Improving Multi-hop Question Answering over Knowledge Graphs using Knowledge Base Embeddings

Apoorv Saxena* **Aditay Tripathi*** **Partha Talukdar**
Indian Institute of Science, Bangalore
{apoorvsaxena,aditayt,ppt}@iisc.ac.in

Abstract

Knowledge Graphs (KG) are multi-relational graphs consisting of entities as nodes and relations among them as typed edges. Goal of the Question Answering over KG (KGQA) task is to answer natural language queries posed over the KG. Multi-hop KGQA requires reasoning over multiple edges of the KG to arrive at the right answer. KGs are often incomplete with many missing links, posing additional challenges for KGQA, especially for multi-hop KGQA. Recent research on multi-hop KGQA has attempted to handle KG sparsity using relevant external text, which isn't always readily available. In a separate line of research, KG embedding methods have been proposed to reduce KG sparsity by performing missing link prediction. Such KG embedding methods, even though highly relevant, have not been explored for multi-hop KGQA so far. We fill this gap in this paper and propose EmbedKGQA. EmbedKGQA is particularly effective in performing multi-hop KGQA over sparse KGs. EmbedKGQA also relaxes the requirement of answer selection from a prespecified neighborhood, a sub-optimal constraint enforced by previous multi-hop KGQA methods. Through extensive experiments on multiple benchmark datasets, we demonstrate EmbedKGQA's effectiveness over other state-of-the-art baselines.

1 Introduction

Knowledge Graphs (KG) are multi-relational graphs consisting of millions of entities (e.g., *San Jose*, *California*, etc.) and relationships among them (e.g., *San Jose-cityInState-California*). Examples of a few large KGs include Wikidata (Google, 2013), DBPedia (Lehmann et al., 2015), Yago (Suchanek et al., 2007), and NELL (Mitchell et al., 2018). Question Answering over Knowledge Graphs (KGQA) has emerged as an important research area over the last few years (Zhang et al., 2018; Sun et al., 2019a). In KGQA systems, given a natural language (NL) question and a KG, the right answer is derived based on analysis of the question in the context of the KG.

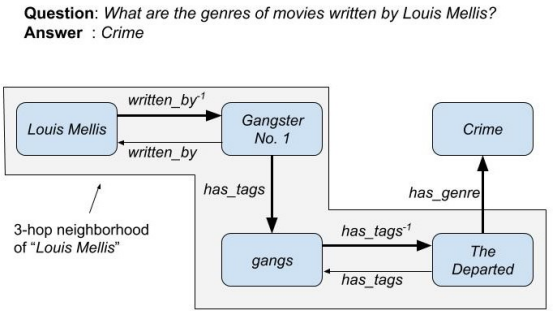

Figure 1: *Challenges with Multi-hop QA over Knowledge Graphs (KGQA) in sparse and incomplete KGs*: Absence of the edge *has_genre(Gangster No. 1, Crime)* in the incomplete KG makes it much harder to answer the input NL question, as the KGQA model potentially needs to reason over a longer path over the KG (marked by bold edges). Existing multi-hop KGQA methods also impose heuristic neighborhood limits (shaded region in the figure), which often makes the true answer (*Crime* in this example) out of reach. EmbedKGQA, our proposed method, overcomes these limitations by utilizing embeddings of the input KG during multi-hop KGQA. For more details, please refer Figure 2 and Section 4.

In *multi-hop KGQA*, the system needs to perform reasoning over multiple edges of the KG to infer the right answer. KGs are often incomplete, which creates additional challenges for KGQA systems, especially in case of multi-hop KGQA. Recent methods have used an external text corpus to handle KG sparsity (Sun et al., 2019a, 2018). For example, the method proposed in (Sun et al., 2019a) constructs a question-specific sub-graph from the KG, which is then augmented with supporting text documents.

*Equal contribution
EmbedKGQA's source code is available at https://github.com/malllabiisc/EmbedKGQA

Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pages 4498–4507
July 5 - 10, 2020.

