



# QAGCN: Answering Multi-relation Questions via Single-Step Implicit Reasoning over Knowledge Graphs

Ruijie Wang<sup>1,2</sup>(✉) , Luca Rossetto<sup>1</sup> , Michael Cochez<sup>3,4</sup> ,  
and Abraham Bernstein<sup>1</sup>

<sup>1</sup> Department of Informatics, University of Zurich, Zurich, Switzerland  
{ruijie,rossetto,bernstein}@ifi.uzh.ch

<sup>2</sup> University Research Priority Program “Dynamics of Healthy Aging”,  
University of Zurich, Zurich, Switzerland

<sup>3</sup> Vrije Universiteit Amsterdam, Amsterdam, The Netherlands  
m.cochez@vu.nl

<sup>4</sup> Elsevier Discovery Lab, Amsterdam, The Netherlands

**Abstract.** Multi-relation question answering (QA) is a challenging task, where given questions usually require long reasoning chains in KGs that consist of multiple relations. Recently, methods with explicit multi-step reasoning over KGs have been prominently used in this task and have demonstrated promising performance. Examples include methods that perform stepwise label propagation through KG triples and methods that navigate over KG triples based on reinforcement learning. A main weakness of these methods is that their reasoning mechanisms are usually complex and difficult to implement or train. In this paper, we argue that multi-relation QA can be achieved via end-to-end single-step implicit reasoning, which is simpler, more efficient, and easier to adopt. We propose QAGCN — a Question-Aware Graph Convolutional Network (GCN)-based method that includes a novel GCN architecture with controlled question-dependent message propagation for the implicit reasoning. Extensive experiments have been conducted, where QAGCN achieved competitive and even superior performance compared to state-of-the-art explicit-reasoning methods. Our code and pre-trained models are available in the repository: <https://github.com/ruijie-wang-uzh/QAGCN>.

**Keywords:** Multi-relation Question Answering · Knowledge Graph · Graph Convolutional Network

## 1 Introduction

Question answering (QA) over knowledge graphs (KGs)—a task that has been pursued almost since the inception of the Semantic Web (cf. [13])—aims to automatically retrieve answers from KGs for given natural language questions.

In recent years, multi-relation questions that require reasoning chains over multiple KG triples have been a focus of this task. An example is “*who is the mayor of the city where the director of Sleepy Hollow was born,*” which mentions one topic entity (“*Sleepy Hollow*”) and three relations (“*mayor of*”, “*director of*”, and “*was born*”). Correspondingly, starting from the entity **Sleepy Hollow**, the expected answer **Jess Talamantes** can be inferred from a reasoning chain that consists of three triples: [(**Sleepy Hollow**, **director**, **Tim Burton**), (**Tim Burton**, **birthplace**, **Burbank**), (**Burbank**, **mayor**, **Jess Talamantes**)]. To answer this kind of question, methods with reasoning mechanisms [5, 21, 24, 31, 32] have been proposed to infer the reasoning chains over KGs step by step. They typically commence with topic entities in given questions as anchors and try to infer reasoning chains by extending triples according to the semantics of the questions. This extension is usually performed in two ways: label propagation from entities in the reasoning chain of the current step to other entities that can be added in the next step, and reinforcement learning-based decision-making on choosing entities to add. Given that the reasoning involves multiple steps and produces explicit states of reasoning chains at each step, we refer to these methods as explicit multi-step reasoning-based. Recently, they have achieved state-of-the-art (SOTA) performance in the task. However, their reasoning mechanisms are often complex and difficult to implement or train. Furthermore, for better supervision during training, some of these methods [32] require annotations of reasoning paths, which are usually unavailable in real-world scenarios.

Inspired by the promising performance that graph convolutional networks (GCNs) [22] achieved in learning semantic representations of KG entities, we propose to adapt GCNs for answering multi-relation questions with single-step implicit reasoning that is simpler, more efficient, and easier to adopt than existing reasoning mechanisms. Specifically, in this paper, we propose a novel Question-Aware GCN-based QA method, called QAGCN, which encodes questions and KG entities in a joint embedding space where questions are close to correct answers (entities). The intuition of our method is as follows: Given a question, if an entity is the correct answer, the reasoning chain of this question would be part of the KG context (i.e., the neighboring triples) of that entity. For example, considering the above question, the reasoning chain is part of the context of **Jess Talamantes** within three hops in the KG. During encoding, we expect the GCN to focus only on messages that pass through the reasoning path and accumulate the messages in the representation of this entity. We refer to this process as question-aware message propagation. In this case, the representation of the entity would contain information that is semantically consistent with the given question and can be aligned with the question in the embedding space. The GCN directly generates semantic representations of the question and KG entities in an end-to-end fashion. Therefore, we classify it as single-step reasoning. Also, the reasoning is implicit, given that it is hidden in the message propagation of the GCN. It is very challenging to achieve this GCN-based reasoning, as the potential KG context of entities could be very large, especially when several hops need to be considered for multi-relation questions.

In summary, we make the following contributions in this paper:

- We propose a novel QA method called QAGCN that can answer multi-relation questions via single-step implicit reasoning. The method is simpler, more efficient, and easier to adopt than existing reasoning-based methods.
- We propose a novel question-aware GCN architecture with contextually controlled message propagation for implicit reasoning.
- We conducted extensive experiments to evaluate the effectiveness and efficiency of QAGCN and demonstrate its competitive and even superior performance compared with recent SOTA reasoning-based methods.

## 2 Related Work

In this section, we first give a detailed introduction to existing reasoning-based methods. Then, we provide a general overview of other QA methods.

As aforementioned, there are primarily two types of reasoning-based methods: label propagation-based and reinforcement learning-based. Examples of label propagation-based methods include NSM [9], SR+NSM [30], and TransferNet [24]. NSM and SR+NSM label entities in the underlying KG with their probabilities of being part of the reasoning chain of a given question. In the first step, only topic entities are labeled with positive probabilities. During reasoning, the probabilities are expected to be passed through the reasoning chain to final answers. TransferNet also starts with labeling topic entities. It leverages the semantics of relations in the KG to perform the transfer of labels between entities. Examples of reinforcement learning-based methods include MINERVA [5], IRN [32], and SRN [21]. MINERVA trains a reinforcement learning agent that walks on KGs from entity to entity through the edges linking them. Starting from topic entities, the agent is trained to arrive at correct answers after several walking steps. IRN tries to infer chains of relations that could link topic entities to correct answers based on jointly learned KG embeddings and question encodings. SRN employs attentional question encoding and potential-based reward shaping in the training of the reinforcement learning agent. These methods are the most relevant to our work and will be used as baselines in experiments.

