Backward Semantic for improving Multi-hop Question Answering on Knowledge Graph

Anonymous ACL submission

Abstract

Multi-hop question answering over knowledge graph utilizes the knowledge graph (KG) structure to infer answers. However, KG often lacks edges in the reasoning path from the question entity to the answer entity. Recent research focused on various KG embedding methods to obtain the semantics of the reasoning path (called forward semantics) to repair missing edges. However, the forward semantics method could drift as the path get longer. This paper proposes a bidirectional semantics embedding and matching method (BSEM) to alleviate the forward semantics drift problem. BSEM first leverages a backward semantics method to deduce the semantics of the opposite direction of the reasoning path. Then, BSEM constructs a two-stage learning method to merge the bidirectional (forward or backward) semantics and find the correct answer. In the two-stage learning method, joint learning is created to learn the bidirectional semantics of the reasoning path simultaneously; contrast learning is also used to improve the ability of the backward semantics to identify the correct answers that are not found by the forward semantics. Experiments on the two benchmarks, MetaQA and WebQSP, show that BSEM surpasses the five baseline methods, PullNet, EmQL, LEGO, EmbedKGQA, and KGT5. Especially for the incomplete KG – WebQSP, compared with the other four methods except for EmQL, BSEM improves the accuracy by 13.1%, 12.0%, 5.2% and 10.0%, respectively.

1 Introduction

Multi-hop question answering over knowledge graph (MHKGQA) (Zhang et al., 2018; Lan and Jiang, 2020; He et al., 2021) refers to discovering a path from the question entity to the answer entity in a knowledge graph (KG). The right path depends on that the KG has all edges related to the question. However, KG often is sparse and incomplete. Furthermore, they often lack some of those right edges so that the right path does not exist. Therefore, MHKGQA needs to have the ability to find edges with similar semantics to compensate for these missing edges, which helps obtain a reasoning path to infer the correct answer.

There are two clues in the MHKGQA research to overcome the KG incompleteness challenge. One is the supplementary text evidence method (Sun et al., 2018, 2019; Han et al., 2020), which extracts evidence from the supplementary text corpus and aggregates them and KG to predict the correct answer. Unfortunately, the method requires the corpus to be large enough to cover all semantics of missing edges in KG. It is difficult, even impossible, to acquire such a comprehensive corpus in most situations. The other clue is the KG embedding and matching method (Sun et al., 2020; Ren et al., 2021; Saxena et al., 2020; Niu et al., 2021). The method utilizes various KG embedding technologies (Bordes et al., 2013; Trouillon et al., 2016; Yang et al., 2015) to transform KG edges into an embedding space to learn their semantics. It then combines the semantics of edges in the path from the question entity to the answer entity (called **forward semantics**). Furthermore, it infers the correct answer by matching the forward semantics and question entity with the answer entity. For example, EmbedKGQA(Saxena et al., 2020) uses the ComplEx embedding method(Trouillon et al., 2016) to learn the representation (embeddings) of edges in the given KG. EmbedKGQA then learns the question embedding by a feed-forward neural network armed with RoBERTa (Liu et al., 2019). Finally, it combines these embeddings to match all candidate answer entities and selects the answer entity with the highest matching score. However, the method only focuses on the forward semantics, which could drift (Yadav et al., 2020) as the length of a reasoning path increases. Moreover, the forward semantics drift would degrade the performance of the KG embedding and matching method.