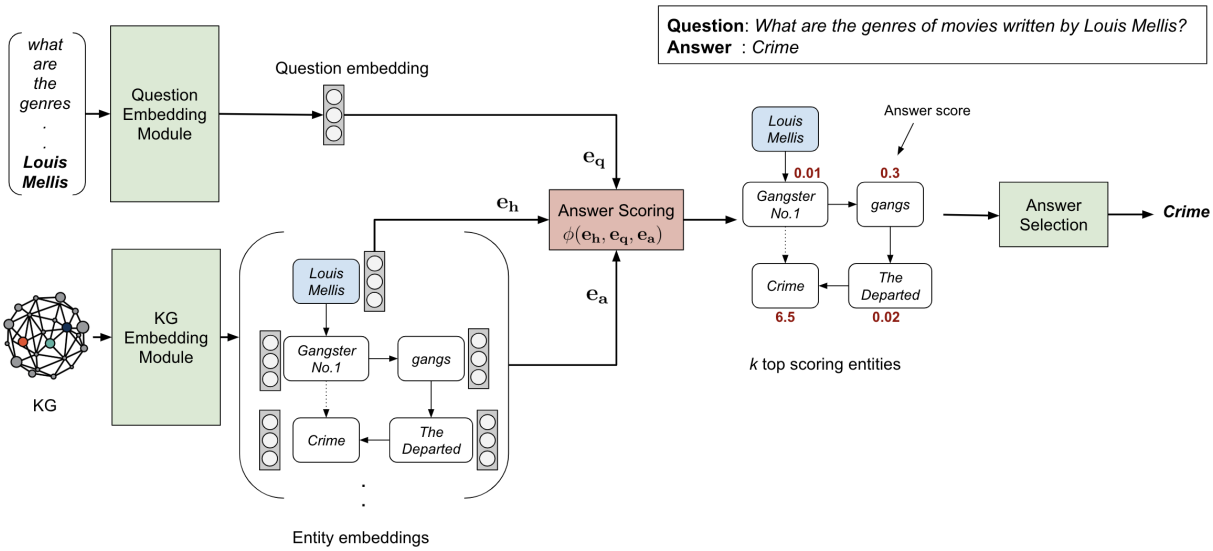

Figure 2: **Overview of EmbedKGQA**, our proposed method for Multi-hop QA over Knowledge Graphs (KGQA). EmbedKGQA has three modules: (1) *KG Embedding Module* (Section 4.2) learns embeddings for all entities in the input KG, (2) *Question Embedding Module* (Section 4.3) learns an embedding for the question, and (3) the *Answer Selection Module* (Section 4.4) selects the final answer by incorporating the question and relation similarity scores. EmbedKGQA's use of embeddings makes it more effective in handling KG sparsity. Moreover, since EmbedKGQA considers all entities as candidate answers, it doesn't suffer from the limited neighborhood out-of-reach issues of existing Multi-hop KGQA methods. Please refer Section 4 for detailed description of EmbedKGQA.

Graph CNN (Kipf and Welling, 2016) is then applied over this augmented sub-graph to arrive at the final answer. Unfortunately, availability and identification of relevant text corpora is a challenge on its own which limits broad-coverage applicability of such methods. Moreover, such methods also impose pre-specified heuristic neighborhood size limitation from which the true answer needs to be selected. This often makes the true answer out of reach of the model to select from.

In order to illustrate these points, please consider the example shown in Figure 1. In this example, *Louis Mellis* is the head entity in the input NL question, and *Crime* is the true answer we expect the model to select. If the edge *has_genre(Gangster No. 1, Crime)* were present in the KG, then the question could have been answered rather easily. However, since this edge is missing from the KG, as is often the case with similar incomplete and sparse KGs, the KGQA model has to potentially reason over a longer path over the KG (marked by bolded edges in the graph). Moreover, the KGQA model imposed a neighborhood size of 3-hops, which made the true answer *Crime* out of reach.

In a separate line of research, there has been a large body of work that utilizes KG embeddings to predict missing links in the KG, thereby reducing KG sparsity (Bordes et al., 2013; Trouillon et al., 2016; Yang et al., 2014a; Nickel et al., 2011). KG embedding methods learn high-dimensional embeddings for entities and relations in the KG, which are then used for link prediction. In spite of its high relevance, KG embedding methods have not been used for multi-hop KGQA – we fill this gap in this paper. In particular, we propose **EmbedKGQA**, a novel system which leverages KG embeddings to perform multi-hop KGQA. We make the following contributions in this paper:

1. We propose EmbedKGQA, a novel method for the multi-hop KGQA task. To the best of our knowledge, EmbedKGQA is the first method to use KG embeddings for this task. EmbedKGQA is particularly effective in performing multi-hop KGQA over sparse KGs.

2. EmbedKGQA relaxes the requirement of answer selection from a pre-specified local neighborhood, an undesirable constraint imposed by previous methods for this task.

3. Through extensive experiments on multiple real-world datasets, we demonstrate EmbedKGQA's effectiveness over state-of-the-art baselines.

We have made EmbedKGQA's source code available to encourage reproducibility.

2 Related Work

KGQA: In prior work (Li et al., 2018) TransE, (Bordes et al., 2013) embeddings have been used to answer factoid based questions. However, this requires ground truth relation labeling for each question and it does not work for multi-hop question answering. In another line of work (Yih et al., 2015) and (Bao et al., 2016) proposed extracting a particular sub-graph to answer the question. The method presented in (Bordes et al., 2014a), the sub-graph generated for a head entity is projected in a high dimensional space for question answering. Memory Networks have also been used to learn high dimensional embeddings of the facts present in the KG to perform QA (Bordes et al., 2015). Methods like (Bordes et al., 2014b) learn a similarity function between the question and the corresponding triple during training, and score the question with all the candidate triples at the test time. (Yang et al., 2014b) and (Yang et al., 2015) utilize embedding based methods to map natural language questions to logical forms. Methods like (Dai et al., 2016; Dong et al., 2015; Hao et al., 2017; Lukovnikov et al., 2017; Yin et al., 2016) utilize neural networks to learn a scoring functions to rank the candidate answers. Some works like (Mohammed et al., 2017; Ture and Jojic, 2016) consider each relation as a label and model QA task as a classification problem. Extending these kinds of approaches for multi-hop question answering is non-trivial.

