Expert Systems With Applications 255 (2024) 124652

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa

Incorporating multi-perspective information into reinforcement learning to address multi-hop knowledge graph question answering

Chuanyang Gong [a], Zhihua Wei [a,*], Rui Wang [b], Ping Zhu [a], Jing Chen [a], Hongyun Zhang [a], Duoqian Miao [a]

[a] Department of Computer Science and Technology, Tongji University, Shanghai, China
[b] iFLYTEK Research, Shanghai, PR China

ARTICLE INFO

Keywords:
Multi-perspective information fusion
Knowledge graph embedding
Multi-hop knowledge graph question answering
Reinforcement learning
Reasoning interpretability

ABSTRACT

Knowledge graph question answering (KGQA) aims to answer natural language questions from structured knowledge graphs (KGs). Traditional KQGA methods are usually limited to single-hop queries and cannot handle complex questions involving multi-hop reasoning well. To overcome this issue, multi-hop KGQA based on reinforcement learning (RL) has been proposed. However, multi-hop KGQA based on RL still faces some challenges. Firstly, due to the insufficient availability of latent environmental information during the reasoning process, the agent finds it challenging to make coherent and correct decisions. Secondly, the agent only receives rewards from the environment upon reaching the answer entity during the exploration, leading to slow or even obstructed learning. To address these shortcomings, we construct multi-perspective information based on the state of the environment, and integrate multi-perspective information with RL framework, thereby creating the Multi-Perspective Information Fusion Reasoning Network (MPIFRN). MPIFRN achieves the goal via three steps. (1) We construct three different views of information, i.e., expectation embedding, instruction-guided embedding, and path-aware embedding. These environmental cues provide more reliable support for decision-making. (2) We still adopt the method of mapping entities and relations into the knowledge graph embedding space to answer multi-hop questions. At each step of reasoning, we use a scoring function to measure the plausibility of each "triple" <*topic entity, question, candidate entity*> in the embedding space. (3) Furthermore, we employ the asynchronous advantage actor-critic (A3C) algorithm to guide the agent in selecting the most promising entities and to expand the reasoning paths in parallel by updating policy and value network parameters, thereby facilitating multi-hop knowledge graph question answering. We conduct extensive experiments on KGQA benchmark datasets, providing substantial evidence to demonstrate the effectiveness of our approach.

1. Introduction

Knowledge graph question answering (KGQA) is a natural language processing task that aims to utilize structured knowledge from knowledge graphs (KGs) to answer natural questions posed by users. Previous methods (Bast & Haussmann, 2015; Berant et al., 2013; Hao et al., 2017) often consisted of multiple intricate and specialized processes, including named entity recognition, entity linking, information retrieval, and other manually designed pipelines. With the rapid development of deep learning, neural networks have transformed it into an end-to-end task, and have achieved remarkable progress, gradually becoming a general paradigm for solving it. Recently, an increasing amount of research has focused on more complex question answering patterns (Chen et al., 2023; Christmann et al., 2022; Jin et al., 2023; Vakulenko et al., 2019), specifically those that require multiple hops of reasoning on a KG to find the answer, known as multi-hop KGQA (Qin et al., 2020; Shi et al., 2021; Wu et al., 2021). To answer multi-hop KGQA, such as "Who is the director of the movies starring Jackie Chan?", we need to retrieve multiple potentially relevant triples from KG that contain a large amount of information, e.g., ⟨*Jackie Chan, starring, New Police Story*⟩ and ⟨*New Police Story, directed by, Chen Musheng*⟩. Starting with the topic entity "Jackie Chan", we continuously expand the most promising relations, gradually forming the intermediate entity "New Police Story" and finally constructing a complete reasoning chain:⟨ *Jackie Chan* → *starring* → *New Police Story* → *directed by* → *Chen Musheng*⟩. Through