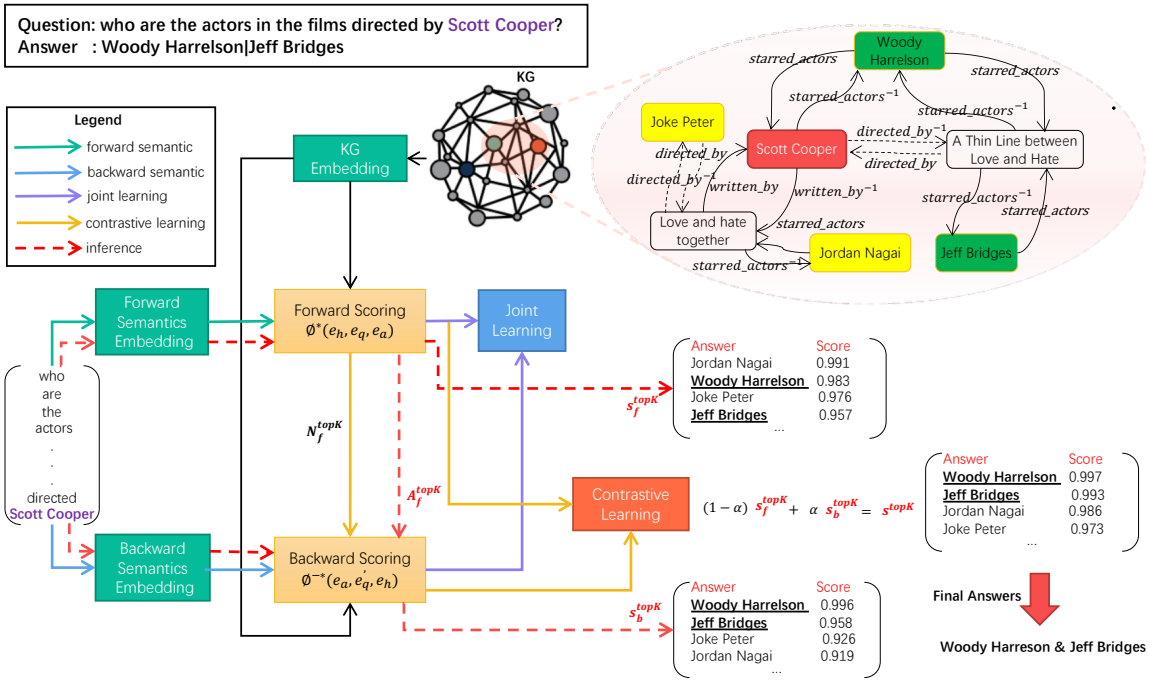

Figure 1: Structure of our model **BSEM**. The model consists of the following modules: 1) the KG embedding module, 2) the bidirectional semantics embedding module, and 3) the two-stage learning module. The KG embedding module uses the Knowledge Graph Embedding method to learn embeddings of all entities and relations in the input KG. The bidirectional semantics embedding module generates reasoning semantics for the given question. The two-stage learning, consisting of joint learning and contrast learning, merges the forward and backward semantics to infer the correct answers.

This paper proposes a new bidirectional semantics embedding and matching method (**BSEM**) to alleviate the forward semantics drift, as shown in Figure 1. BSEM first leverages a backward semantics embedding module to capture the semantics of edges in the reasoning path from the answer entity to the question entity (called **backward semantics**). The backward semantics provides the representation that restores the question entity from the given answer entity. The representation can distinguish failure answers obtained by the forward semantics, especially when these failure answers are close to the correct answer in the KG embedding space. For example, as shown in the KG in Figure 1, "Joke Peter", "Jordan Nagai", "Woody Harrelson", and "Jeff Bridges" are semantically closed because they have similar semantics of neighbour nodes and edges. Owing to semantic drift, the forward semantics embedding module could find the failure answer – "Jordan Nagai". While the back semantics embedding module can identify that the answer is unsuccessful because the restored question obtained by the module is far from the right question entity. Therefore, the back semantics embedding module can repair potential failures caused by the forward semantics drift.

BSEM then constructs a two-stage learning method that consists of joint learning and contrast learning to merge the forward and backward semantics and infer the correct answer. Joint learning is designed to learn the bidirectional (forward and backward) semantics simultaneously. Furthermore, it creates a novel evaluation strategy to improve the performance of the forward semantics by using the backward semantics. After joint learning, contrast learning is imported to enhance the ability of the backward semantics to identify the unsuccessful answer. Finally, BSEM combines the outputs of the forward and backward semantics to predict the correct answer.

Compared with previous MHKGQAs with semantic embeddings, such as EmbedKGQA (Saxena et al., 2020) and KGT5 (Saxena et al., 2022), BSEM has a significant advantage in reasoning paths. The reason is as follows: 1) the backward embedding has the ability to restore the question from the given answer, like the solution verification in mathematics that ensures the solution satisfies any given condition. Moreover, the backward embedding can verify whether an answer is right or

wrong. 2) With the verification, the backward semantics module can alleviate the forward semantics drift by the above joint learning. 3) According to the same argument, the forward semantics module can also enhance the verification ability of the backward semantics via contrast learning.

The main contributions in the paper are summarized as follows.

- We propose a backward semantics embedding method to verify whether an answer is right or wrong. To learn the backward semantics, we create a novel backward score function based on the forward score function of the embedding model (Trouillon et al., 2016), as defined in Equation 7. Since the novel score function only uses the partial forward score function, it avoids the disadvantage of no backward semantics dataset. This approach complements and enhances existing forward semantic-only methods.

- We propose a bidirectional semantics embedding and matching method (**BSEM**) to alleviate the forward semantics drift problem. Armed with the two-stage learning consisting of joint learning and contrast learning, BSEM can improve the performance of the forward semantics and enhance the ability of the backward semantics to distinguish the incorrect answer.

- Experiments using two benchmarks, MetaQA and WebQSP, illustrate that BSEM surpasses the five baseline methods of MHKGQA, PullNet (Sun et al., 2019), EmQL (Sun et al., 2020), LEGO (Ren et al., 2021), EmbedKGQA (Saxena et al., 2020) and KGT5 (Saxena et al., 2022). Especially on the incomplete KG benchmark – WebQeustionSP, compared with the other four methods except for EmQL, BSEM improves the accuracy by 13.1%, 12.0%, 5.2% and 10.0%, respectively.

The rest of this paper is organized as follows. Section 2 introduces related works on Multi-hop KGQA. Then, Section 3 provides details for the proposed BSEM method. Moreover, Section 4 shows the experimental results and analysis. Finally, Section 5 concludes the paper.

