKGQA uses neutral network to get the representation of question and uses ComplEx embeddings of entities. Then it computes the similarity between question and candidate entities.

NSM [10]: NSM uses a teach-student framework to generate and learn supervision signal, thus enhancing the performance of reasoning against the KG.

6.4 Main Results

The results of our method and baselines are shown in Table 2. We can find that:

(1) The experimental results indicate that the existing methods, e.g. GRAFT-Net and PullNet, have poor performance on LC-QuAD2.0, while our approach achieves state-of-the-art performance on LC-QuAD2.0 and is much better than the above baselines. PullNet trains an effective subgraph retrieval module based on the shortest path between topic entities and answer entities. However, PullNet uses external corpus to enhance performance. UniQORN uses Group Stein Tree (GST) to reason against Wikidata, but ineffective to cover multi-hop queries with constraints. Our method exhibits a powerful ability to extract information from Wikidata that contains thousands of relations and billions of entities.

Table 2. Results on LC-QuAD2.0 and WebQSP. We copy the results on LC-QuAD2.0 in [20] and copy the results on WebQSP in [10].

Methods	LC-QuAD2.0		WebQSP
	Hits@1	Hits@5	Hits@1
QAnswer	–	31.8	–
Platypus	–	10.9	–
GRAFT-Net	26.5	41.8	66.4
PullNet	11.9	28.1	68.1
UniQORN	25.2	41.4	–
EmbedKGQA	–	–	66.6
NSM	–	–	74.3
ours	**42.6**	**53.7**	**75.2**

Table 3. Adaptive Ability of MRC model (Hits@1).

Methods	WebQSP
vanilla model (no fine-tune)	62.2
+ LC-QuAD2.0 train set	66.0
+ WebQSP train set	**75.2**

(2) Our method also achieves state-of-the-art performance on WebQSP. EmbedKGQA uses ComplEx embeddings to obtain the representation of entities and computes the similarity between the question and entities, ignoring the path information. NSM shows a competitive performance as it applies a teacher-student framework to learn intermediate supervision signals. However, it highly depends on the golden path, and is difficult to find all the semantically-equivalent answers. In contrast, we sketch the structure around the topic entity by schema trees and fit the refined paragraphs that are generated from decomposed schema subtrees into a multi-span MRC model. Thus, we can answer questions that even do not have explicit paths via natural language understanding.

6.5 Detailed Analysis

Adaptive Ability. We emphasize the importance of our proposed framework based upon MRC to select answers in Sect. 3. To further study its adaptive ability, we perform experiments to compare two variants including: (1) +LC-QuAD2.0 train set (fine-tune) using question-answer pair on LC-QuAD2.0 as train set. (2) +WebQSP train set (fine-tune) using WebQSP to fine-tune MRC model. As shown in Table 3, our proposed MRC model exhibits a certain inference ability even without fine-tuning. It has significant improvement when fine-tuned on the corresponding dataset. Thus, our framework based on MRC shows promising performance in adapting to other question-answering datasets.

Effect of KG2Text. To study the effect of KG2Text, we compare the performance of *w/o KG2Text* (without using KG2Text model to generate texts, concatenating the triples in schema tree instead) with our complete model equipped with the KG2Text module. The experimental results are shown in Table 4 (line 1 and line 3). We can see that the performance on both datasets decreases, indicating the contribution of KG2Text module. However, there is a significant difference on effect of this module on two datasets. Without KG2Text module, the decline on LC-QuAD2.0 is particularly severe, nearly three forth. While on WebQSP, the decline is relatively small, only 6.0%. The questions of LC-QuAD2.0 are more difficult to understand for the machine as they correspond to more complex subgraphs and more fluent and logical text. Hence, when the text is simply concatenated by strings, it is almost impossible for the model to answer the complex questions.

Table 4. Ablation study (Hits@1).

Methods	LC-QuAD2.0	WebQSP
fine-tuned model	**42.6**	**75.2**
w/o SimCPE	19.45	66.5
w/o KG2TEXT	10.77	69.18

Table 5. Effect of depth k (Hits@1).

Methods	LC-QuAD2.0	WebQSP
$k=1$	20.5	34.3
$k=2$	**42.6**	**75.2**

Effect of SimCPE. To study the effect of SimCPE, we compare *w/o SimCPE* (fine-tune MRC model without SimCPE) with our fine-tuned approach on LC-QuAD2.0 and WebQSP. As shown in Table 4 (lines 1–2), the performance on both datasets decreases when disabling SimCPE, which proves the efficiency and necessity of SimCPE. However, the degree of decline in two datasets has a different picture. Specifically, removing SimCPE causes more than half loss in Hits@1 on LC-QuAD2.0 dataset, while a slight decrease of 8.7% in WebQSP. The main reason for this result may be resulted from the size of the corresponding knowledge graph. The knowledge graph LC-QuAD2.0 is very large, thus the generated subgraph is also very large as well. With the help of SimCPE, we can quickly eliminate the unrelated texts and improve the accuracy. As for WebQSP, the knowledge graph is relatively small, so the truncation effect of SimCPE is much smaller than the former.