Other important types of QA methods include parsing-based [1, 11, 13, 27, 29] and embedding-based [3, 4, 12, 17, 23, 26]. Parsing-based methods construct logical queries, such as SPARQL [8] and  $\lambda$ -DCS [16], that represent the semantics of given questions and can be evaluated to retrieve answers from KGs. They can provide parsing results of questions (e.g., dependency tree and intermediate query structures) that can be used to comprehend and assess the QA process of the model. However, their performance is usually limited by available resources for question parsing, such as dictionaries, corpora of queries, templates, and heuristics. Embedding-based methods learn numerical representations of questions and KGs that can be used for direct answer retrieval without logical query construction. They can learn the knowledge that parsing-based methods require people to manually define. However, most existing methods are limited to answering single-hop questions due to the lack of multi-step reasoning ability.

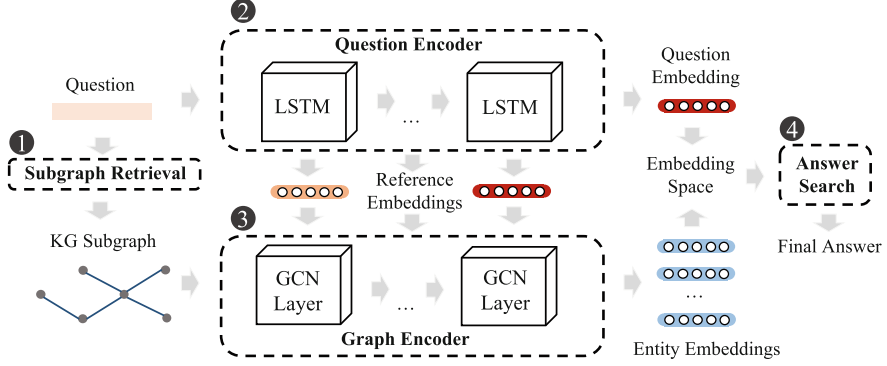


Fig. 1. An overview of the QAGCN model.

### 3 QAGCN – A Question-Aware GCN-Based Question Answering Model

In this section, we define the multi-relation QA task and elaborate on the proposed QAGCN model.

#### 3.1 Task Definition

A **knowledge graph (KG)** is denoted as  $\mathcal{G} = (E, R, T)$ , where  $E$  and  $R$  denote the entity set and relation set of  $\mathcal{G}$ , and  $T \subseteq E \times R \times E$  denotes the triple set of  $\mathcal{G}$ . A triple  $(e_h, r, e_t) \in T$  denotes that there is a relation  $r \in R$  linking the head entity  $e_h \in E$  to the tail entity  $e_t \in E$ . Entities and relations are labeled by their names in natural language. We denote the set of all labels as  $\mathcal{L}$  and define a function  $f_l : E \cup R \rightarrow \mathcal{L}$  that returns the label of a given entity or relation.

The multi-relation QA task is to answer unseen questions from a test set  $Q$  over a given KG  $\mathcal{G} = (E, R, T)$ , with a disjoint training set  $\hat{Q}$  provided. Each question  $q \in Q \cup \hat{Q}$  is expressed in natural language and can be answered by entities in  $E$ . Also, in line with existing work [9, 21, 25, 32], a topic entity is assumed to be annotated.<sup>1</sup> Finally, to simplify the following discussion, we define the reasoning chain of  $q$  as a path  $P \subseteq T$  that links the topic entity to the final answers using relations specified in  $q$ . Note that our method does not require annotations of reasoning chains.

#### 3.2 The Question-Aware GCN

We present an overview of the proposed question-aware GCN in Fig. 1. Given a multi-relation question, we feed it together with a subgraph extracted for the question from the underlying KG as input. The model consists of a question

<sup>1</sup> If topic entities are not annotated, they can still be easily obtained via named entity recognition, which has been widely studied for decades.

encoder and a graph encoder that compute semantic representations (embeddings) of the question and subgraph entities, respectively, through several layers of encoding. We select answers from subgraph entities according to their distances to the question in the output embedding space and further improve the precision of final answers based on relation labels. The entire process can be segmented into four components: subgraph extraction, question encoding, subgraph encoding, and answer search. We present detailed elaboration on each of these components in the following.

**Subgraph Extraction.** In this component, we aim to extract a more tractable subgraph, which encompasses the reasoning chain of the given question, from the potentially very large underlying KG. Specifically, given the question  $q \in Q \cup \hat{Q}$  posed over the KG  $\mathcal{G} = (E, R, T)$ , we aim to extract a subgraph  $\mathcal{G}_q = (E_q, R_q, T_q)$  that covers the reasoning chain  $P$ , i.e.,  $P \subseteq T_q \subseteq T$ . If  $q$  is a  $x$ -relation question (i.e.,  $x$  denotes the number of hops of the question),  $P$  would consist of  $x$  triples linking the topic entity  $e_q$  to the answers. Therefore, we first train a question classifier to predict  $x$  for  $q$ . The classifier consists of three linear layers with two rectified linear unit (ReLU) layers in between and a softmax output layer. For the input of the classifier, we filter out stop words and annotated topic entities from questions. Then, the rest words are represented as bag-of-words vectors with two additional entries denoting the number of words and the number of out-of-vocabulary (OOV) words. Questions in the training set  $\hat{Q}$  are used for vocabulary construction and model training. In addition, we remove 15 least used words from the vocabulary to provide training questions containing OOV words. Based on the estimated  $x$  for  $q$ , in this step, we add all paths of length  $x$  or shorter in  $\mathcal{G}$  that start from the topic entity  $e_q$  to  $\mathcal{G}_q$ . Please note that we ignore triple directions to ensure that our method is applicable to different KG schemas. For example, the information that Tim Burton directed Sleepy Hollow can be represented as (Sleepy Hollow, director, Tim Burton) or (Tim Burton, directed, Sleepy Hollow), depending on the schema.

**Question Encoding.** In this component, we filter out stop words from the question  $q$  and segment  $q$  into a sequence of  $n$  chunks  $[c_1, c_2, \dots, c_n]$ . The segmentation is performed by splitting on whitespace, with the exception that the mention of the topic entity  $e_q$  remains complete in one chunk. Then, a pre-trained BERT model [6] (bert-base-uncased<sup>2</sup>) is used to encode each chunk into an embedding vector. As suggested by bert-as-service<sup>3</sup>, the average pooling of hidden embeddings of the second-to-last BERT layer is computed as the embedding of each chunk. We denote the embeddings of  $[c_1, c_2, \dots, c_n]$  as  $[\mathbf{c}_1, \mathbf{c}_2, \dots, \mathbf{c}_n]$ , where  $\mathbf{c}_i \in \mathbb{R}^{d_0}$ ,  $i = 1, \dots, n$ , and  $d_0$  is the initial embedding size. The BERT model is only used for the initial encoding of texts. It is not fine-tuned in training and could be replaced by other methods that provide contextual text embeddings.

<sup>2</sup> <https://huggingface.co/bert-base-uncased>.

<sup>3</sup> <https://bert-as-service.readthedocs.io/>.