2 Related Work

Multi-hop KGQA (Zhang et al., 2018; Lan and Jiang, 2020; He et al., 2021) requires reasoning over multiple edges in a KG to predict the correct answer. The method may fail in an incomplete KG because the KG lacks some edges between the question and answer. Some prior works (Sun et al., 2018, 2019; Han et al., 2020) use external relevant text corpora to augment the KG and reason over the combinations of the text corpora and KG. GRAFT-Net (Sun et al., 2018) and PullNet (Sun et al., 2019) both supplement the incomplete KG with information extracted from text corpora and reason over a question-specific sub-graph containing the KG and text corpus. The method proposed by (Han et al., 2020) utilizes the semantics of the text corpus to enrich the representation of entities and complete the edges through the structural information in the text. However, the text corpora are difficult to satisfy these methods.

Recently, other works have attempted to overcome the KG incompleteness challenge using KG embeddings methods(Sun et al., 2020; Ren et al., 2021; Saxena et al., 2020; **?**). EmQL(Sun et al., 2020) uses neural retrieval over embedded triples in a KG to implement some logical operations and applies a randomized data structure called a count-min sketch to propagate scores of logically-entailed entities. LEGO(Ren et al., 2021) executes a query synthesis module and a KG embedding module iteratively. At each step, the query synthesis module uses the question embedding to infer a reasoning action and then the KG embedding module matches the action in the embedding space. EmbedKGQA(Saxena et al., 2020) uses the ComplEx embedding method(Trouillon et al., 2016) to learn the representation (embeddings) of edges and entities in the given KG. EmbedKGQA then calculates the score of each candidate answer entity by using the question embedding to approximate the semantics of the reasoning path. PKEEQA(Niu et al., 2021) utilizes a customizable path representation mechanism to evaluate the ambipolar correlation between a path embedding and a question embedding. KGT5(Saxena et al., 2022) regards KG link prediction and KGQA as sequence-to-sequence tasks; thus KGT5 trains an encoder-decoder transformer model (Raffel et al., 2020) in the link prediction task. These above methods only focus on the forward semantic from the question entity to the answer entity. Different from these methods, this paper imports the backward semantics to compensate for the forward semantics to find the correct answer.

3 Bidirectional Semantics Embedding and Matching Method

In this section, we first formalize the problem that needs to be addressed by our proposed bidirectional semantics embedding and matching method (BSEM). Then we describe the implementation details of the method. The method consists of three modules: the KG embedding module, the question embedding module and the semantics reasoning module. As shown in Figure 1, the KG embedding module utilizes the Knowledge Graph Embedding method – ComplEx (Trouillon et al., 2016) to learn embeddings of all entities and edges in a given KG. The question embedding module generates the forward and backward semantics for the given question. The semantics reasoning module then imports joint learning and contrast learning to combine the forward and backward semantics and infers the correct answers. Finally, we introduce how to train and predict using BSEM.

3.1 Problem Definition

Knowledge Graph($\mathcal{G}$) consists of a large number of triples $\langle h, r, t\rangle$s, where h and t are entities and r is an edge. Let $\mathcal{E}$ be the entity set and $\mathcal{R}$ be the edge set. Then, $\mathcal{G} = \mathcal{E} \times \mathcal{R} \times \mathcal{E}$. Given a question q and a $\mathcal{G}$, the goal of Multi-hop KGQA (MHKGQA) is to find the correct answer a from the question entity h via several hops in $\mathcal{G}$ (called reasoning path p), such as $p = \langle h, r_1, t_1, r_2, ..., t_{n-1}, r_n, a\rangle$, defined by the following equation.

$$\underset{a,p\in\mathcal{G}}{\operatorname{argmax}}\mathcal{F}(q, p, a) \tag{1}$$

, where $\mathcal{F}$ is a match function, i.e., $\mathcal{F} : q \times p \times a \rightarrow R$.

For MHKGQA using KG embedding (EMHKGQA), since each embedded entity contains the semantics of its neighbour edges and entities in $\mathcal{G}$, it contains the semantics of its reasoning path. So, EMHKGQA learns the reasoning path semantics p from a given question q to an answer and matches p with each embedded entity to find the correct answer a. Previous EMHKGQA studies only focus on extracting the forward reasoning semantic paths from the question to the answer $\phi^*(h, r^*, a)$ (called the forward semantics), and this could lead to semantic drift as the length of the path between the question and answer becomes longer. To alleviate the semantic drift problem, we leverage the backward reasoning path semantics $\phi^{-*}(a, r^{-*}, h)$ from the answer to the question (called the backward semantics). Then we define BSEM by the following equation.

$$\underset{a,p\in\mathcal{G}}{\operatorname{argmax}}\mathcal{F}(q, p, a) \approx \underset{e_a\in\mathcal{G}}{\operatorname{argmax}}\mathcal{F}(e_q, \langle\phi^*, \phi^{-*}\rangle, e_a) \tag{2}$$

, where q_e is the question embedding, and e_a is an entity embedding in $\mathcal{G}$.