Effect of Depth k. We further study the influence of depth k. The results are reported in Table 5. Obviously, the Hits@1 drops sharply as there are many two-hop questions on both datasets. It reveals that equipped with higher-order relations our approach can handle more complex questions demanding multi-hop reasoning. Besides, since most questions have 2-hop structures in the knowledge graph, we set k to 1 and 2 in the experiments.

6.6 Case Study

As shown in Table 6, we provide several cases in LC-QuAD2.0 to further investigate and analyze our proposed approach.

Table 6. Case study on LC-QuAD2.0, where * means this entity is an aggregation node with its members in parentheses.

Case	Id	Question	Generated Text	Returned Answers	Ground Truth
1	14791	What has influenced the sculptors of Man in Shower in Beverly Hills?	The creator of Man in Shower in Beverly Hills is David Hockney, who was influenced by *Francis Bacon (Francis Bacon,Pablo Picasso).	Francis Bacon Pablo Picasso	Pablo Picasso Francis Bacon
2	9652	What are the names of Keira Knightley's sibling and father?	The sister of Keira Knightley is Caleb Knightley who is also the son of Will Knightley.	Caleb Knightley Will Knightley	Caleb Knightley Will Knightley
3	27990	Who is the curator of São Paulo Museum of Art?	The architect of São Paulo Museum of Art, which is located in Sao Paulo, Brazil, was Lina Bo Bardi. It is the home of the architect, Adriano Pedrosa and the museum 's director, Heitor Martins.	Adriano Pedrosa Heitor Martins	Adriano Pedrosa
4	20699	Which is the rock band, member of which was Tom Petty?	Tom Petty is a member of Tom Petty and the Heartbreakers, a band that includes the band, Traveling Wilburys.	Tom Petty and the Heartbreakers Traveling Wilburys	Tom Petty and the Heartbreakers
5	25382	Name an English written daily newspaper that starts with letter "T".	*the New York Times (The New York Times, Daily Times, ...) are all owned and operated by the company Nagaland Post.	The New York Times Daily Times ...	The Times... The Dallas Morning News The Guardian

Both Case 1 and Case 2 are successful cases. The former shows that our approach can process questions with many answers. By merging all entities that influence the entity "David Hockney" into an aggregation node, the model is able to find all the answers. The latter one illustrates that our method can handle the complex question with multiple special interrogative words. Powered by the MRC model that targets at processing multi-span questions, our approach can find the sibling and father of "Keira Knightley" simultaneously.

Case 3 and Case 4 are both surprising cases. The answers are partially correct because some answers do not belong to ground truths. The main reason is that our method retrieves all relations related to the question, and the MRC model fails to distinguish "curato" from "manager" (or "rock band" from "musical group") quite clearly. However, from a perspective of real-world applications, the result may be pretty good. In most cases, the user is prone to use common words rather than professional words as she may not know the vocabulary of the huge knowledge graph. Moreover, it demonstrates that our methods can solve questions with semantically-equivalent subgraphs.

Case 5 is a failed case where our approach fails to handle character-level constraints. It returns all English written daily newspapers without considering the constraint "starts with letter 'T"'. Although the MRC model can distinguish them in the text by dealing with the nodes in aggregation nodes separately, it is overwhelmed when encountering character-level constraints. That is because all character-level information has been hidden by aggregation nodes and is invisible in generated texts. If we present all entities in the aggregation node to the MRC model, the text will be long-winded for the model to understand.