These embeddings are subsequently fed to  $L$  stacked LSTM [10] layers, as depicted in Fig. 1. In each layer, there are  $n$  LSTM units that sequentially update a cell state passing through them given a sequence of embeddings from the previous layer (or the initial embeddings  $[\mathbf{c}_1, \mathbf{c}_2, \dots, \mathbf{c}_n]$  in the first layer). The computed cell state of the last LSTM layer (denoted as  $\mathbf{q}_L \in \mathbb{R}^{d_L}$ ) is used as the final embedding of  $q$ , while the cell state of the  $l$ -th layer, i.e.,  $\mathbf{q}_l \in \mathbb{R}^{d_l}$ ,  $l = 1, \dots, L$ , is used as a reference embedding for the computation of attention weights in the  $l$ -th GCN layer of the graph encoder, which we introduce below.

**Subgraph Encoding.** We analogously use a pre-trained BERT model to initialize embeddings of entities and relations in  $\mathcal{G}_q$  based on their labels. Specifically, each entity  $e \in E_q$  and each relation  $r \in R_q$  are encoded into  $\mathbf{e}_0, \mathbf{r}_0 \in \mathbb{R}^{d_0}$  based on their labels  $f_l(e)$  and  $f_l(r)$ , respectively. Following other GCN work [15, 28], we add inverse edges and self-loops to  $\mathcal{G}_q$ , i.e.,  $T_q \leftarrow T_q \cup \{(e_t, r^{-1}, e_h) | (e_h, r, e_t) \in T_q\} \cup \{(e, r^s, e) | e \in E_q\}$  and  $R_q \leftarrow R_q \cup \{r^{-1} | r \in R_q\} \cup \{r^s\}$ , where  $r^{-1}$  denotes the inverse of  $r$ , and  $r^s$  denotes the self-loop relation. The inverse relation  $r^{-1}$  allows information to be passed from the tail entity  $e_t$  to the head entity  $e_h$  during graph encoding, and its initial embedding  $\mathbf{r}_0^{-1} = -1 \cdot \mathbf{r}_0$ . The self-loop relation  $r^s$  allows entities to receive information from themselves. Also, its initial embedding is set to a zero vector, i.e.,  $\mathbf{r}_0^s = \mathbf{0}$ .

The subgraph  $\mathcal{G}_q$  with initialized entity and relation embeddings are fed to  $L$  stacked GCN layers. And the graph encoding in each layer includes two steps: *message passing* and *message aggregation*. In the *message passing step*, we employ a linear transformation to compute messages that each entity receives from its context. Specifically, for an entity  $e$  in the subgraph  $\mathcal{G}_q$ , its context is defined as a set of all incoming triples  $C(e) = \{(\hat{e}, r, e) | (\hat{e}, r, e) \in T_q\}$ . In the  $l$ -th GCN layer, the message that  $e$  receives from an incoming triple  $(\hat{e}, r, e) \in C(e)$  is computed as follows:

$$m_l(\hat{e}, r, e) = \mathbf{W}_l \cdot [\hat{\mathbf{e}}_{l-1} \parallel \mathbf{r}_{l-1}] + \mathbf{b}_l, \quad (1)$$

where  $\mathbf{W}_l \in \mathbb{R}^{d_l \times 2d_{l-1}}$  and  $\mathbf{b}_l \in \mathbb{R}^{d_l}$  denote the weight and bias for the message computation,  $\hat{\mathbf{e}}_{l-1}, \mathbf{r}_{l-1} \in \mathbb{R}^{d_{l-1}}$  are embeddings of  $\hat{e}$  and  $r$  computed by the previous GCN layer (or initial embeddings if  $l = 1$ ), and  $[\hat{\mathbf{e}}_{l-1} \parallel \mathbf{r}_{l-1}]$  denotes the concatenation of  $\hat{\mathbf{e}}_{l-1}$  and  $\mathbf{r}_{l-1}$ .

Depending on the size of  $C(e)$ , the entity  $e$  may receive a large number of messages in each GCN layer. However, we expect the model to only consider messages related to the given question, i.e., question-aware message propagation. For example, given “*who is the director of Sleepy Hollow*,” only the message from *Sleepy Hollow* should be considered when encoding *Tim Burton*. To this end, in the *message aggregation step*, we propose an attention mechanism to compute weights for passed messages with reference to the given question. Specifically, in the  $l$ -th GCN layer, the weight of the message that  $e$  receives from  $(\hat{e}, r, e) \in C(e)$  is computed as follows:

$$w_l(\hat{e}, r, e) = \frac{\exp(\tanh(\mathbf{A}_l \cdot [m_l(\hat{e}, r, e) \parallel \mathbf{q}_l]))}{\sum_{(\hat{e}', r', e) \in C(e)} \exp(\tanh(\mathbf{A}_l \cdot [m_l(\hat{e}', r', e) \parallel \mathbf{q}_l]))}, \quad (2)$$

where  $\mathbf{q}_l \in \mathbb{R}^{d_l}$  is the reference embedding computed by the  $l$ -th LSTM layer of the question encoder,  $m_l(\cdot)$  computes the message passed to  $e$  from an incoming triple according to Eq. (1), and  $\mathbf{A}_l \in \mathbb{R}^{1 \times 2d_l}$  is the weight to learn for the attention mechanism in the  $l$ -th GCN layer. The goal of Eq. (2) is to compute higher weights for messages that are related to the given question while lower weights for irrelevant messages.

Based on computed weights, the embedding of  $e$  in the  $l$ -th GCN layer is updated as follows:

$$\mathbf{e}_l = \sum_{(\hat{e}, r, e) \in C(e)} w_l(\hat{e}, r, e) \cdot m_l(\hat{e}, r, e), \quad (3)$$

where  $w_l(\hat{e}, r, e) \in \mathbb{R}$  and  $m_l(\hat{e}, r, e) \in \mathbb{R}^{d_l}$ .

**Answer Search.** After  $L$  GCN layers, each entity  $e \in E_q$  is encoded as an embedding  $\mathbf{e}_L \in \mathbb{R}^{d_L}$ . The likelihood of  $e$  being the answer of the question  $q$  is measured by the Euclidean distance between  $\mathbf{e}_L$  and the question embedding  $\mathbf{q}_L$ . We rank all entities in the subgraph  $\mathcal{G}_q$  according to computed Euclidean distances to retrieve top-ranked candidate answers, which are denoted as a set  $E_c$ . Then, we leverage label information of relations in the KG to filter out outliers in  $E_c$  and further improve the precision of our results via the reranking of top-ranked candidate answers. Specifically, for each entity  $e_c \in E_c$ , we employ NetworkX and Graph Tool<sup>4</sup> to extract relation paths of the predicted length  $x$  that link  $e_c$  to the topic entity  $e_q$  in the KG subgraph. The semantics of the relation paths can be assessed based on the labels of constituent relations. For example, regarding a question estimated to be 2-hop “*where was the director of Sleepy Hollow born,*” we can extract a 2-hop relation path [birthplace, director] for the candidate answer Burbank given triples  $\{(\text{Burbank}, \text{birthplace}, \text{Tim Burton}), (\text{Tim Burton}, \text{director}, \text{Sleepy Hollow})\}$ <sup>5</sup> in the subgraph. Then, the semantics of this path can be examined based on “*birthplace*” and “*director*.”