[*] Corresponding author.
E-mail addresses: gongchuanyang@tongji.edu.cn (C. Gong), zhihua_wei@tongji.edu.cn (Z. Wei), rwang@tongji.edu.cn (R. Wang), pingzhu@tongji.edu.cn (P. Zhu), chenjing_miss@tongji.edu.cn (J. Chen), zhanghongyun@tongji.edu.cn (H. Zhang), dqmiao@tongji.edu.cn (D. Miao).

https://doi.org/10.1016/j.eswa.2024.124652
Received 8 November 2023; Received in revised form 11 June 2024; Accepted 20 June 2024
Available online 10 July 2024

the association of entities, we ultimately obtain the answer entity “Chen Musheng”.

However, in large-scale KGs, as the path expands, the number of candidate relations and entities increases exponentially, resulting in a dramatically expanded search space, which places very high demands on computational resources and efficiency. At the same time, during the path expansion process, some noise or erroneous entities may be introduced, and these incorrect entities can potentially propagate further in the path expansion, ultimately leading to inaccurate answers.

Some works (Chen et al., 2019; Han et al., 2020; Lv et al., 2020; Zhu et al., 2022) were inspired by human-like step-by-step reasoning and proposed multi-stage interpretable KGQA models based on dynamic relations. Although these methods have achieved some degree of effectiveness, they still need to “label” the relations required for reasoning in advance. For datasets with varying numbers of hops, these methods are not flexible enough. Recent studies (Gardner et al., 2013; Kaiser et al., 2021; Xiong et al., 2017) on multi-hop KGQA based on reinforcement learning (RL) have been carried out. In this method, multi-hop KGQA is modeled as a sequential decision-making task, where each reasoning step corresponds to a decision. RL guides the model to explore different action sequences in the KG through reward, facilitating the discovery of potentially effective paths and answers. Additionally, RL involves a well-defined action selection process at each decision step, enhancing the interpretability of the model’s reasoning paths. Therefore, this paper continues to follow this technical route and builds a multi-hop KGQA model based on RL.

Although multi-hop KGQA based on RL has achieved promising performance, it still faces various challenges. Due to the agent cannot obtain sufficient information from the environment during the reasoning process, it is difficult to make coherent and correct decisions, and the accumulated errors lead to the failure of subsequent reasoning. Specifically, the agent based on policy learning usually only relies on limited explicit state information to make simple decisions. Even if the agent makes incorrect decisions, it cannot promptly acquire additional information from the environment to correct its behavior, which to some extent hinders the agent from effectively exploring the state space and reduces the reliability of the strategy.

In addition, training data in multi-hop KGQA typically appear in the form of question–answer pairs rather than reasoning chains, i.e., ⟨*question, reasoning paths, answer*⟩. During the exploration process, the agent can only rely on the sparse and delayed rewards received after reaching the end entity for biased reasoning, but cannot fully utilize the explicit information contained in the reasoning chain, which further increases the difficulty of the agent exploration. Inspired by reward shaping (Lin et al., 2018), some studies have proposed soft reward and action pruning to improve the model’s biased reasoning. However, most of these methods depend on complex and cumbersome expert knowledge, and this problem has not yet been effectively solved. Through in-depth analysis, we find that the model tends to generate false and erroneous reasoning paths, which is due to the lack of effective environmental information in the reasoning process.