3.2 KG Embedding Module

To obtain each entity embedding e_a in $\mathcal{G}$, we apply the ComplEx embedding method(ComplEx)(Trouillon et al., 2016) to embed entities and relations into a complex space. Given $h, t \in \mathcal{E}$ and $r \in \mathcal{R}$ in the KG, ComplEx applies a triple scoring function $\phi : \mathcal{E} \times \mathcal{R} \times \mathcal{E} \rightarrow R$ to learn the representations as $e_h, e_r, e_t \in C^d$. The scoring function is as follows:

$$\begin{aligned}\phi(h, r, t) &= Re\left(\langle e_h, e_r, \bar{e}_t\rangle\right) \\ &= Re\left(\sum_{k=1}^{d} e_h^{(k)} e_r^{(k)} \bar{e}_t^{(k)}\right)\end{aligned} \tag{3}$$

, where $Re(\cdot)$ refers to the real part of a complex number. $e_h^{(k)}$, $e_r^{(k)}$, and $\bar{e}_t^{(k)}$ are the k-th dimension values in e_h, e_r, and $\bar{e}_t$, respectively. For each correct triple $\langle h, r, t\rangle \in \mathcal{G}$, the model assigns the score $\phi(h, r, t) > 0$. While, for each incorrect triple $\langle h^{'}, r^{'}, t^{'}\rangle \notin \mathcal{G}$, the model assigns the score $\phi(h^{'}, r^{'}, t^{'}) < 0$.

3.3 Question Embedding Module

To acquire the question embedding e_q, we construct a feed-forward neural network (NN) that embeds the question q to the same dimension vector $e_q \in C^d$. The NN consists of a Bi-LSTM (or pre-trained RoBERTa (Liu et al., 2019)), and two fully connected layers with ReLU activation and dropout. The Bi-LSTM embeds the question q into a 512-dimensional vector for the large question dataset (MetaQA in Section 4.1), while the pre-trained RoBERTa transforms the question q into a 768-dimensional vector to capture rich semantics for the small question dataset (WebQSP in Section 4.1). The final two fully connected layers project the 512-dimensional (or 768-dimensional) vector into a complex space C^d.

As the question semantics are different in the forward reasoning path and the backward reasoning path, we extend e_q to the forward question

embeddings $e_{q'}$ and backward embeddings $e_{q^{-\prime}}$, i.e., $e_q = \langle e_{q'}, e_{q^{-\prime}} \rangle$. We use two NNs described above to obtain $e_{q'}$ and $e_{q^{-\prime}}$ respectively, as shown in Figure 1.

3.4 Semantic Reasoning Module

The module utilizes joint learning to simultaneously capture the forward and backward semantics, ϕ^* and ϕ^{-*}. Joint learning can improve the forward semantics ϕ^*; in addition, the module uses contrast learning to enhance the backward semantics ϕ^{-*} to identify the unsuccessful answer.

3.4.1 Forward Semantics

The forward semantics ϕ^* is defined as a score function, i.e., $\phi^*(e_h, e_{r^*}, e_a) > 0$ if e_a is the correct answer and e_{r^*} is the right reasoning path; otherwise, $\phi^*(e_h, e_{r^*}, e_a) < 0$. In this paper, we assume that the question q contains the reasoning path semantics e_{r^*}. So, we replace e_{r^*} with the forward question embeddings $e_{q'}$, i.e., $\phi^*(e_h, e_{r^*}, e_a) = \phi^*(e_h, e_{q'}, e_a)$.

To obtain $\phi^*(e_h, e_{q'}, e_a)$, we apply the same method as EmbedKGQA (Saxena et al., 2020) that utilizes Equation 3 to generate the score function. In detail, the method is defined by the following two equations 4 and 5.

$$\begin{aligned} &\phi^*(e_h, e_{q'}, e_a) > 0, a \in A \\ &\phi^*(e_h, e_{q'}, e_{a'}) < 0, a^{'} \notin A \end{aligned} \tag{4}$$

$$\begin{aligned} \phi^*(e_h, e_{q'}, e_a) &= Re\left(\langle e_h, e_{q'}, \bar{e}_a \rangle\right) \\ &= Re\left(\sum_{k=1}^{d} e_h^{(k)} e_{q'}^{(k)} \bar{e}_a^{(k)}\right) \end{aligned} \tag{5}$$

3.4.2 Backward Semantics

Similar to the forward semantics, the backward semantics ϕ^{-*} is also a score function defined by the following equation.

$$\begin{aligned} &\phi^{-*}(e_a, e_{q^{-\prime}}, e_h) > 0, a \in A \\ &\phi^{-*^{'}}(e_{a'}, e_{q^{-\prime}}, e_h) < 0, a^{'} \notin A \end{aligned} \tag{6}$$

However, there is not any question dataset that represents the backward reasoning path from the question to the answer. So, it is difficult to obtain the backward semantics $\phi^{-*}(e_a, e_{q^{-\prime}}, e_h)$ directly from the question dataset. To overcome this difficulty, we utilize the ComplEx symmetry to compute $\phi^{-*}(e_a, e_{q^{-\prime}}, e_h)$ by $\phi^*(\bar{e}_h, e_{q^{-\prime}}, \bar{e}_a)$ according to the following equation.