Recently, there has been some work in which text corpus is incorporated as a knowledge source in addition to KG to answer complex questions on KGs (Sun et al., 2018, 2019a). Such approaches are useful in case the KG is incomplete. However, this leads to another level of complexity in the QA system, and text corpora might not always be available.

KG completion methods: Link prediction in Knowledge Graphs using KG embeddings has become a popular area of research in recent years. The general framework is to define a score function for a set of triples (h, r, t) in a KG and constraining them in such a way that the score for a correct triple is higher than the score for an incorrect triple.

RESCAL (Nickel et al., 2011) and DistMult (Yang et al., 2015) learn a score function containing a bi-linear product between head entity and tail entity vectors and a relation matrix. ComplEx (Trouillon et al., 2016) represents entity vectors and relation matrices in the complex space. SimplE (Kazemi and Poole, 2018) and TuckER (Balažević et al., 2019) are based on Canonical Polyadic (CP) decomposition (Hitchcock, 1927) and Tucker decomposition (Tucker, 1966) respectively.

TransE (Bordes et al., 2013) embeds entities in high dimensional real space and relation as translation between the head and the tail entities. RotatE (Sun et al., 2019b) on the other hand projects entities in complex space and relations are represented as rotations in the complex plane.

ConvE (Dettmers et al., 2018) utilizes Convolutional Neural Networks to learn a scoring function between the head entity, tail entity and relation. InteractE (Vashishth et al., 2019) improves upon ConvE by increasing feature interaction.

3 Background

In this section, we formally define a Knowledge Graph(KG) and then describe link prediction task on incomplete KGs. We then describe KG embeddings and explain the ComplEx embedding model.

3.1 Knowledge Graph

Given a set of entities $\mathcal{E}$ and relations $\mathcal{R}$, a Knowledge Graph $\mathcal{G}$ is a set of triples $\mathcal{K}$ such that $\mathcal{K} \subseteq \mathcal{E} \times \mathcal{R} \times \mathcal{E}$. A triple is represented as (h, r, t), with $h, t \in \mathcal{E}$ denoting subject and object entities respectively and $r \in \mathcal{R}$ the relation between them.

3.2 Link Prediction

In link prediction, given an incomplete Knowledge Graph, the task is to predict which unknown links are valid. KG Embedding models achieve this through a *scoring function* ϕ that assigns a score $s = \phi(h, r, t) \in \mathbb{R}$, which indicates whether a triple is true, with the goal of being able to score all missing triples correctly.

3.3 Knowledge Graph Embeddings

For each $e \in \mathcal{E}$ and $r \in \mathcal{R}$, Knowledge Graph Embedding (KGE) models generate $e_e \in \mathbb{R}^{d_e}$ and $e_r \in \mathbb{R}^{d_r}$, where e_e and e_r are d_e and d_r dimensional vectors respectively. Each of the embedding methods also has a scoring function $\phi : \mathcal{E} \times \mathcal{R} \times \mathcal{E} \rightarrow \mathbb{R}$ to assign some score $\phi(h, r, t)$ to a possible triple (h, r, t), $h, t \in \mathcal{E}$ and $r \in \mathcal{R}$. Models are trained in a way such that for every correct triple $(h, r, t) \in \mathcal{K}$ and incorrect triple $(h', r', t') \notin \mathcal{K}$ the model assign scores such that $\phi(h, r, t) > 0$ and $\phi(h', r', t') < 0$. A scoring function is generally a function of (e_h, e_r, e_t).

3.3.1 ComplEx Embeddings

ComplEx (Trouillon et al., 2016) is a tensor factorization approach that embeds relations and entities in complex space. Given $h, t \in \mathcal{E}$ and $r \in \mathcal{R}$, ComplEx generates $e_h, e_r, e_t \in \mathbb{C}^d$ and defines a scoring function:

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

such that $\phi(h, r, t) > 0$ for all true triples, and $\phi(h, r, t) < 0$ for false triples. Re denotes the real part of a complex number.

4 EmbedKGQA: Proposed Method

In this section, we first define the problem of KGQA and then describe our model.

4.1 Problem Statement

Let $\mathcal{E}$ and $\mathcal{R}$ be the set of all entities and relations respectively in a KG $\mathcal{G}$, and $\mathcal{K} \subseteq \mathcal{E} \times \mathcal{R} \times \mathcal{E}$ is the set of all available KG facts. The problem in KGQA involves, given a natural language question q and a topic entity $e_h \in \mathcal{E}$ present in the question, the task is to extract an entity $e_t \in \mathcal{E}$ that correctly answers the question q.