Differing from previous studies, from the perspective of assisting agent reasoning, we integrate multi-perspective information with RL to build an interpretable multi-hop KGQA model, which we refer to as Multi-Perspective Information Fusion Reasoning Network (MPIFRN). Specifically, we construct different perspectives of information, i.e., expectation embedding, instruction-guided embedding, and path-aware embedding. As for expectation embedding, we apply the knowledge embedding-based question answering (KEQA) framework to compute the probability distribution of candidate entities in the knowledge graph embedding (KGE) space. Then, we perform a weighted average of the embedding vectors of these candidate entities according to the probability distribution to get the expectation embedding. Subsequently, expectation embedding is injected into the reasoning module of the RL framework to alleviate the agent’s biased reasoning and improve exploration efficiency and reasoning performance. Given that the question representation in multi-hop KGQA needs to change with time during each hop reasoning process, we begin with the question representation and construct a dynamic instruction-guided embedding that can facilitate agent reasoning. Inspired by bidirectional search on the graph (Xiong et al., 2017), we employ a depth-first search algorithm to find paths from topic entity to answer entity and use the KGE method to encode these paths to get path-aware embedding that is integrated into reinforcement learning to improve reasoning ability. Finally, we integrate expectation embedding, instruction-guided embedding, and path-aware embedding with the multi-hop KGQA framework based on RL, and continuously optimize the agent’s strategy through the asynchronous advantage actor-critic (A3C) algorithm to complete the multi-hop KGQA task.

In summary, the expectation embedding is a weighted average of embedding vectors for all candidate entities, while the path-aware embedding encodes the paths from the subject entity to the candidate answer entities. These embeddings are derived from knowledge graph embeddings, providing a static description of the agent’s environment and serving as prior global supervision information. For instruction-guided embedding, it focuses on specific parts of the question at different stages of multi-hop question answering, acting as a dynamic local guiding signal to track the reasoning state. This information, viewed from multiple perspectives (static to dynamic, global to local), not only accurately reflects the agent’s environment and reasoning state but also serves as a basis for policy learning, guiding the agent to make more informed decisions.

The main contributions of this paper can be summarized as follows: To enable the agent to make more reliable decisions in complex environments, we construct three different types of information from different perspectives, i.e., expectation embedding, instruction-guided embedding, and path-aware embedding. This multi-perspective information exhibits model-agnostic generality, allowing for flexible and broad applicability across various KGQA models, thereby effectively enhancing their performance in multi-hop KGQA tasks. Furthermore, we integrate multi-perspective information into a meticulously designed KEQA framework and combine it with the A3C policy learning algorithm (Mnih et al., 2016). This integration not only enhances the interpretability of the model’s reasoning paths but also mitigates biased reasoning by the agent, improving its effective exploration of critical paths.

To evaluate the effectiveness of the proposed method, we conduct extensive experiments and detailed ablation studies on three benchmark datasets for multi-hop KGQA. We progressively integrate multi-perspective information into the A3C reinforcement learning module, which in turn enhances the interpretability of the model’s reasoning to some extent. Experimental results demonstrate that not only does the multi-perspective information enable MPIFRN to outperform most KGQA models in Hits@1 score, but it also accelerates the convergence speed of RL, thereby enhancing the exploration efficiency and reasoning performance of the agent. In addition, we also study the impact of different ways of information fusion on model performance. The experimental results demonstrate the effectiveness of our method in the multi-hop KGQA task.

2. Related work

In this section, we briefly summarize the existing research on KGQA and illustrate the connection and difference between our work and existing studies. KGQA is a question answering technique based on KGs, designed to extract structured information from large-scale KGs and respond to users’ questions through querying and reasoning.

Based on the required length of the relation paths between the topic entity in the question and the answer entity, KGQA can be divided into three primary categories: single-hop QA (Cui et al., 2021; Zhou et al., 2021), multi-hop QA (Cui, Peng, Bao et al., 2023; Qiu et al., 2020), and complex QA (Shin & Lee, 2020; Yang, Lee et al., 2015; Zhang et al., 2018). In comparison to single-hop QA, multi-hop KGQA poses a greater challenge. Multi-hop QA requires the question answering system to have robust reasoning capabilities and be able to perform complex reasoning across multiple relation paths. Due to the incompleteness of the KG itself, the absence of critical triples may hinder the question answering system to accurately pinpoint the answer. Additionally, entities and relations in KG exhibit ambiguity and polysemy, which also pose a significant challenge to multi-hop KGQA. Further exacerbating the difficulty of the problem is that a large-scale KG contains a large number of relations and entities, resulting in an exponential growth of the search space for multi-hop KGQA, consuming a large amount of computing resources.