$$\begin{aligned} &\phi^{-*}(e_a, e_{q^{-\prime}}, e_h) = Re(\langle e_a, e_{q^{-\prime}}, \bar{e}_h \rangle) = \\ &Re\left(\sum_{k=1}^{d} e_a^{(k)} e_{q^{-\prime}}^{(k)} \bar{e}_h^{(k)}\right) = Re\left(\sum_{k=1}^{d} \bar{e}_h^{(k)} e_{q^{-\prime}}^{(k)} e_a^{(k)}\right) \\ &= Re(\langle \bar{e}_h, e_{q^{-\prime}}, e_a \rangle) = \phi^*(\bar{e}_h, e_{q^{-\prime}}, \bar{e}_a) \end{aligned} \tag{7}$$

3.4.3 Joint Learning

Joint learning is designed to simultaneously learn the bidirectional (forward and backward) semantics. In joint learning, we leverage the worse evaluation strategy to improve the bidirectional semantics according to the following equation.

$$s = \begin{cases} \min\{\phi^*, \phi^{-*}\}, a \in A \\ \max\{\phi^*, \phi^{-*}\}, a' \notin A \end{cases} \tag{8}$$

The worse evaluation strategy takes the smaller value between ϕ^* and ϕ^{-*} when a is the correct answer. Conversely, the worse evaluation strategy adopts the bigger value between ϕ^* and ϕ^{-*} when a is not the correct answer. The smaller and bigger values both benefit the above two NNs of question embedding to improve their semantics accuracy. The reason is that the smaller and bigger values allow the loss function to obtain a bigger loss value. The bigger loss value can improve the NN performance.

We use the Kullback-Leibler divergence(KLD)(Kullback and Leibler, 1951) as the loss function to train the two NNs of question embeddings.

$$\mathcal{L} = \mathcal{KLD}(s, s^*) \tag{9}$$

, where s^* is the target labels. The target label is 1 if a is the correct answer. Otherwise, it is 0. Considering the total number of entities in KGs is large, we use label smoothing to improve the generalization capability.

3.4.4 Contrast Learning

Contrast learning aims to enhance the ability of the backward semantics ϕ^{-*} to identify unsuccessful answers found by the forward semantics ϕ^*. We apply the following method to provide positive and negative samples for contrast learning. For each question q, the forward semantics ϕ^* computes each entity in the KG $\mathcal{G}$, generating the candidate answer set sorted by ϕ^*. In the candidate answer set, if an entity is a correct answer, the entity is the

positive sample; The top-K entities that are not the correct answers are the negative samples.

Different from the traditional contrast learning (Khosla et al., 2020), the contrast learning method leverages a new contrast loss function. The contrast loss function gives each positive sample a weighted score to show its importance for the backward semantics ϕ^{-*}. The reason is that different positive samples have different contributions to ϕ^{-*}. Considering all questions in the training dataset, the contrast loss function is defined by the following equation.

$$\mathcal{L}_{cl} = -\sum_{j=1}^{n}\sum_{i\in P_j} \gamma_i * \log \frac{exp\left(\phi_i^{-*}/\tau\right)}{exp\left(\phi_i^{-*}/\tau\right) + \sum_{k\in N_j} exp\left(\phi_k^{-*}/\tau\right)} \tag{10}$$

, where n is the number of questions in the training dataset, and P_j (N_j) is the positive (negative) sample set of the jth question. γ_i is the weight score of the ith entity in P_j. The weight scores γ of P_j are attained by passing the scores obtained from the forward reasoning path semantics through the softmax function. i.e., $\gamma = softmax(\phi^*(e_h, e_{q_j'}, e_{P_j}))$.

When training the backward semantics ϕ^{-*} with the above contrast loss function (Equation 10), the forward reasoning path semantics ϕ^* remain unchanged.

3.5 Prediction

In the prediction stage, the forward reasoning path semantics ϕ^* is used to select top-K entities in $\mathcal{G}$ as the candidate answer set S. Then, for each entity e in S, e is graded by the following equation.

$$s = (1-\alpha)\phi^* + \alpha\phi^{-*} \tag{11}$$

, where $\alpha \in (0, 1)$ is a hyper-parameter. After sorting the candidate answer set, the prediction chooses the first (top-1) entity as the correct answer.

4 Experiments

We evaluate our method (BSEM) by comparing it with five baseline MHKGQA methods, PullNet(Sun et al., 2019), EmQL(Sun et al., 2020), EmbedKGQA(Saxena et al., 2020), LEGO(Ren et al., 2021), and KGT5(Saxena et al., 2022), on the two benchmark datasets, MetaQA(Zhang et al., 2018) and WebQSP(Yih et al., 2016). We first introduce the two benchmark datasets and BSEM parameters. Then, we give the experimental results and analysis.