Specifically, we denote the set of extracted relation paths for each candidate answer  $e_c \in E_c$  as  $\mathcal{P}_r(e_c)$ . The plausibility of  $e_c$  is evaluated by measuring the semantic similarities between its relation paths and the given question  $q$ . We denote a relation path  $P_r \in \mathcal{P}_r(e_c)$  as a sequence of relations, i.e.,  $P_r = [r_1, \dots, r_x]$ , where  $r_i \in R_q, i = 1, \dots, x$ . Analogous to the initial encoding of question chunks, relations in  $P_r$  are also encoded by the pre-trained BERT model, i.e.,  $P_r$  is encoded as  $[\mathbf{r}_1, \dots, \mathbf{r}_x]$ , where  $\mathbf{r}_i \in \mathbb{R}^{d_o}, i = 1, \dots, x$ . Then,  $[\mathbf{r}_1, \dots, \mathbf{r}_x]$  is fed to a single-layer LSTM, and the cell state of the LSTM is used as the embedding of  $P_r$ , denoted as  $\mathbf{P}_r \in \mathbb{R}^{d_p}$ , where  $d_p$  is the embedding size. For the given question  $q$ , we reuse the segmented chunks  $[c_1, \dots, c_n]$  and their initial embeddings computed in the question encoder, i.e.,  $[\mathbf{c}_1, \dots, \mathbf{c}_n]$ , where  $\mathbf{c}_i \in \mathbb{R}^{d_o}, i = 1, \dots, n$ . The chunk embeddings are fed to another single-layer LSTM, and its cell state is used as the embedding of  $q$ , i.e.,  $\mathbf{q}_p \in \mathbb{R}^{d_p}$ . The Euclidean distance between  $\mathbf{P}_r$  and

<sup>4</sup> <https://networkx.org/> and <https://graph-tool.skewed.de>.

<sup>5</sup> We ignore relation directions in this process.

**Table 1.** Statistics of adopted benchmark datasets.

Datasets	Underlying KG			Question Sets		
	#Entity	#Relation	#Triple	#Train	#Valid	#Test
PQ-2hop	1,056	13	1,211	1,526	190	192
PQ-3hop	1,836	13	2,839	4,158	519	521
PQL-2hop	5,034	363	4,247	1,276	159	159
PQL-3hop	6,505	411	5,597	825	103	103
MetaQA 1-hop	43,234	9	134,741	96,106	9,992	9,947
MetaQA 2-hop	43,234	9	134,741	118,980	14,872	14,872
MetaQA 3-hop	43,234	9	134,741	114,196	14,274	14,274

$\mathbf{q_p}$  is computed to measure the semantic similarity between  $P_r$  and  $q$ . We train the above two LSTMs based on training questions in  $\hat{Q}$ . Specifically, for each question  $\hat{q} \in \hat{Q}$ , relation paths between its final answer and topic entity are used as positive samples. Using other existing relations in the KG, we randomly generate paths different from the positive ones as negative samples. The distances between positive samples and  $\hat{q}$  are trained to be close to zero, while those for negative samples are trained to be close to one. For each candidate answer  $e_c$ , we take the minimum computed distance of all its relation paths in  $\mathcal{P}_r(e_c)$  as the final distance of  $e_c$ . Finally, the candidate answer in  $E_c$  with the minimum final distance is selected as the final answer.

## 4 Experiments

In this section, we evaluate the effectiveness and efficiency of our model on widely used benchmark datasets, scrutinize the contribution of each component of the model in an ablation study, and present a case study for a better understanding of our model.

### 4.1 Effectiveness Evaluation

**Baselines.** We first evaluate the overall effectiveness of QAGCN in the multi-relation QA task. Given that the main goal of this paper is to propose a simple method that is competitive with existing reasoning-based methods that rely on complex reasoning mechanism, we mainly choose reasoning-based QA methods as baselines: MINERVA [5], IRN [32], SRN [21], TransferNet [24], and NSM [9]. Among these, MINERVA, IRN, and SRN are reinforcement learning-based. TransferNet and NSM are label propagation-based. Also, TransferNet and NSM are state-of-the-art (SOTA) reasoning-based methods in multi-relation QA. Furthermore, considering that our answer search in the learned embedding space is similar to embedding-based methods, we also select a prominently adopted embedding-based method: EmbedKGQA [23].



**Datasets.** In line with the baselines, we use three collections of datasets that are particularly constructed for the multi-relation QA task: PathQuestion (PQ), PathQuestion-Large (PQL) [32] and MetaQA [31]. They consist of QA sets that are named according to the complexity (number of required reasoning hops) of respectively included questions: PQ-2hop, PQ-3hop, PQL-2hop, PQL-3hop, MetaQA 1-hop, MetaQA 2-hop, and MetaQA 3-hop. PQ and PQL are both open-domain QA datasets based on Freebase [2], while PQL is a more challenging version of PQ with less training data and larger KGs. One example question in PQ is: *what is the place of birth of Marguerite Louise Dorleans’s other half’s kid?*<sup>6</sup> MetaQA is constructed based on WikiMovies [19], mainly including questions in the movie domain. One example is: *what genres do the films that share directors with Scarlet Street fall under?* Table 1 reports statistics of the QA sets, including the number of included training/validation/test questions and the number of entities/relations/triples in the underlying KG for each question set.

**Experimental Details.** The experiments on MetaQA and PQ were conducted on a Linux server with two Intel Xeon Gold 6230 CPUs and one NVIDIA GeForce RTX 2080 Ti GPU being used. The experiments on PQL were conducted on another Linux server with two AMD EPYC 9124 16-Core CPUs and one NVIDIA GeForce RTX 4090 being used. We set hyper-parameters based on grid-search: Learning rates are set to 0.0002, 0.0005, 0.001, 0.0005, 0.0005, 0.0005, and 0.001 for MetaQA 1,2,3-hop, PQ 2,3-hop, and PQL 2,3-hop, respectively. Dropouts of 0.1 are applied in each of the GCN layers, excluding the output layer.

**Question Classification.** The classifier for predicting the complexity of a given question (cf. Subgraph Extraction in Sect. 3.2) was trained and evaluated on MetaQA, PQ, and PQL respectively using the mix of their QA sets (e.g., the mix of 2-hop and 3-hop sets of PQ). The ground-truth complexity of a question (i.e., 1, 2, or 3 hops) is determined by the QA set from which the question originates. We used the negative log-likelihood loss and the Adam optimizer [14] to train the classifier with a learning rate of  $5 \times 10^{-4}$  on each dataset. This classifier achieved 100% accuracy on the test sets of all three datasets.