Given the complexity of multi-hop KGQA, it has gradually become a research hotspot. According to different routes, the research on multi-hop KGQA can be divided into three main branches: embedding-based multi-hop KGQA, path-based multi-hop KGQA, and logic-based multi-hop KGQA. Embedding-based multi-hop KGQA generally applies the KGE method to map entities and relations to semantic vectors in the embedding space and calculates the plausibility of the "triple" ⟨*topic entity, question, candidate entities*⟩ by defining a scoring function to select the best candidate answer entity, thereby facilitating multi-hop reasoning. For instance, methods that combine memory networks and interactive reasoning mechanisms include MemNN (Sukhbaatar et al., 2015), KVMemNN (Miller et al., 2016), and IRN (Zhou et al., 2018), etc. Another category of methods utilizes graph neural networks for multi-hop reasoning, with representative models such as GraftNet (Sun et al., 2018), SGReader (Xiong et al., 2019), PullNet (Sun et al., 2019), 2HR-DR (Han et al., 2020), and HyperTransformer (Heo et al., 2022). Additionally, there are typical semantic matching methods such as EmbedKGQA (Saxena et al., 2020). These embedding-based multi-hop KGQA methods can reduce the dependence on complex rules and manually defined templates. At the same time, the KGE method makes entities and relations have uncertain semantics, and can well capture the semantic correlation between them. Therefore, inspired by embedding-based multi-hop KGQA, we have designed a KEQA framework that can adapt to different KGE models. Although the embedding-based multi-hop KGQA model has end-to-end training capabilities, its interpretability is relatively limited. To enhance the interpretability of the multi-hop KGQA model, some research has introduced path-based reasoning methods (Chen et al., 2019; Das et al., 2018; Lee et al., 2021; Niu et al., 2021; Qiu et al., 2020; Zhou et al., 2018). Due to the presence of misleading or irrelevant relations in the KG, the model cannot actively correct these erroneous relations during reasoning. These erroneous relations will be further accumulated and propagated along with path extension, resulting in the failure of reasoning. To alleviate the above issues, recent studies have applied RL (Chen et al., 2019; Han et al., 2020; Lin et al., 2018; Lv et al., 2020; Zhu et al., 2022) to the multi-hop KGQA, transforming it into a Markov decision process. By constructing an RL agent that simulates the dynamic interaction with the KG, a path selection strategy is learned.

However, RL still faces challenges in addressing multi-hop KGQA. Firstly, the agent cannot acquire enough information from the environment and only relies on the current limited state to make decisions, which reduces the reliability of the strategy (Cui, Peng, Xiao et al., 2023). Secondly, training data in multi-hop KGQA typically appear in the form of "question–answer" pairs rather than an explicit path, i.e., ⟨*question, reasoning paths, answer*⟩, which makes the model unable to reason deeply and solve complex problems and can only answer questions through local entity matching. To enhance the model's reasoning capabilities, some research (He et al., 2021) has proposed the construction of a teacher-student model that incorporates bidirectional reasoning to enhance the learning of intermediate entity distributions, thereby providing supervision signals for multi-hop KGQA. While these additional intermediate supervision signals are beneficial for multi-hop KGQA, they are not sufficient to address all its challenges. Consequently, we construct three different types of information from multi-perspective, i.e., expectation embedding, instruction-guided embedding, and path-aware embedding. Carefully designed information can not only be integrated into a well-designed KEQA framework as general knowledge, but can also be combined with the A3C policy learning algorithm to provide a more reliable foundation for the agent's decision-making.

To the best of our knowledge, this is the first attempt to incorporate multi-perspective information into an RL framework to solve complex multi-hop KGQA.