4.1 Datasets and Parameter Setting

MetaQA(Zhang et al., 2018) and WebQSP(Yih et al., 2016) are both multi-hop KGQA datasets. MetaQA focuses on the movie domain, with more than 400k questions consisting of 1-hop, 2-hop and 3-hop questions. Moreover, the answers to these questions can be found in the KG WikiMovies(Miller et al., 2016), which has 135k triples, 43k entities and nine relations. WebQSP contains 4737 natural language questions consisting of 1-hop and 2-hop questions that need to be answered by reasoning on the KG Freebase(Bollacker et al., 2008). However, our experiments do not use Freebase but its subset from (Saxena et al., 2022). Our QA datasets are the same as (Saxena et al., 2020, 2022), listed in Table 1.

Dataset	Train	Valid	Test
MetaQA 1-hop	96,106	9,992	9,947
MetaQA 2-hop	118,980	14,872	14,872
MetaQA 3-hop	114,196	14,274	14,274
WebQSP	2,998	100	1,639

Table 1: Statistics for MetaQA and WebQSP datasets used in our experiments.

To simulate KG incompleteness to evaluate our method, we directly use the incomplete KGs in (Saxena et al., 2022). These incomplete KGs are generated by randomly cutting down 50% of edges from the two KGs: WikiMovies and the subset of Freebase. Armed with different KGs, the two above datasets are called different names. For example, MetaQA with full KG (WikeMovies) is called "MetaQA full-KG", while it with half KG (random cut 50% edges) is called "MetaQA half-KG". As some state-of-the-art methods apply these incomplete KGs and the above two benchmark datasets (MetaQA and WebQSP), it facilitates comparisons between our method BSEM and the above five baseline methods. However, these incomplete KGs are not the same owing to the above generation of random cuts. So, the ground truth SPARQL query (GT query) method for test questions is imported as the estimated standard. Comparing the GT query with all other methods in the experiment facilitates distinguishing their performance.

We apply hits@1 as our evaluation metric. For the parameters of our method BSEM, we set top-100 in the contrast learning and $\tau = 1.0$. Through

the grid search in hyperparameter tuning, we set α 0.8, 0.3 and 0.8 for the datasets MetaQA 1-hop-half, 2-hop-half and 3-hop-half, respectively. For the datasets WebQSP with half KG and WebQSP with full KG, we set α 0.5 and 0.3 respectively. All methods run on the machine with Nvidia 3090Ti GPU.

4.2 Experimental Results and Analysis

Table 2 shows experimental results on the MetaQA dataset with half-KG. BSEM surpasses the five baseline methods, PullNet, EmQL, LEGO, EmbedKGQA and KGT5, on the "1-hop" and "2-hop" MetaQA datasets. The reason is that the backward semantics can alleviate the semantic drift of the multi-hop reasoning path. For example, as shown in Figure 1, "Jordan Nagai" has a higher score than "Woody Harrelson" and "Jeff Bridges" in the result attained by the forward semantics, while "Jordan Nagai" has a weaker score by performing the backward semantics. After embedding the bidirectional semantics, BSEM has the ability to find answers more accurately. Especially compared with EmbedKGQA that only has the forward semantics, BSEM improves accuracy by 0.4%, 1.5%, and 2.7% in the "1-hop", "2-hop", and "3-hop". The more hops, the more accuracy improves.

Model	1-hop	2-hop	3-hop
GT query	63.3	45.8	45.3
PullNet	65.1(+1.8)	52.1(+6.3)	59.7(+14.4)
EmbedKGQA	70.6(+7.3)	54.3(+8.5)	53.5(+8.2)
EmQL	63.8(+0.5)	47.6(+1.8)	48.1(+2.8)
LEGO	69.3(+6.0)	57.8(+12.0)	63.8(+18.5)
GT query	67.7	48.7	44.4
KGT5	75.0(+7.3)	36.2(-8.2)	**64.4(+20.0)**
EmbedKGQA	75.2(+7.5)	63.7(+15.0)	43.5(-0.9)
Our BSEM	**75.6(+7.9)**	**65.2(+16.5)**	46.2(+1.8)

Table 2: Results on the dataset MetaQA half-KG with hits@1. Values in parentheses indicate increased accuracy compared with "GT query". A bold value is the best result.

However, on the "3-hop" tasks, BSEM cannot outperform three baseline methods, PullNet, LEGO, and KGT5. Compared with KGT5, which has the transformer-based encoder-decoder model, BSEM only uses bidirectional semantics to find the correct answer. The bidirectional semantics can alleviate the multi-hop semantic drift but cannot prevent it. While KGT5 has better accuracy than BSEM on the "3-hop", KGT5 has very low accuracy (36.2%) on the "2-hop". KGT5 is against the commonsense, i.e., the more hops in reasoning, the more uncertain in answer. The reason is that the transformer-based encoder-decoder model in KGT5 could concentrate on hops in the reasoning path but overlook associations between these hops. The other two baseline methods, PollNet and LEGO, have similar accuracies and reasons to KGT5 on "2-hop" and "3-hop". Due to the limitation on the number of hops in the two datasets, we cannot predict these anti-commonsense methods, KGT5, PullNet, and LEGO, will definitely be better than our BSEM in more hops (4-hop, 5-hop,...). In short, BSEM has the best overall accuracy on the dataset MetaQA half-KG.