4.1.1 EmbedKGQA Overview

We work in a setting where there is no fine-grained annotation present in the dataset, such as the question type or the exact logic reasoning steps. For example, co-actor is a combination of $starred\_actor^{-1}$ and $starred\_actor$ relations, but our model does not require this annotation.

EmbedKGQA uses Knowledge Graph embeddings to answer multi-hop natural language questions. First it learns a representation of the KG in an embedding space. Then given a question it learns a question embedding. Finally it combines these embedding to predict the answer.

In the following sections, we introduce the EmbedKGQA model. It consists of 3 modules:

1. **KG Embedding Module** creates embeddings for all entities in the KG.
2. **Question Embedding Module** finds the embedding of a question
3. **Answer Selection Module** reduces the set of candidate answer entities and selects the final answer

4.2 KG Embedding Module

ComplEx embeddings are trained for all $h, t \in \mathcal{E}$ and all $r \in \mathcal{R}$ in the KG such that $e_h, e_r, e_t \in \mathbb{C}^d$. The entity embeddings are used for learning a triple scoring function between the head entity, question, and answer entity. Based on the coverage of the KG entities in the QA training set, the entity embeddings learned here are either kept frozen or allowed to be fine-tuned in the subsequent steps.

4.3 Question Embedding Module

This module embeds the natural language question q to a fixed dimension vector $e_q \in \mathbb{C}^d$. This is done using a feed-forward neural network that first embeds the question q using RoBERTa (Liu et al., 2019) into a 768-dimensional vector. This is then passed through 4 fully connected linear layers with ReLU activation and finally projected onto the complex space $\mathbb{C}^d$.

Given a question q, topic entity $h \in \mathcal{E}$ and set of answer entities $\mathcal{A} \subseteq \mathcal{E}$, it learns the question embedding in a way such that

$$\phi(e_h, e_q, e_a) > 0 \quad \forall a \in \mathcal{A}$$

$$\phi(e_h, e_q, e_{\bar{a}}) < 0 \quad \forall \bar{a} \notin \mathcal{A}$$

where ϕ is the ComplEx scoring function (1) and $e_a, e_{\bar{a}}$ are entity embeddings learnt in the previous step.

For each question, the score $\phi(.)$ is calculated with all the candidate answer entities $a' \in \mathcal{E}$. The model is learned by minimizing the binary cross-entropy loss between the sigmoid of the scores and the target labels, where the target label is 1 for the correct answers and 0 otherwise. Label smoothing is done when the total number of entities is large.

4.4 Answer Selection Module

At inference, the model scores the (head, question) pair against all possible answers $a' \in \mathcal{E}$. For relatively smaller KGs like MetaQA, we simply select the entity with the highest score.

$$e_{ans} = \arg\max_{a' \in \mathcal{E}} \phi(e_h, e_q, e_{a'})$$

However if the knowledge graph is large, pruning the candidate entities can significantly improve the performance of EmbedKGQA. The pruning strategy is described in the following section.

	Train	Dev	Test
MetaQA 1-hop	96,106	9,992	9.947
MetaQA 2-hop	118,948	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 WebQuestionsSP datasets. Please refer section 5.1 for more details.

4.4.1 Relation matching

Similar to PullNet (Sun et al., 2019a) we learn a scoring function $S(r, q)$ which ranks each relation $r \in \mathcal{R}$ for a given question q. Let h_r be the embedding of a relation r and $q' = (< \text{s} >, w_1, .., w_{|q|}, < /\text{s} >)$ be the sequence of words in question q which are input to RoBERTa. The scoring function is defined as the sigmoid of the dot product of the final output of the last hidden layer of RoBERTa (h_q) and the embedding of relation r (h_r).

$$h_q = \text{RoBERTa}(q')$$
$$S(r, q) = \text{sigmoid}(h_q^T h_r)$$

Among all the relations, we select those relations which have score greater than 0.5 It is denoted as the set $\mathcal{R}_a$. For each candidate entity a' that we have obtained so far (Section 4.4), we find the relations in the shortest path between head entity h and a'. Let this set of relations be $\mathcal{R}_{a'}$. Now the relation score for each candidate answer entity is defined as the size of their intersection.

$$\text{RelScore}_{a'} = |\mathcal{R}_a \cap \mathcal{R}_{a'}|$$

We use a linear combination of the relation score and ComplEx score to find the answer entity.

$$e_{ans} = \arg\max_{a' \in \mathcal{N}_h} \phi(e_h, e_q, e_{a'}) + \gamma * \text{RelScore}_{a'}$$

where γ is a tunable hyperparameter.

5 Experimental Details

In this section, we first describe the datasets that we evaluated our method on, and then explain the experimental setup and the results.