3. Multi-perspective information fusion reasoning network

In this section, we first define the problem of KGQA and then introduce our MPIFRN model. The MPIFRN model mainly consists of a Knowledge Graph Embedding (KGE) module, a Knowledge Embedding-based Question Answering (KEQA) module, a multi-perspective information module, and a Reinforcement Learning (RL) framework. Among these components, the KEQA framework utilizes the embedding vectors learned by the KGE module to evaluate the plausibility of triples. Simultaneously, by constructing an RL framework that integrates multi-perspective information with the RL agent, the agent can dynamically interact with the knowledge graph, enhancing both the model's performance and interoperability.

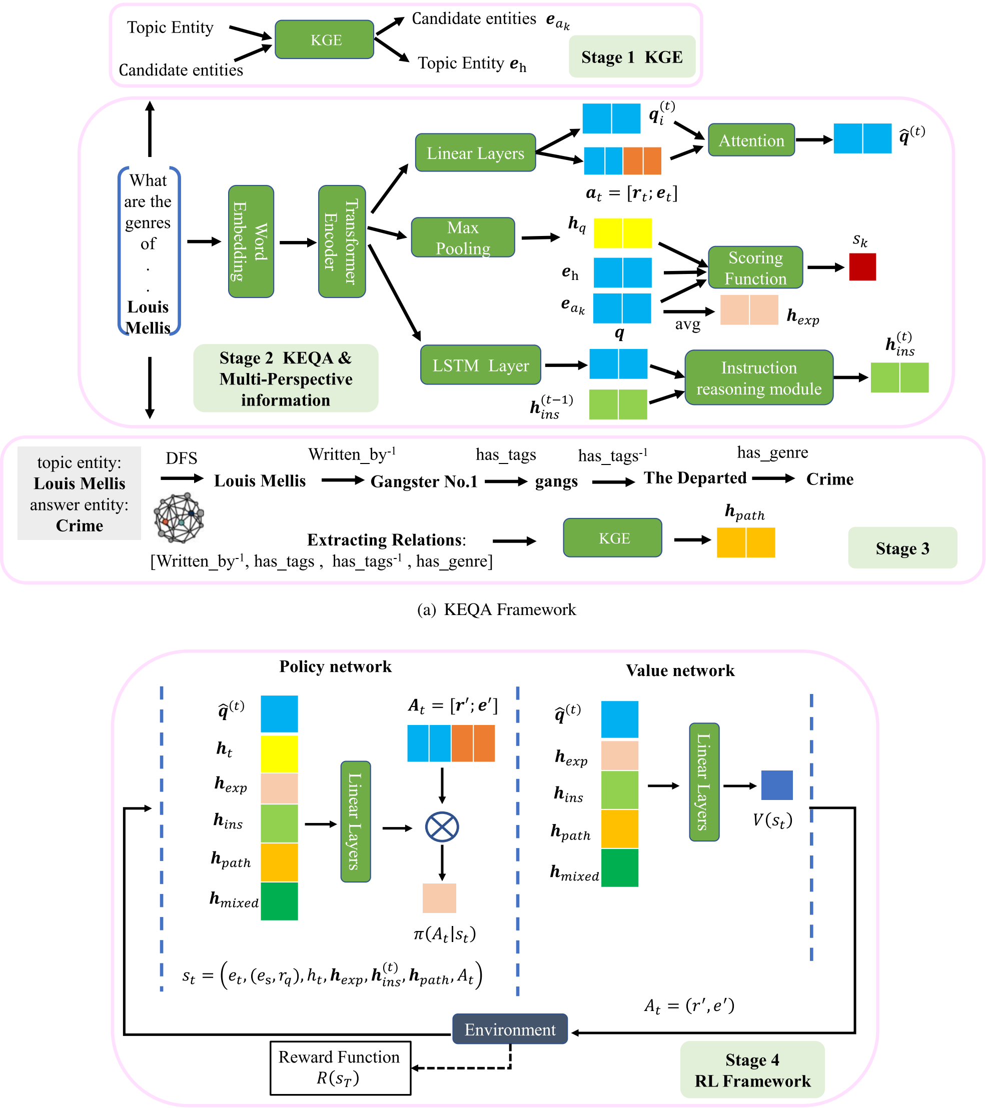
3.1. Formal problem definition

A knowledge graph $\mathcal{G} = (\mathcal{E}, \mathcal{R})$ is a structured data model that represents associations between entities, relations, and attributes, where $\mathcal{E}$ is the set of entities, and $\mathcal{R}$ is the set of relations. The knowledge graph $\mathcal{G}$ contains a large number of directed links $\mathcal{K}$, such as $\mathcal{K} \subseteq \mathcal{E} \times \mathcal{R} \times \mathcal{E}$. A triple can be represented as $(e_h, r, e_t) \in \mathcal{K}$, with $e_h, e_t \in \mathcal{E}$ denoting subject and object entities respectively and $r \in \mathcal{R}$ the relation between them. A fundamental problem in the KGQA task involves taking a natural language question q and the subject entity or topic entity $e_h \in \mathcal{E}$ mentioned within it as the starting point for reasoning. The task of KEQA is to efficiently search on knowledge graph $\mathcal{G}$ to find an entity $e_t \in \mathcal{E}$ that can correctly answer the question q.

3.2. Model overview

In this paper, our core idea is to incorporate carefully crafted multi-perspective information into the RL framework to solve multi-hop KGQA. To achieve this, our model consists of four components, i.e., knowledge graph embedding (KGE) module, knowledge embedding-based question answering (KEQA) module, multi-perspective information module, and reinforcement learning (RL) framework.