7 Related Work

KGQA: Zheng et al. apply template-based methods including two steps, generating templates offline and understanding questions online [34]. Existing SP-based methods [2,6,12,17,31] try to transform natural language questions into a symbolic logic form which can be executed over the KG to return answers. Ding et al. introduce frequent query substructures to rank existing query structures or build new queries [6]. Kapanipathi et al. present a system that successfully utilizes a generic semantic parser, particularly AMR, to deal with a KGQA task for the first time [12]. Chen et al. propose a two-stage formal query building approach that automatically predicts the query structure and uses it as a constraint to avoid generating noisy candidate queries [2]. Liang et al. present a novel approach that can first identify the type of each question by training a Random Forest model and use it to guide different processes to generate SPARQL queries [17]. IR-based methods [3,10,11,21,24] regard answer selection as a classification task. Most studies use deep neural networks like GNN to score candidate entities. Qiu et al. introduce reinforcement learning to formulate multi-relation question answering as a sequential decision problem [21]. The proposed model performs an efficient path search on the knowledge graph to obtain answers and utilizes beam search to significantly reduce the number of candidates. Meanwhile, based on attention mechanism and neural network, policy network can enhance the unique influence of different parts of a given question on triple choice. He et al. propose a teacher-student framework to find a reasonable path via supervision signals at intermediate steps [10]. The main student model learns how to find answers to specific questions, while the teacher model strives to learn intermediate state supervision signals.

KG2Text: KG2Text is an important problem that generates natural language sentences from structured facts in the knowledge graph. Traditional methods [14] mainly focus on rule-based algorithms but subject to low adaptability. [1] proposes seq-to-seq and graph-to-seq framework to generate sentences from well-structured data and a large dataset KGText. KGText is a new pretrained corpus in which English sentences from Wikipedia are aligned with subgraphs from Wikidata for a total of about 1.8 million (subgraph, text) pairs. The method ensures that each subgraph and its paired sentences describe nearly the same facts. Based on that, [13] adds a structure-aware self-attention layer to better instruct sentence generation and proposes three pre-training tasks, including graph enhanced text reconstruction, text enhanced graph reconstruction, and graph-text embedding alignment, to explicitly promote the graph-text alignment.

MRC: Machine Reading Comprehension (MRC) requires a machine to answer questions based on a given textual context. Some pre-trained language models, like Bert [4] and BART [16], are trained on a large-scale corpus and fine-tuned on the downstream dataset, showing impressive performance on MRC. As for long-text, [9] proposes a model that learns to chunk text more flexibly via reinforcement learning and decides the next segment to process. However, forcing

answers to a single span limits its ability to understand questions as some recent datasets contain multi-span questions, i.e., questions whose answers are a set of non-contiguous spans in the text. Naturally, models that return a single span cannot answer these questions. To answer multi-span questions, [26] proposes a new architecture to cast the problem as a sequence tagging task. In other words, the model predicts whether each token is part of an output.

8 Conclusion

We present a novel framework for KGQA based on machine reading comprehension. To facilitate KG2Text, we propose schema tree that sketches the subgraph centered on the topic entity. The search space is reduced by removing the irrelevant triples and paragraphs through relation linking and constrastive learning. The empirical results show that our approach outperforms the existing methods.

Acknowledgement. This work was supported by National Natural Science Foundation of China (Grant No. 61902074). Weiguo Zheng is the corresponding author.

References

1. Chen, W., Su, Y., Yan, X., Wang, W.: KGPT: knowledge-grounded pre-training for data-to-text generation. In: EMNLP (2020)
2. Chen, Y., Li, H., Hua, Y., Qi, G.: Formal query building with query structure prediction for complex question answering over knowledge base. In: IJCAI (2021)
3. Chen, Z.Y., Chang, C.H., Chen, Y.P., Nayak, J., Ku, L.W.: UHop: an unrestricted-hop relation extraction framework for knowledge-based question answering, pp. 345–356, June 2019
4. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: BERT: pre-training of deep bidirectional transformers for language understanding. In: NAACL, pp. 4171–4186 (2019)
5. Diefenbach, D., Migliatti, P.H., Qawasmeh, O., Lully, V., Singh, K., Maret, P.: Qanswer: A question answering prototype bridging the gap between a considerable part of the lod cloud and end-users. In: WWW, pp. 3507–3510 (2019)
6. Ding, J., Hu, W., Xu, Q., Qu, Y.: Leveraging frequent query substructures to generate formal queries for complex question answering. In: EMNLP-IJCNLP, pp. 2614–2622, November 2019
7. Dubey, M., Banerjee, D., Abdelkawi, A., Lehmann, J.: Lc-quad 2.0: a large dataset for complex question answering over wikidata and dbpedia. In: ISWC, pp. 69–78 (2019)
8. Gao, T., Yao, X., Chen, D.: SimCSE: simple contrastive learning of sentence embeddings. In: EMNLP, pp. 6894–6910 (2021)
9. Gong, H., Shen, Y., Yu, D., Chen, J., Yu, D.: Recurrent chunking mechanisms for long-text machine reading comprehension. In: ACL, pp. 6751–6761 (2020)
10. He, G., Lan, Y., Jiang, J., Zhao, W.X., Wen, J.R.: Improving multi-hop knowledge base question answering by learning intermediate supervision signals. In: WSDM, pp. 553–561 (2021)
11. Jain, S.: Question answering over knowledge base using factual memory networks. In: Proceedings of the NAACL Student Research Workshop, pp. 109–115 (2016)