In addition, BSEM also can obtain the most accurate answers in the dataset WebQSP half-KG, as shown in Table 3. Especially, comparing the improvements of all baseline methods, PullNet, LEGO, EmbedKGQA, and KGT5, on the GT query, BSEM improves the accuracy by 13.1%, 12.0%, 5.2% and 10.0%, respectively. Results in Tables 2 and 3 illustrate that, compared with the five baseline methods, BSEM can find a more accurate answer to the multi-hop reasoning in a sparse KG.

Model	WebQSP half-KG
GT query	56.9
PullNet	47.4 (-9.5)
EmbedKGQA	42.5 (-14.4)
LEGO	48.5 (-8.4)
GT query	56.9
KGT5	50.5 (-6.4)
EmbedKGQA	55.3 (-1.6)
Our BSEM	**60.5 (+3.6)**

Table 3: Results on the dataset WebQSP half-KG with hits@1. As the dataset WebQSP does not provide the obvious information on which question is "one hop" or "two hop", the values are the accuracy of all methods on the overall dataset WebQSP half-KG.

Table 4 show experimental results on the dataset WebQSP with full-KG. These results show that BESM is better than KGT5 and EmbedKGQA. Moreover, BSEM can still find the more accurate answer on the full KG; however, BSEM is weaker than CBR-KGQA (Das et al., 2021). The reason is that CBR-KGQA imports additional information, such as SPARQL queries, to help it obtain the right logical query form. Furthermore, CBR-KGQA depends on the expensive annotations of this additional information at scale. Once the KG changes, CBR-KGQA needs to relabel these annotations. So, the BSEM result is acceptable without

the above additional information.

Model	WebQSP full-KG
KGT5	56.1
EmbedKGQA*	68.3
Our BSEM	70.8
CBR-KGQA	**73.1**

Table 4: Results on the dataset WebQSP full-KG with hits@1.

4.3 Ablation Studies

Our method BSEM mainly contains the two sub-models, the bidirectional reason path semantics and the two-stage learning consisting of joint learning (JL) and contrast learning (CL). To prove that these sub-models can improve the performance of BSEM running on the KG, we ablate the above sub-models step by step.

Table 5 gives the ablation results on the dataset WebQSP half-KG. "{-JL}" means that BSEM cuts down joint learning; "{-CL}" cuts down contrast learning. "{-JL and -CL}" deletes joint learning, contrast learning and backward semantics. So, after "{-JL and -CL}", BSEM degrades into EmbedKGQA that only contains the forward semantics.

Model	WebQSP half-KG
Our BSEM	**60.5**
{-JL}	58.4
{-CL}	57.8 (56.7)
{-JL and -CL}	55.3

Table 5: Ablation experiments. The value in the parentheses is the result outputted by the forward reasoning path semantics.

Compared with "{-JL and -CL}" (EmbedKGQA), BSEM with "{-CL}" (i.e., joint learning) improves the accuracy from 55.3% to 56.7%. After injecting the backward semantics, joint learning has 57.8% accuracy and is 2.5% more accurate than EmbedKGQA. Similarly, BSEM with "{-JL}" (contrast learning) has 58.4% accuracy and is more 3.1% accurate than EmbedKGQA. After merging joint learning and contrast learning, our BSEM obtains 60.5% accuracy. The above ablation experiments show that these proposed sub-models can improve the performance of Multi-hop KGQA.

4.4 The Weight Scores in Contrast Learning

Our method BSEM leverages a weighted score γ_i in contrast learning (Equation 10) to show the importance of the positive sample to the backward semantics ϕ^{-*}. Table 6 lists the results obtained by BSEM with/without γ_i in contrast learning. The weighted score γ_i can improve accuracy by 0.1%, 0.2%, 0.7% on the datasets MetaQA half-KG with "1-hop", "2-hop", and "3-hop". As the number of hops increases, the γ_i effect becomes strong. Table 6 shows that importing γ_i improves contrast learning accuracy.

DataSet	WS	NWS
MetaQA half-KG 1-hop	75.6	75.5
MetaQA half-KG 2-hop	65.2	65.0
MetaQA half-KG 3-hop	46.2	45.5
WebQSP half-KG	60.5	59.8
WebQSP full-KG	70.8	70.4

Table 6: **WS** represents our method results and **NWS** shows results obtained by our method without the weighted score γ_i in contrast learning.

5 Conclusion

This paper proposes BSEM, a novel bidirectional semantic embedding and matching method to implement the multi-hop KGQA. BSEM utilizes the backward semantics to repair the forward semantics drift. BSEM then applies joint learning and contrast learning to merge the bidirectional semantics to find the correct answer. Joint learning creates the worse evaluation strategy to improve the performance of training bidirectional semantics. While contrast learning imports the weighted score to enhance the ability of the backward semantics to identify the unsuccessful answers.