5.1 Datasets

1. **MetaQA** (Zhang et al., 2018) dataset is a large scale multi-hop KGQA dataset with more than 400k questions in the movie domain. It has 1-hop, 2-hop, and 3-hop questions. In our experiments, we used the "vanilla" version of the questions. Along with the QA data, MetaQA also provides a KG with 135k triples, 43k entities, and nine relations.

2. **WebQuestionsSP** (tau Yih et al., 2016) is a smaller QA dataset with 4,737 questions. The questions in this dataset are 1-hop and 2-hop questions and are answerable through Freebase KG. For ease of experimentation, we restrict the KB to be a subset of Freebase which contains all facts that are within 2-hops of any entity mentioned in the questions of WebQuestionsSP. We further prune it to contain only those relations that are mentioned in the dataset. This smaller KB has 1.8 million entities and 5.7 million triples.

5.2 Baselines

We compare our model with the Key-Value Memory Network (Miller et al., 2016), the GraftNet (Sun et al., 2018) and the Pullnet (Sun et al., 2019a) for WebQuestionsSP dataset. For MetaQA dataset we also compare with the VRN (Zhang et al., 2018). These methods implement multi-hop KGQA, and except VRN, use additional text corpus to mitigate the KG sparsity problem.

- **VRN** (Zhang et al., 2018) uses variational learning algorithm to perform Multi-Hop QA over KG.

- **Key-Value Memory Network (KVMem)** (Miller et al., 2016) is one of the first models that attempts to do QA over incomplete KBs by augmenting it with text. It maintains a memory table which stores KB facts and text encoded into key-value pairs and uses this for retrieval.

- **GraftNet** (Sun et al., 2018) uses heuristics to create a question-specific subgraph containing KG facts, entities and sentences from the text corpora and then uses a variant of graph CNN (Kipf and Welling, 2016) to perform reasoning over it.

- **PullNet** (Sun et al., 2019a) also creates a question-specific sub-graph but instead of using heuristics, it learns to "pull" facts and sentences from the data to create a more relevant

Model	MetaQA KG-Full			MetaQA KG-50		
	1-hop	2-hop	3-hop	1-hop	2-hop	3-hop
VRN	**97.5**	89.9	62.5	-	-	-
GraftNet	97.0	94.8	77.7	64.0 (91.5)	52.6 (69.5)	59.2 (66.4)
PullNet	97.0	**99.9**	91.4	65.1 (92.4)	52.1 (90.4)	59.7 (85.2)
KV-Mem	96.2	82.7	48.9	63.6 (75.7)	41.8 (48.4)	37.6 (35.2)
EmbedKGQA (Ours)	**97.5**	98.8	**94.8**	**83.9**	**91.8**	**70.3**

Table 2: Results on MetaQA dataset. All baseline results were taken from Sun et al. (2019a). We have considered both full KG (**MetaQA KG-Full**) and 50% KG (**MetaQA KG-50**) settings. The numbers reported in this table are hits@1. Numbers in brackets correspond to a setting where text was used to augment the incomplete KG (**MetaQA KG-50**). For more details please refer section 5.3.1.

sub-graph. It also uses a graph CNN approach to perform reasoning.

The complete KG setting is the easiest setting for QA because the datasets are created in such a way that the answer always exists in the KG, and there is no missing link in the path. However, it is not a realistic setting, and the QA model should also be able to work on an incomplete KG. So we simulate an incomplete KB by randomly removing half of the triples in the KB (we randomly drop a fact with probability = 0.5). We call this setting **KG-50** and we call full KG setting **KG-Full** in the text.

In the next section we will answer the following questions:

Q1. Can Knowledge Graph embeddings be used to perform multi-hop KGQA? (Section 5.3)

Q2. Can EmbedKGQA be used to answer questions when there is no direct path between the head entity and the answer entity? (Section 5.4)

Q3. How much does the answer selection module help in the final performance of our model? (Section 5.5)

5.3 KGQA results

In this section, we have compared our model with baseline models on MetaQA and WebQuestionsSP datasets.

5.3.1 Analysis on MetaQA

MetaQA has different partitions of the dataset for 1-hop, 2-hop, and 3-hop questions. In the full KG setting (MetaQA KG-Full) our model is comparable to the state-of-the-art for 2-hop questions and establishes the state-of-the-art for 3-hop questions. EmbedKGQA performs similar to the state-of-the in case of 1-hop question which is expected because the answer node is directly connected to the head node and it is able to learn the corresponding relation embedding from the question. On the other hand performance on 2-hop and 3-hop questions suggest that EmbedKGQA is able to infer the correct relation from the neighboring edges because the KG embeddings can model composition of relations. Pullnet and GraftNet also perform similarly well because the answer entity lies in the question sub-graph most of the times.