12. Kapanipathi, P., Abdelaziz, I., Ravishankar, S., Roukos, S., Yu, M.: Leveraging abstract meaning representation for knowledge base question answering. In: Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021 (2021)
13. Ke, P., et al.: JointGT: graph-text joint representation learning for text generation from knowledge graphs. In: Findings of ACL-IJCNLP, pp. 2526–2538 (2021)
14. Kukich, K.: Design of a knowledge-based report generator. In: ACL, pp. 145–150. Association for Computational Linguistics, USA (1983)
15. Lan, Y., He, G., Jiang, J., Jiang, J., Zhao, W.X., Wen, J.R.: A survey on complex knowledge base question answering: Methods, challenges and solutions. In: IJCAI (2021)
16. Lewis, M., et al.: BART: denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In: ACL, pp. 7871–7880, July 2020
17. Liang, S., Stockinger, K., de Farias, T.M., Anisimova, M., Gil, M.: Querying knowledge graphs in natural language. J. Big Data **8**(1), 1–23 (2021). https://doi.org/10.1186/s40537-020-00383-w
18. Liu, K., Wang, F., Ding, Z., Liang, S., Yu, Z., Zhou, Y.: A review of knowledge graph application scenarios in cyber security. CoRR abs/2204.04769 (2022)
19. Pellissier Tanon, T., de Assunção, M.D., Caron, E., Suchanek, F.M.: Demoing platypus - a multilingual question answering platform for Wikidata. In: The Semantic Web: ESWC 2018 Satellite Events, pp. 111–116 (2018)
20. Pramanik, S., Alabi, J., Roy, R.S., Weikum, G.: UNIQORN: unified question answering over RDF knowledge graphs and natural language text. CoRR (2021)
21. Qiu, Y., Wang, Y., Jin, X., Zhang, K.: Stepwise reasoning for multi-relation question answering over knowledge graph with weak supervision. In: WSDM, pp. 474–482 (2020)
22. Rossiello, G., et al.: Generative relation linking for question answering over knowledge bases. In: ISWC, pp. 321–337 (2021)
23. Sakor, A., Singh, K., Patel, A., Vidal, M.E.: Falcon 2.0: an entity and relation linking tool over Wikidata. In: CIKM, pp. 3141–3148 (2020)
24. Saxena, A., Tripathi, A., Talukdar, P.: Improving multi-hop question answering over knowledge graphs using knowledge base embeddings. In: ACL, pp. 4498–4507 (2020)
25. Scarselli, F., Gori, M., Tsoi, A.C., Hagenbuchner, M., Monfardini, G.: The graph neural network model. IEEE Trans. Neural Networks **20**(1), 61–80 (2009)
26. Segal, E., Efrat, A., Shoham, M., Globerson, A., Berant, J.: A simple and effective model for answering multi-span questions. In: EMNLP, pp. 3074–3080 (2020)
27. Sun, H., Bedrax-Weiss, T., Cohen, W.: PullNet: open domain question answering with iterative retrieval on knowledge bases and text. In: EMNLP-IJCNLP, pp. 2380–2390 (2019)
28. Sun, H., Dhingra, B., Zaheer, M., Mazaitis, K., Salakhutdinov, R., Cohen, W.: Open domain question answering using early fusion of knowledge bases and text. In: EMNLP, pp. 4231–4242 (2018)
29. Vaswani, A., et al.: Attention is all you need. In: NIPS, pp. 6000–6010 (2017)
30. Wang, X., Liu, K., Wang, D., Wu, L., Fu, Y., Xie, X.: Multi-level recommendation reasoning over knowledge graphs with reinforcement learning. In: WWW, pp. 2098–2108 (2022)
31. Xiao, G., Corman, J.: Ontology-mediated SPARQL query answering over knowledge graphs. Big Data Res. **23**, 100177 (2021)

32. Yih, W.t., Chang, M.W., He, X., Gao, J.: Semantic parsing via staged query graph generation: Question answering with knowledge base. In: ACL, pp. 1321–1331 (2015)
33. Zhang, L., Wu, T., Chen, X., Lu, B., Na, C., Qi, G.: Auto insurance knowledge graph construction and its application to fraud detection. In: IJCKG, pp. 64–70 (2021)
34. Zheng, W., Yu, J.X., Zou, L., Cheng, H.: Question answering over knowledge graphs: question understanding via template decomposition. Proc. VLDB Endow. **11**(11), 1373–1386 (2018)