The KGE module is responsible for mapping entities and relations to semantic vectors within the same embedding space. Then, the KEQA framework is utilized to estimate the plausibility of "triple" ⟨*topic entity, question, answer entity*⟩. The multi-perspective information module is mainly responsible for generating multi-perspective information, including expectation embedding, instruction-guided embedding, and path-aware embedding. Due to the reliance on RL for reasoning in multi-hop question answering, we construct an RL agent. By integrating the multi-perspective fusion with RL framework, the agent can dynamically interact with the KG to learn the optimal strategy, thereby enhancing the model's interpretability. The overall workflow of our proposed method is shown in Fig. 1, and we will elaborate on each component in the following sections.



(a) KEQA Framework

(b) RL Framework

Fig. 1. The model architecture of our proposed multi-perspective Information Fusion Reasoning Network. Firstly, we utilize the KGE method to map entities and relations to distributed vectors within a low-dimensional vector space. Secondly, the KEQA framework is utilized to estimate the plausibility of "triple" ⟨*topic entity, question, candidate entities*⟩. We treat this probability distribution as the weight distribution of candidate entities. Then, we perform a weighted average of embedding vectors for all candidate entities to obtain the expectation embedding. Additionally, we also utilize an instruction reasoning module to generate instruction-guided embedding and construct path-aware embedding. Finally, we incorporate multi-perspective information into an RL framework to guide the agent to explore different sequences of actions in order to find promising paths and answers.

3.3. Knowledge graph embedding

Knowledge Graph Embedding (KGE) method aims to map entities and relations in KG into representations within a low-dimensional vector space. The core purpose of this mapping process is to capture the semantic associations between entities and relations, so that these associations are preserved in the semantic vector space, thereby providing an effective tool for measuring the strength of relationships between entities.