Compared with the state-of-the-art methods, PullNet, EMQL, LEGO, EmbedKGQA, and KGT5, BSEM has the best overall performance on the multi-hop KGQA. However, BSEM only can alleviate the semantic drift in the reasoning path but cannot prevent it. The reason is that the forward and backward semantics could still drift as the reasoning path gets longer. Inspired by the KGT5 model, in future, we plan to explore the encoder-decoder model based on bidirectional semantics to prevent semantic drift.

References

Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: a collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, pages 1247–1250.

Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. *Advances in neural information processing systems*, 26.

Rajarshi Das, Manzil Zaheer, Dung Thai, Ameya Godbole, Ethan Perez, Jay Yoon Lee, Lizhen Tan, Lazaros Polymenakos, and Andrew McCallum. 2021. Case-based reasoning for natural language queries over knowledge bases. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 9594–9611.

Jiale Han, Bo Cheng, and Xu Wang. 2020. Open domain question answering based on text enhanced knowledge graph with hyperedge infusion. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1475–1481.

Gaole He, Yunshi Lan, Jing Jiang, Wayne Xin Zhao, and Ji-Rong Wen. 2021. Improving multi-hop knowledge base question answering by learning intermediate supervision signals. In *Proceedings of the 14th ACM International Conference on Web Search and Data Mining*, pages 553–561.

Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. 2020. Supervised contrastive learning. *Advances in Neural Information Processing Systems*, 33:18661–18673.

Solomon Kullback and Richard A Leibler. 1951. On information and sufficiency. *The annals of mathematical statistics*, 22(1):79–86.

Yunshi Lan and Jing Jiang. 2020. Query graph generation for answering multi-hop complex questions from knowledge bases. Association for Computational Linguistics.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Alexander Miller, Adam Fisch, Jesse Dodge, Amir-Hossein Karimi, Antoine Bordes, and Jason Weston. 2016. Key-value memory networks for directly reading documents. *arXiv preprint arXiv:1606.03126*.

Guanglin Niu, Yang Li, Chengguang Tang, Zhongkai Hu, Shibin Yang, Peng Li, Chengyu Wang, Hao Wang, and Jian Sun. 2021. Path-enhanced multi-relational question answering with knowledge graph embeddings. *arXiv preprint arXiv:2110.15622*.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, Peter J Liu, et al. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.*, 21(140):1–67.

Hongyu Ren, Hanjun Dai, Bo Dai, Xinyun Chen, Michihiro Yasunaga, Haitian Sun, Dale Schuurmans, Jure Leskovec, and Denny Zhou. 2021. Lego: Latent execution-guided reasoning for multi-hop question answering on knowledge graphs. In *International Conference on Machine Learning*, pages 8959–8970. PMLR.

Apoorv Saxena, Adrian Kochsiek, and Rainer Gemulla. 2022. Sequence-to-sequence knowledge graph completion and question answering. *arXiv preprint arXiv:2203.10321*.

Apoorv Saxena, Aditay Tripathi, and Partha Talukdar. 2020. Improving multi-hop question answering over knowledge graphs using knowledge base embeddings. In *Proceedings of the 58th annual meeting of the association for computational linguistics*, pages 4498–4507.

Haitian Sun, Andrew Arnold, Tania Bedrax Weiss, Fernando Pereira, and William W Cohen. 2020. Faithful embeddings for knowledge base queries. *Advances in Neural Information Processing Systems*, 33:22505–22516.

Haitian Sun, Tania Bedrax-Weiss, and William W Cohen. 2019. Pullnet: Open domain question answering with iterative retrieval on knowledge bases and text. *arXiv preprint arXiv:1904.09537*.

Haitian Sun, Bhuwan Dhingra, Manzil Zaheer, Kathryn Mazaitis, Ruslan Salakhutdinov, and William W Cohen. 2018. Open domain question answering using early fusion of knowledge bases and text. *arXiv preprint arXiv:1809.00782*.

Théo Trouillon, Johannes Welbl, Sebastian Riedel, Éric Gaussier, and Guillaume Bouchard. 2016. Complex embeddings for simple link prediction. In *International conference on machine learning*, pages 2071–2080. PMLR.

Vikas Yadav, Steven Bethard, and Mihai Surdeanu. 2020. Unsupervised alignment-based iterative evidence retrieval for multi-hop question answering. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4514–4525.

Min-Chul Yang, Do-Gil Lee, So-Young Park, and Hae-Chang Rim. 2015. Knowledge-based question answering using the semantic embedding space. *Expert Systems with Applications*, 42(23):9086–9104.

Wen-tau Yih, Matthew Richardson, Christopher Meek, Ming-Wei Chang, and Jina Suh. 2016. The value of semantic parse labeling for knowledge base question answering. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 201–206.

Yuyu Zhang, Hanjun Dai, Zornitsa Kozareva, Alexander J Smola, and Le Song. 2018. Variational reasoning for question answering with knowledge graph. In *Thirty-second AAAI conference on artificial intelligence*.