**Overall Results.** Table 2 reports the overall QA performance of our model and other baselines on PQ and PQL, while Table 3 reports the results on MetaQA. We follow the baselines to measure the performance by Hits@1, which is the percentage of test questions that have been correctly answered by the returned *top-1* answers. In the above tables, the results of all baselines on MetaQA and the results of MINERVA, IRN, and SRN on PQ and PQL are from existing publications [9, 21]. The results of EmbedKGQA, TransferNet, and NSM on PQ and PQL are computed in our experiments with the source code and training configurations released by the original authors. The following can be observed:

<sup>6</sup> In the original form of this question, all letters are lowercase, the entity phrase is connected by underlines (e.g., marguerite\_louise\_dorleans). We slightly change the format for better readability.

**Table 2.** Effectiveness performance on PathQuestion (PQ) and PathQuestion-Large (PQL). (% Hits@1, best performance in **bold**, second best underlined)

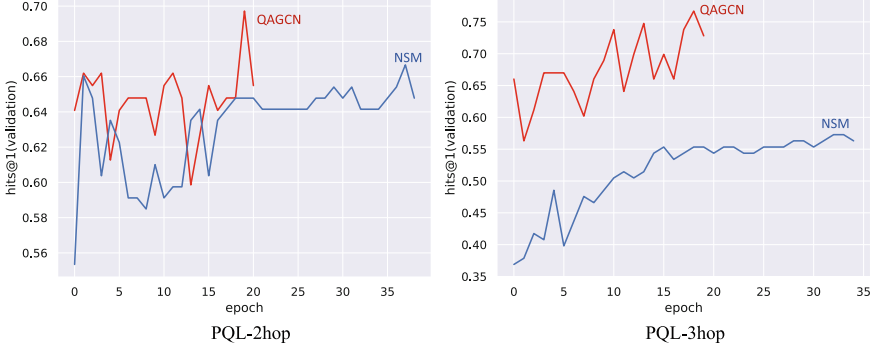
	PQ-2hop	PQ-3hop	PQL-2hop	PQL-3hop
MINERVA [5]	75.9	71.2	71.8	65.7
IRN [32]	91.9	83.3	63.0	61.8
EmbedKGQA [23]	90.1	86.2	<u>79.2</u>	61.2
SRN [21]	<u>96.3</u>	89.2	78.6	<b>77.5</b>
TransferNet [24]	91.1	<u>96.5</u>	54.7	62.1
NSM [9]	94.2	<b>97.1</b>	74.2	67.0
QAGCN	<b>98.5</b> *	90.6*	<b>87.5</b>	<u>70.9</u>

Results marked with \* are the average results of five runs using different random training/validation/test splits.

**Table 3.** Effectiveness performance on MetaQA. (% Hits@1)

	MetaQA 1-hop	MetaQA 2-hop	MetaQA 3-hop
MINERVA [5]	96.3	92.9	55.2
IRN [32]	85.9	71.3	35.6
EmbedKGQA [23]	<b>97.5</b>	98.8	94.8
SRN [21]	97.0	95.1	75.2
TransferNet [24]	<b>97.5</b>	<b>100</b>	<b>100</b>
NSM [9]	<u>97.3</u>	<u>99.9</u>	<u>98.9</u>
QAGCN	<u>97.3</u>	<u>99.9</u>	97.6

- QAGCN achieved superior performance than existing SOTA methods on PQ-2hop and PQL-2hop with relative improvements of 2.3% and 10.5%, respectively. Also, on MetaQA 1-hop and MetaQA 2-hop, QAGCN achieved the second-best performance, which is very close to the SOTA methods with only relative drops of  $-0.2\%$  and  $-0.1\%$ . This demonstrates the competitiveness of QAGCN, considering its simplicity in comparison with the SOTA methods.
- On PQL-3hop, QAGCN is ranked second with a drop of  $-8.5\%$  in comparison with SRN. This demonstrates the strength of SRN’s reinforcement learning(RL)-based policy when answering complex questions. However, please note that QAGCN also has a large margin of 5.8% in comparison with the third-best method NSM. This demonstrates that, on complex questions, the simple single-step reasoning of QAGCN could perform better than the SOTA methods with complex multi-step label propagation.
- On PQ-3hop and MetaQA-3hop, QAGCN are both ranked third. This reflects that it is indeed challenging for the simple reasoning of QAGCN to answer complex 3-hop questions. However, the performance of QAGCN is better than most reasoning-based methods, e.g., 29.8% and 1.6% higher than the best-performing RL-based method SRN on MetaQA 3-hop and PQ-3hop, respectively.



**Fig. 2.** Training curves of QAGCN and NSM on PQL-2hop and PQL-3hop.

- In general, QAGCN exhibits less promising performance on 3-hop questions compared to 2-hop questions. We could see three possible explanations: First, for 3-hop questions, one more layer of GCN/LSTM would be needed in the graph/question encoders. Hence, it is more challenging to train the model with limited data. Second, the message passing in the GCNs and cell state passing in the LSTMs become longer for 3-hop questions, which are more prone to errors. Third, the search space substantially grows with 3-hop questions. As shown in Table 1, the KGs of 3-hop PQ/PQL questions are substantially larger than those of 2-hop PQ/PQL questions.

## 4.2 Comparison with SOTA Reasoning-Based Method — NSM

In this section, we compare with the SOTA reasoning-based method NSM to demonstrate that QAGCN is simpler and easier to train.

The training curves of QAGCN and NSM on PQL-2hop and PQL-3hop are shown in Fig. 2. For clarity, we truncate the curves from the second epoch after the optimal epoch of each training. It can be observed that QAGCN only requires about half of the epochs needed by NSM, specifically 51.4% and 54.5% on PQL-2hop and PQL-3hop, respectively. This demonstrates that, while maintaining the above-demonstrated effectiveness, QAGCN is easier to train than NSM. The reason is two-fold: First, NSM relies on a teacher network to provide intermediate supervision for the multi-step reasoning of its student network. Within the teacher network, two additional multi-step reasoning processes over KGs are performed. Therefore, for each training question, NSM needs to reason three times over KGs (once for the student network and twice for the teacher network), which is inefficient given that our model only needs to reason once. Second, the intermediate supervision of the teacher network is represented as probability distributions of entities. The Kullback-Leibler divergence and Jensen-Shannon divergence [7] need to be additionally computed for several pairs of distributions in the loss function of NSM, which makes the model difficult to implement and optimize.