We have also tested our method on the incomplete KG setting, as explained in the previous section. Here we find that the accuracy of all baselines decreases significantly compared to the full KG setting, while EmbedKGQA achieves state-of-the-art performance. This is because MetaQA KG is fairly sparse, with only 135k triples for 43k entities. So when 50% of the triples are removed (as is done in MetaQA KG-50), the graph becomes very sparse with an average of only 1.66 links per entity node. This causes many head entity nodes of questions to have much longer paths (>3) to their answer node. Hence models that require question-specific sub-graph construction (GraftNet, PullNet) are unable to recall the answer entity in their generated sub-graph and therefore performs poorly. However, their performance improves only after including additional text corpora. On the other hand, EmbedKGQA does not limit itself to a sub-graph and utilizing the link prediction properties the KG embeddings, EmbedKGQA is able to infer the relation on missing links.

5.3.2 Analysis on WebQuestionsSP

WebQuestionsSP has a relatively small number of training examples but uses a large KG (Freebase) as background knowledge. This makes multi-hop KGQA much harder. Since all the entities of the KG are not covered in the training set, freezing the

Model	WebQSP KG-Full	WebQSP KG-50
KV-Mem	46.7	32.7 (31.6)
GraftNet	66.4	48.2 (49.7)
PullNet	**68.1**	50.1 (51.9)
EmbedKGQA	66.6	**53.2**

Table 3: Performance on WebQuestionsSP dataset. All baseline results were taken from Sun et al. (2019a). The values reported are hits@1. Numbers in brackets correspond to a setting where text was used to augment the incomplete KG (WebQSP KG-50). For more details please refer Section 5.3.2.

entity embeddings after learning them during KG embedding learning phase (Section 4.2) is necessary. Results on WebQuestionsSP (Table 3) highlight the fact that, even with a small number of training examples EmbedKGQA can learn good question embeddings that can infer the multi-hop path required to answer the questions.

Our method on WebQSP KG-50 outperforms all baselines including PullNet, which uses extra textual information and is the state-of-the-art model. Even though WebQuestionsSP has fewer questions, EmbedKGQA is able to learn good question embeddings that can infer mission links in KG. This can be attributed to the fact that relevant and necessary information is being captured through KG embeddings, implicitly.

5.4 QA on KG with missing links

State-of-the-art KGQA models like PullNet and GraftNet require a path between the head entity and the answer entity to be present in the Knowledge Graph to answer the question. For example, in PullNet, the answer is restricted to be one of the entities present in the extracted question subgraph. For the incomplete KG case where only 50% of the original triples are present, PullNet (Sun et al., 2019a) reports a recall of 0.544 on the MetaQA 1-hop dataset. This means that only for 54.4 percent of questions, all the answer entities are present in the extracted question subgraph, and this puts a hard limit on how many questions the model can answer in this setting.

EmbedKGQA, on the other hand, uses Knowledge Graph Embeddings rather than a localized sub-graph to answer the question. It uses the head embedding and question embedding, which implicitly captures the knowledge of all observed and unobserved links around the head node. This is possible because of the link prediction property of

Model	Accuracy
ComplEx	20.1
EmbedKGQA	**29.9**

Table 4: QA results on MetaQA 1-hop for the experiments in which there is no link between head entity and answer entity. We have compared the results with the KG completion methods in which gold relation of the question is known. The details are provided in Section 5.4.

Model	WebQSP KG-Full	WebQSP KG-50
EmbedKGQA	66.6	53.2
{+ 2-hop filtering}	72.5	51.8
{+ 2-hop filtering, – Relation matching}	58.7	48.5
{– Relation matching}	48.1	47.4

Table 5: This table show the importance of relation matching module (Section 4.4.1) and effect of neighbourhood based filtering on EmbedKGQA in the WebQuestionsSP dataset. EmbedKGQA in itself contains the relation matching module. Here we try to see the effect of ablating the relation matching module and adding a 2-hop neighbourhood filtering during answer selection. Please refer Section 5.5 for more details.

Knowledge Graph Embeddings.

So unlike other QA systems, even if there is no path between the head and answer entity, our model should be able to answer the question if there is sufficient information in the KG to be able to predict that path (See Fig. 1).

We design an experiment to test this capability of our model. For all questions in the validation set of the MetaQA 1-hop dataset, we removed all the triples from the Knowledge Graph that can be directly used to answer the question. For example, given the question '*what language is [PK] in*' in the validation set, we removed the triple $(PK, in\_language, Hindi)$ from the KG. The dataset also contains paraphrases of the same question, for, e.g., '*what language is the movie [PK] in*' and '*what is the language spoken in the movie [PK]*'. We also removed all paraphrases of validation set questions from the training dataset since we only want to evaluate the KG completion property of our model and not a linguistic generalization.

In such a setting, we expect models that rely only on sub-graph retrieval to achieve 0 hits@1. However, our model delivers a significantly better 29.9 hits@1 in this setting. This shows that our

model can capture the KG completion property of ComplEx embeddings and apply it to answer questions which was otherwise impossible.