**Table 4.** Average run-time of each step per test question in ms.

	MetaQA			PathQuestion (PQ)	
	1-hop	2-hop	3-hop	2-hop	3-hop
Question Processing	1.4	1.6	1.1	0.4	0.7
Subgraph Extraction	3.9	6.6	357.7	1.2	18.3
Question Encoding	16.3	16.3	10.0	8.4	9.9
Subgraph Encoding	1.1	2.9	26.5	1.3	2.4
Answer Search	0.6	5.2	287.1	0.4	2.3
Answer Re-ranking	79.5	123.2	1012.1	8.0	15.4
Total	102.8	155.8	1694.5	19.7	49.0

### 4.3 Efficiency Evaluation

Question answering is supposed to be an online service that answers questions in real-time. We evaluated the average time cost of each step of our model on MetaQA and PathQuestion. The results are reported in Table 4. In this table, Question Processing includes the segmentation and classification of questions. The others are in accordance with the steps introduced in Sect. 3. Please note that we list the path extraction-based answer refinement separately as Answer Re-ranking, given that the explicit path extraction in this step could be particularly time-consuming and is worth specific attention. Most time costs are below or near 100ms, which can be seen as instantaneous in user interfaces [20]. Only 3-hop questions in MetaQA require an average of 1.7s, which we consider acceptable when contrasted with the complexity of 3-hop questions. It is worth mentioning that our implementation can be further optimized for a real-world deployment. For example, the subgraph extraction can be performed offline given a KG, and the extraction and encoding of relation paths for candidate answers can be performed in a completely parallel way.

### 4.4 Ablation Study

To examine the contribution of each component of our model, we evaluate the model in a series of degraded scenarios and report the performance changes in Table 5. **Complete Model** reports the results of the complete QAGCN model. **Re-ranking Removed** shows the performance when candidate answers are not re-ranked in Answer Search. The importance of the re-ranking is demonstrated by the performance degradation (e.g.,  $-43.3\%$  on MetaQA 3-hop and  $-44.2\%$  on PathQuestion 3-hop). In **Q&G-Encoders Removed**, the question and graph encoders are removed, and we use the pre-trained BERT model to directly encode questions and entities. Top- $k$  candidate answers are retrieved and then re-ranked by the original re-ranking module.<sup>7</sup> In this scenario, there is

<sup>7</sup> The values of  $k$  were set to be consistent with those used in the complete model.

**Table 5.** Model performance changes in the ablation study. (% Hits@1).

	MetaQA			PathQuestion (PQ)	
	1-hop	2-hop	3-hop	2-hop	3-hop
Complete Model	97.3	99.9	97.6	98.4*	92.1*
Re-ranking Removed	95.8	87.4	55.3	72.9	51.4
Q&G-Encoders Removed	97.0	91.3	86.9	74.5	43.2
G-Encoder (Conv)	64.6	63.9	40.8	67.7	28.0
G-Encoder (Linear)	72.0	55.6	15.2	71.4	43.0
Q&G-Encoders (Linear)	70.7	52.4	0.1	63.5	16.9

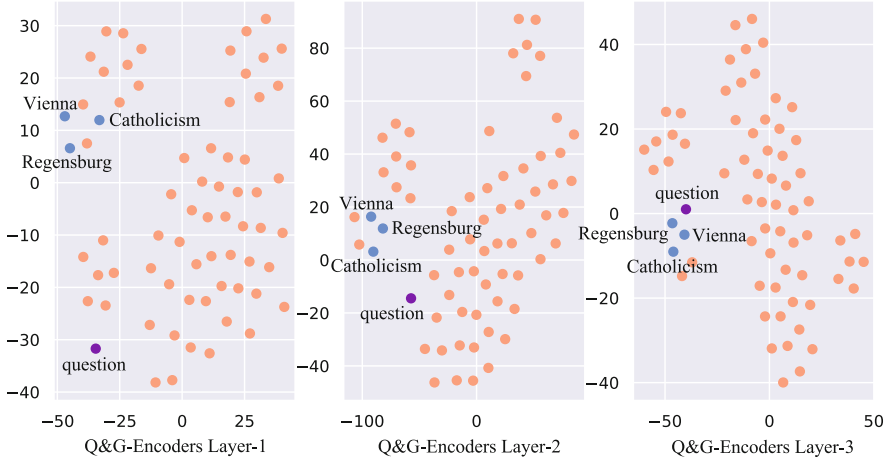
\* We report the results of QAGCN on the first run of PQ-2/3hop. Hence, these results differ from the averages reported in Table 2.

also a significant performance drop (e.g.,  $-11.0\%$  on MetaQA 3-hop and  $-53.1\%$  on PathQuestion 3-hop). From the above three scenarios, we observe that both the re-ranking module and question/graph encoders are necessary, and that they are complementary to each other.

In **G-Encoder (Conv)**, we replace the proposed attentional GCNs with conventional GCNs [15]. The question encoder remains unchanged, and we report direct results without re-ranking. Compared to Re-ranking Removed, i.e., the original encoders, performance degradations can be observed on all datasets (e.g.,  $-26.2\%$  on MetaQA 3-hop and  $-45.5\%$  on PathQuestion 3-hop). In **G-Encoder (Linear)**, we further replace the conventional GCNs with feedforward neural networks. In this scenario, entities are encoded without considering their contexts in KGs. Compared with G-Encoder (Conv), performance drops can be observed on questions with large subgraphs (e.g.,  $-13.0\%$  on MetaQA 2-hop and  $-62.7\%$  on MetaQA 3-hop). While for questions with small subgraphs, in which entities do not have sufficient contexts, improvements are observed (e.g.,  $+11.5\%$  on MetaQA 1-hop and  $+5.5\%$  on PathQuestion 2-hop). In **Q&G-Encoders (Linear)**, based on G-Encoder (Linear), we further replace the LSTMs in the question encoder with feedforward neural networks. In this case, the model mainly relies on the initial encoding of questions and entities computed by the BERT model. Since the BERT model was pre-trained on other tasks, the performance on these QA tasks is expected to be low. In Table 5, we can observe substantial performance drops on all datasets compared to the last scenario (e.g.,  $-99.3\%$  on MetaQA 3-hop and  $-60.7\%$  on PathQuestion 3-hop).

#### 4.5 Case Study with Embedding Visualization

For an intuitive understanding of the proposed QAGCN model, we select an example question to conduct a case study with the visualization of question and entity embeddings. Specifically, we picked the following 3-hop test question from PathQuestion: *Isabella of Portugal's child's spouse's place of death?* The topic entity is *Isabella of Portugal* with a support path of three triples:



**Fig. 3.** Visualization of the embedding space after each layer of the Q&G-Encoders. Please note that the axes do not have a same scale. The ■ purple point indicates the question embedding after each layer. The ■ blue points indicate the top-3 candidates finally retrieved. The ■ orange points indicate all other entities in the subgraph.

$\{$ (Isabella of Portugal, children, Maria of Spain),  
 (Maria of Spain, spouse, Maximilian II Holy Roman Emperor),  
 (Maximilian II Holy Roman Emperor, place of death, Regensburg) $\}$ .

For this question, three GCN layers are used in the graph encoder to encode entities in the subgraph that covers all paths of length one, two, and three starting from the topic entity *Isabella of Portugal*. The question is accordingly encoded by the question encoder with three layers of LSTMs. Then, according to Euclidean distances between the question embedding and entity embeddings, we obtain the top-3 candidate answers: *Vienna* (distance: 1.53), *Regensburg* (distance: 1.89), and *Catholicism* (distance: 3.17). The correct answer *Regensburg* is ranked after *Vienna*. Since *Vienna* is the place of birth of *Maximilian II Holy Roman Emperor*, it is likely that this trained model struggles to differentiate between “birth” and “death.” A second possible reason is that this trained model is biased towards *Vienna* given that *Vienna* is the answer of 12 training questions that ask for the place of death of people, while *Regensburg* is the answer of only one training question.

To visualize the joint embedding space of the LSTM-based question encoder and GCN-based graph encoder for each of the three layers, we use t-SNE [18] to compute dimension-reduced entity and question embeddings and plot them in Fig. 3. We highlight the final three top candidate answers in blue and plot all other entities in the subgraph as orange points. In addition, we plot the location of the question embedding after each of the LSTM layers with a purple point. It can be observed that initially (i.e., in Layer-1) the correct answer is not at all close to the question in the embedding space. However, after more layers of encoding, more information consistent with the given question is passed to

**Regensburg** by the graph encoder, and the question embedding is accordingly transformed by the question encoder. Therefore, we can observe that **Regensburg** and the question approach each other gradually in the embedding space.

Then, we extract relation paths between the top-3 candidate answers and the topic entity, which are respectively  $\{[\text{place of birth, spouse, children}]\}$ ,  $\{[\text{place of death, spouse, children}]\}$ , and  $\{[\text{religion, spouse, children}], [\text{religion, parents, children}]\}$  for **Vienna**, **Regensburg**, and **Catholicism**. The minimum path-question distances of **Vienna**, **Regensburg**, and **Catholicism** computed by the re-ranking module are respectively 0.99, 0.69, and 1.00. Finally, with the minimum distance 0.69, **Regensburg** is correctly returned as the final answer.

## 5 Conclusion and Outlook

In this paper, we propose a novel QA model called QAGCN, which answers multi-relation questions via single-step implicit reasoning over KGs. A novel question-aware GCN is proposed to perform question-dependent message propagation while encoding KGs. It is able to represent questions and KG entities in a joint embedding space, where questions are close to their answers. Our model has been demonstrated to be competitive and even superior against SOTA reasoning-based methods in experiments, while our model is simpler and easier to train and implement. The efficiency and contribution of each component of our model have also been examined and analyzed in detail.

The task we solve in this paper is in line with other existing QA work that assumes all given questions are answerable. Therefore, if the answer to a question does not exist in the accompanied KG, our model will still confidently report an answer, which is actually incorrect. In future work, we plan to investigate a method that can detect if a given question is answerable regarding a given KG.

**Acknowledgement.** This work has been partially supported by the University Research Priority Program “Dynamics of Healthy Aging” at the University of Zurich and the Swiss National Science Foundation through project MediaGraph (contract no. 202125). Michael Cochez is partially funded by the Graph-Massivizer project, funded by the Horizon Europe programme of the European Union (grant 101093202).

## References

1. Berant, J., Liang, P.: Semantic parsing via paraphrasing. In: Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, ACL 2014, June 22-27, 2014, Baltimore, MD, USA, Volume 1: Long Papers, pp. 1415–1425. The Association for Computer Linguistics (2014). <https://doi.org/10.3115/v1/p14-1133>
2. Bollacker, K.D., Evans, C., Paritosh, P.K., Sturge, T., Taylor, J.: Freebase: a collaboratively created graph database for structuring human knowledge. In: Wang, J.T. (ed.) Proceedings of the ACM SIGMOD International Conference on Management of Data, SIGMOD 2008, Vancouver, BC, Canada, June 10-12, 2008, pp. 1247–1250. ACM (2008). <https://doi.org/10.1145/1376616.1376746>

3. Bordes, A., Chopra, S., Weston, J.: Question answering with subgraph embeddings. In: Moschitti, A., Pang, B., Daelemans, W. (eds.) *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014*, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL, pp. 615–620. ACL (2014). <https://doi.org/10.3115/v1/d14-1067>
4. Bordes, A., Usunier, N., Chopra, S., Weston, J.: Large-scale simple question answering with memory networks. *CoRR* **abs/1506.02075** (2015). <http://arxiv.org/abs/1506.02075>
5. Das, R., et al.: Go for a walk and arrive at the answer: reasoning over paths in knowledge bases using reinforcement learning. In: *6th International Conference on Learning Representations, ICLR 2018, Vancouver, BC, Canada, April 30 - May 3, 2018, Conference Track Proceedings*. OpenReview.net (2018). <https://openreview.net/forum?id=Syg-YfWCW>
6. Devlin, J., Chang, M., Lee, K., Toutanova, K.: BERT: pre-training of deep bidirectional transformers for language understanding. In: Burstein, J., Doran, C., Solorio, T. (eds.) *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pp. 4171–4186. Association for Computational Linguistics (2019). <https://doi.org/10.18653/v1/n19-1423>
7. Fuglede, B., Topsøe, F.: Jensen-shannon divergence and Hilbert space embedding. In: *Proceedings of the 2004 IEEE International Symposium on Information Theory, ISIT 2004, Chicago Downtown Marriott, Chicago, Illinois, USA, June 27 - July 2, 2004*, p. 31. IEEE (2004). <https://doi.org/10.1109/ISIT.2004.1365067>
8. Harris, S., Seaborne, A., Prud'hommeaux, E.: SPARQL 1.1 query language. w3c recommendation (2013). <https://www.w3.org/TR/sparql11-query/>
9. He, G., Lan, Y., Jiang, J., Zhao, W.X., Wen, J.: Improving multi-hop knowledge base question answering by learning intermediate supervision signals. In: Lewin-Eytan, L., Carmel, D., Yom-Tov, E., Agichtein, E., Gabrilovich, E. (eds.) *WSDM '21, The Fourteenth ACM International Conference on Web Search and Data Mining*, Virtual Event, Israel, March 8-12, 2021, pp. 553–561. ACM (2021). <https://doi.org/10.1145/3437963.3441753>
10. Hochreiter, S., Schmidhuber, J.: Long short-term memory. *Neural Comput.* **9**(8), 1735–1780 (1997)
11. Hu, S., Zou, L., Yu, J.X., Wang, H., Zhao, D.: Answering natural language questions by subgraph matching over knowledge graphs. *IEEE Trans. Knowl. Data Eng.* **30**(5), 824–837 (2018). <https://doi.org/10.1109/TKDE.2017.2766634>
12. Huang, X., Zhang, J., Li, D., Li, P.: Knowledge graph embedding based question answering. In: Culpepper, J.S., Moffat, A., Bennett, P.N., Lerman, K. (eds.) *Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining, WSDM 2019, Melbourne, VIC, Australia, February 11-15, 2019*, pp. 105–113. ACM (2019). <https://doi.org/10.1145/3289600.3290956>
13. Kaufmann, E., Bernstein, A.: How useful are natural language interfaces to the semantic web for casual end-users? In: Aberer, K., et al. (eds.) *The Semantic Web*, pp. 281–294. Springer, Berlin Heidelberg, Berlin, Heidelberg (2007). [https://doi.org/10.1007/978-3-540-76298-0\\_21](https://doi.org/10.1007/978-3-540-76298-0_21)
14. Kingma, D.P., Ba, J.: Adam: a method for stochastic optimization. In: Bengio, Y., LeCun, Y. (eds.) *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings* (2015). <http://arxiv.org/abs/1412.6980>