Further, if we know the relation corresponding to each question, then the problem of 1-hop KG QA is the same as KG completion in an incomplete Knowledge Graph. Using the same training KG as above and using the removed triples as the test set, we do tail prediction using KG embeddings. Here we obtain 20.1 hits@1. The lesser score can be attributed to the fact that ComplEx embedding uses only the KG while our model uses the QA data as well - which in itself represents knowledge. Our model is first trained on the KG and then uses these embeddings to train the QA model, and thus it can leverage the knowledge present in both the KG and QA data.

5.5 Effect of Answer Selection Module

We analyse the effect of the answer selection module (Section 4.4) on EmbedKGQA in the WebQuestionsSP dataset by ablating the relation matching module. Furthermore, in order to compare with other methods that restrict the answer to a neighbourhood in the KG (Sun et al. (2019a), Sun et al. (2018)), we experimented with restricting the candidate set of answer entities to only the **2-hop neighbourhood** of the head entity. The results can be seen in Table 5. As we can see, relation matching has a significant impact on the performance of EmbedKGQA on both WebQSP KG-full and WebQSP KG-50 settings.

Also, as mentioned earlier, WebQSP KG (Freebase subset) has an order of magnitude more entities than MetaQA (1.8M versus 134k in MetaQA) and the number of possible answers is large. So reducing the set of answers to a 2-hop neighbourhood of the head entity showed improved performance in the case of WebQSP KG-Full. However, this caused a *degradation* in performance on WebQSP KG-50. This is because restricting the answer to a 2-hop neighbourhood on an incomplete KG may cause the answer to not be present in the candidates (Please refer figure 1).

In summary, we find that relation matching is an important part of EmbedKGQA. Morever, we suggest that n-hop filtering during answer selection may be included on top of EmbedKGQA for KGs which are reasonably complete.

6 Conclusion

In this paper, we propose EmbedKGQA, a novel method for Multi-hop KGQA. KGs are often incomplete and sparse which poses additional challenges for multi-hop KGQA methods. Recent recent for this problem have tried to address the incompleteness problem by utilizing an additional text corpus. However, the availability of a relevant text corpus is often limited, thereby reducing broad-coverage applicability of such methods. In a separate line of research, KG embedding methods have been proposed to reduce KG sparsity by performing missing link prediction. EmbedKGQA utilizes the link prediction properties of KG embeddings to mitigate the KG incompleteness problem without using any additional data. It trains the KG entity embeddings and uses it to learn question embeddings, and during the evaluation, it scores (head entity, question) pair again all entities, and the highest-scoring entity is selected as an answer. EmbedKGQA also overcomes the shortcomings due to limited neighborhood size constraint imposed by existing multi-hop KGQA methods. EmbedKGQA achieves state-of-the-art performance in multiple KGQA settings, suggesting that the link prediction properties of KG embeddings can be utilized to mitigate the KG incompleteness problem in Multi-hop KGQA.

Acknowledgements

We would like to thank the anonymous reviewers for their constructive feedback, and Ashutosh Kumar, Aditya Rastogi and Chandrahas from the Indian Institute of Science for their insightful comments. This research is supported in part by a grant from Intel and the Ministry of Human Resource Development, Government of India.

References

Ivana Balažević, Carl Allen, and Timothy M Hospedales. 2019. Tucker: Tensor factorization for knowledge graph completion. *arXiv preprint arXiv:1901.09590*.

Junwei Bao, Nan Duan, Zhao Yan, Ming Zhou, and Tiejun Zhao. 2016. Constraint-based question answering with knowledge graph. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers*, pages 2503–2514.

Antoine Bordes, Sumit Chopra, and Jason Weston.

2014a. Question answering with subgraph embeddings. *arXiv preprint arXiv:1406.3676*.

Antoine Bordes, Nicolas Usunier, Sumit Chopra, and Jason Weston. 2015. Large-scale simple question answering with memory networks. *arXiv preprint arXiv:1506.02075*.

Antoine Bordes, Nicolas Usunier, Alberto Garcia-Duran, Jason Weston, and Oksana Yakhnenko. 2013. Translating embeddings for modeling multi-relational data. In *Advances in neural information processing systems*, pages 2787–2795.

Antoine Bordes, Jason Weston, and Nicolas Usunier. 2014b. Open question answering with weakly supervised embedding models. In *Joint European conference on machine learning and knowledge discovery in databases*, pages 165–180. Springer.

Zihang Dai, Lei Li, and Wei Xu. 2016. Cfo: Conditional focused neural question answering with large-scale knowledge bases. *arXiv preprint arXiv:1606.01994*.

Tim Dettmers, Pasquale Minervini, Pontus Stenetorp, and Sebastian Riedel. 2018. Convolutional 2d knowledge graph embeddings. In *Thirty-Second AAAI Conference on Artificial Intelligence*.

Li Dong, Furu Wei, Ming Zhou, and Ke Xu. 2015. Question answering over freebase with multi-column convolutional neural networks. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 260–269.

Google. 2013. Freebase data dumps. https://developers.google.com/freebase/data.

Yanchao Hao, Yuanzhe Zhang, Kang Liu, Shizhu He, Zhanyi Liu, Hua Wu, and Jun Zhao. 2017. An end-to-end model for question answering over knowledge base with cross-attention combining global knowledge. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 221–231.

Frank L Hitchcock. 1927. The expression of a tensor or a polyadic as a sum of products. *Journal of Mathematics and Physics*, 6(1-4):164–189.

Seyed Mehran Kazemi and David Poole. 2018. Simple embedding for link prediction in knowledge graphs. In *Advances in Neural Information Processing Systems*, pages 4284–4295.

Thomas N Kipf and Max Welling. 2016. Semi-supervised classification with graph convolutional networks. *arXiv preprint arXiv:1609.02907*.

Jens Lehmann, Robert Isele, Max Jakob, Anja Jentzsch, Dimitris Kontokostas, Pablo N Mendes, Sebastian Hellmann, Mohamed Morsey, Patrick Van Kleef, Sören Auer, et al. 2015. Dbpedia–a large-scale, multilingual knowledge base extracted from wikipedia. *Semantic Web*, 6(2):167–195.

Dingcheng Li, Jingyuan Zhang, and Ping Li. 2018. Representation learning for question classification via topic sparse autoencoder and entity embedding. In *2018 IEEE International Conference on Big Data (Big Data)*, pages 126–133. IEEE.

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*.

Denis Lukovnikov, Asja Fischer, Jens Lehmann, and Sören Auer. 2017. Neural network-based question answering over knowledge graphs on word and character level. In *Proceedings of the 26th international conference on World Wide Web*, pages 1211–1220. International World Wide Web Conferences Steering Committee.

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*.

Tom M. Mitchell, William W. Cohen, Estevam R. Hruschka Jr., Partha P. Talukdar, Bo Yang, Justin Betteridge, Andrew Carlson, Bhavana Dalvi Mishra, Matt Gardner, Bryan Kisiel, Jayant Krishnamurthy, Ni Lao, Kathryn Mazaitis, Thahir Mohamed, Ndapandula Nakashole, Emmanouil A. Platanios, Alan Ritter, Mehdi Samadi, Burr Settles, Richard C. Wang, Derry Wijaya, Abhinav Gupta, Xinlei Chen, Abulhair Saparov, Malcolm Greaves, and Joel Welling. 2018. Never-ending learning. *Commun. ACM*, 61(5):103–115.

Salman Mohammed, Peng Shi, and Jimmy Lin. 2017. Strong baselines for simple question answering over knowledge graphs with and without neural networks. *arXiv preprint arXiv:1712.01969*.

Maximilian Nickel, Volker Tresp, and Hans-Peter Kriegel. 2011. A three-way model for collective learning on multi-relational data. In *ICML*, volume 11, pages 809–816.

Fabian M Suchanek, Gjergji Kasneci, and Gerhard Weikum. 2007. Yago: a core of semantic knowledge. In *Proceedings of the 16th international conference on World Wide Web*, pages 697–706. ACM.

Haitian Sun, Tania Bedrax-Weiss, and William W Cohen. 2019a. 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*.

Zhiqing Sun, Zhi-Hong Deng, Jian-Yun Nie, and Jian Tang. 2019b. Rotate: Knowledge graph embedding by relational rotation in complex space. *arXiv preprint arXiv:1902.10197*.

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.

Ledyard R Tucker. 1966. Some mathematical notes on three-mode factor analysis. *Psychometrika*, 31(3):279–311.

Ferhan Ture and Oliver Jojic. 2016. No need to pay attention: Simple recurrent neural networks work!(for answering" simple" questions). *arXiv preprint arXiv:1606.05029*.

Shikhar Vashishth, Soumya Sanyal, Vikram Nitin, Nilesh Agrawal, and Partha Talukdar. 2019. Interacte: Improving convolution-based knowledge graph embeddings by increasing feature interactions. *arXiv preprint arXiv:1911.00219*.

Bishan Yang, Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. 2014a. Embedding entities and relations for learning and inference in knowledge bases. *arXiv preprint arXiv:1412.6575*.

Min-Chul Yang, Nan Duan, Ming Zhou, and Hae-Chang Rim. 2014b. Joint relational embeddings for knowledge-based question answering. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 645–650.

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.

Scott Wen-tau Yih, Ming-Wei Chang, Xiaodong He, and Jianfeng Gao. 2015. Semantic parsing via staged query graph generation: Question answering with knowledge base.

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 *ACL*.

Wenpeng Yin, Mo Yu, Bing Xiang, Bowen Zhou, and Hinrich Schütze. 2016. Simple question answering by attentive convolutional neural network. *arXiv preprint arXiv:1606.03391*.

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*.