15. Kipf, T.N., Welling, M.: Semi-supervised classification with graph convolutional networks. In: 5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings. OpenReview.net (2017). <https://openreview.net/forum?id=SJU4ayYgl>
16. Liang, P.: Lambda dependency-based compositional semantics. CoRR **abs/1309.4408** (2013). <http://arxiv.org/abs/1309.4408>
17. Lukovnikov, D., Fischer, A., Lehmann, J., Auer, S.: Neural network-based question answering over knowledge graphs on word and character level. In: Barrett, R., Cummings, R., Agichtein, E., Gabrilovich, E. (eds.) Proceedings of the 26th International Conference on World Wide Web, WWW 2017, Perth, Australia, April 3-7, 2017, pp. 1211–1220. ACM (2017). <https://doi.org/10.1145/3038912.3052675>
18. van der Maaten, L., Hinton, G.: Visualizing data using t-SNE. J. Mach. Learn. Res. **9**(86), 2579–2605 (2008), <http://jmlr.org/papers/v9/vandermaaten08a.html>
19. Miller, A.H., Fisch, A., Dodge, J., Karimi, A., Bordes, A., Weston, J.: Key-value memory networks for directly reading documents. In: Su, J., Carreras, X., Duh, K. (eds.) Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP 2016, Austin, Texas, USA, November 1-4, 2016, pp. 1400–1409. The Association for Computational Linguistics (2016). <https://doi.org/10.18653/v1/d16-1147>
20. Nielsen, J.: Response times: the 3 important limits (1991). <https://www.nngroup.com/articles/response-times-3-important-limits/>
21. Qiu, Y., Wang, Y., Jin, X., Zhang, K.: Stepwise reasoning for multi-relation question answering over knowledge graph with weak supervision. In: Caverlee, J., Hu, X.B., Lalmas, M., Wang, W. (eds.) WSDM '20: The Thirteenth ACM International Conference on Web Search and Data Mining, Houston, TX, USA, February 3-7, 2020, pp. 474–482. ACM (2020). <https://doi.org/10.1145/3336191.3371812>
22. Ren, H., et al.: Graph convolutional networks in language and vision: a survey. Knowl. Based Syst. **251**, 109250 (2022). <https://doi.org/10.1016/J.KNOSYS.2022.109250>
23. Saxena, A., Tripathi, A., Talukdar, P.P.: 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, ACL 2020, Online, July 5-10, 2020, pp. 4498–4507. Association for Computational Linguistics (2020). <https://doi.org/10.18653/v1/2020.acl-main.412>, <https://doi.org/10.18653/v1/2020.acl-main.412>
24. Shi, J., Cao, S., Hou, L., Li, J., Zhang, H.: TransferNet: an effective and transparent framework for multi-hop question answering over relation graph. In: Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, EMNLP 2021, Virtual Event / Punta Cana, Dominican Republic, 7-11 November, 2021, pp. 4149–4158. Association for Computational Linguistics (2021). <https://doi.org/10.18653/v1/2021.emnlp-main.341>
25. Sun, H., Bedrax-Weiss, T., Cohen, W.W.: PullNet: open domain question answering with iterative retrieval on knowledge bases and text. In: Inui, K., Jiang, J., Ng, V., Wan, X. (eds.) Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019, pp. 2380–2390. Association for Computational Linguistics (2019). <https://doi.org/10.18653/v1/D19-1242>
26. Sun, H., Dhingra, B., Zaheer, M., Mazaitis, K., Salakhutdinov, R., Cohen, W.W.: Open domain question answering using early fusion of knowledge bases and text.

- In: Riloff, E., Chiang, D., Hockenmaier, J., Tsujii, J. (eds.) *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, Brussels, Belgium, October 31 - November 4, 2018, pp. 4231–4242. Association for Computational Linguistics (2018). <https://doi.org/10.18653/v1/d18-1455>
27. Thai, D., et al.: CBR-iKB: a case-based reasoning approach for question answering over incomplete knowledge bases. *CoRR* **abs/2204.08554** (2022). <https://doi.org/10.48550/ARXIV.2204.08554>
  28. Vashishth, S., Sanyal, S., Nitin, V., Talukdar, P.P.: Composition-based multi-relational graph convolutional networks. In: *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net (2020). [https://openreview.net/forum?id=BylA\\_C4tPr](https://openreview.net/forum?id=BylA_C4tPr)
  29. Wang, R., Wang, M., Liu, J., Chen, W., Cochez, M., Decker, S.: Leveraging knowledge graph embeddings for natural language question answering. In: Li, G., Yang, J., Gama, J., Natwichai, J., Tong, Y. (eds.) *Database Systems for Advanced Applications. Lecture Notes in Computer Science()*, vol. 11446, pp. 659–675. Springer, Cham (2019). [https://doi.org/10.1007/978-3-030-18576-3\\_39](https://doi.org/10.1007/978-3-030-18576-3_39)
  30. Zhang, J., et al.: Subgraph retrieval enhanced model for multi-hop knowledge base question answering. In: Muresan, S., Nakov, P., Villavicencio, A. (eds.) *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2022, Dublin, Ireland, May 22-27, 2022*, pp. 5773–5784. Association for Computational Linguistics (2022). <https://doi.org/10.18653/V1/2022.ACL-LONG.396>
  31. Zhang, Y., Dai, H., Kozareva, Z., Smola, A.J., Song, L.: Variational reasoning for question answering with knowledge graph. In: McIlraith, S.A., Weinberger, K.Q. (eds.) *Proceedings of the Thirty-Second AAAI Conference on Artificial Intelligence, (AAAI-18), the 30th innovative Applications of Artificial Intelligence (IAAI-18), and the 8th AAAI Symposium on Educational Advances in Artificial Intelligence (EAAI-18), New Orleans, Louisiana, USA, February 2-7, 2018*, pp. 6069–6076. AAAI Press (2018). <https://www.aaai.org/ocs/index.php/AAAI/AAAI18/paper/view/16983>
  32. Zhou, M., Huang, M., Zhu, X.: An interpretable reasoning network for multi-relation question answering. In: Bender, E.M., Derczynski, L., Isabelle, P. (eds.) *Proceedings of the 27th International Conference on Computational Linguistics, COLING 2018, Santa Fe, New Mexico, USA, August 20-26, 2018*, pp. 2010–2022. Association for Computational Linguistics (2018), <https://aclanthology.org/C18-1171/>