In this paper, we apply four KGE methods, i.e., ConvE, TuckER, DistMult, and ComplEx. Among them, the plausibility of the "triple" (e_h, r, e_t) is measured by the scoring function, and different KG embedding methods use different scoring functions. This evaluation process is crucial for relational reasoning and helps deepen our understanding and application of KGE. Taking ComplEx as an example, it maps entities and relations to the complex vector space, and can better capture the semantic relationship between entities and relations through complex vector representation and complex dot product calculation. Specifically, given $e_h, e_t \in \mathcal{E}$ and $r \in \mathcal{R}$, $\boldsymbol{e}_h, \boldsymbol{r}, \boldsymbol{e}_t \in \mathbb{C}^d$ are generated by ComplEx, then, a ComplEx scoring function is defined as follows:

$$\phi(e_h, r, e_t) = \mathrm{Re}\left(\langle \boldsymbol{e}_h, \boldsymbol{r}, \bar{\boldsymbol{e}}_t \rangle\right) \tag{1}$$

$$= \mathrm{Re}\Big(\sum_{k=1}^{d} e_h^{(k)} e_r^{(k)} \bar{e}_t^{(k)}\Big) \tag{2}$$

where Re() denotes the real part of the complex vector, and ⟨⟩ denotes the inner product of complex vectors. As for model training, firstly, an embedding vector is initialized for each entity and relation. Secondly, for each correct triple $(e_h, r, e_t) \in \mathcal{K}$ and incorrect triple $(e_h', r', e_t') \notin \mathcal{K}$, the model assigns scores in a manner that $\phi(e_h, r, e_t) > 0$ and $\phi(e_h', r', e_t') < 0$, respectively. Finally, the model minimizes the binary cross-entropy loss using optimization algorithms such as gradient descent to update the embedding vectors.

3.4. Knowledge embedding based question answering

Since KEQA mainly utilizes KGE to solve multi-hop questions, we first need to encode the question, and then measure the plausibility of the "triple" ⟨*topic entity, question, answer entity*⟩ in the KG embedding space. Specifically, an embedding layer first encodes the question q to obtain the word embedding. Then, the word embedding sequence is fed into a Transformer encoder to obtain hidden states. Finally, a max-pooling layer is applied to get the question representation $\boldsymbol{h}_q \in \mathbb{R}^{d\times 1}$:

$$\boldsymbol{h}_q = Max - Pooling(Transformer - Encoder \\ (Word - Embedding(q))) \tag{3}$$

Furthermore, as described in Section 3.3, pre-trained KG embeddings are applied to initialize the answer entity $\{e_a | e_a \in \mathcal{E}\}$ and the topic entity $e_h \in \mathcal{E}$ within the question q, obtaining their respective embedding representations as$\{\mathbf{e}_{a_k}\}_{k=1}^{|\mathcal{E}|} \in \mathbb{R}^d$ and $\boldsymbol{e}_h$. Subsequently, the plausibility of each candidate answer entity is evaluated using a scoring function $\phi(\cdot)$:

$$s_k = \phi(\boldsymbol{e}_h, \boldsymbol{h}_q, \boldsymbol{e}_{a_k}) \tag{4}$$

The KGE module is trained by minimizing the binary cross-entropy loss, which is achieved by comparing the sigmoid of the scores with the target labels.

3.4.1. Expectation embedding

In this part, we will introduce expectation embedding. The trained KGE models are leveraged to calculate the probability distribution of candidate entities through the "triple" ⟨*topic entity, question, candidate entities*⟩, and we treat this probability distribution as the weight distribution of candidate entities. Then, we perform a weighted average of embedding vectors for all candidate entities to obtain the expectation embedding, denoted as $\boldsymbol{h}_{exp}$. At last, $\boldsymbol{h}_{exp}$ is injected into the RL framework as part of the basis for the agent's decision-making. The expectation embedding $\boldsymbol{h}_{exp}$ has the following advantages. First, the potential candidate entity vectors are weighted as supervisory information and combined with the reasoning process of RL, which can enhance the learning of the policy network and improve the exploration efficiency of the agent. Second, $\boldsymbol{h}_{exp}$ is derived through a scoring function, rendering it versatile and applicable to different KGE methods.

3.4.2. Instruction-guided embedding

In this section, we will introduce a detailed explanation of the process by which the instruction-guided embedding is generated. Intuitively, we need to focus on specific parts of the question at different stages of multi-hop question answering, and this process can be controlled by the instruction reasoning module. The input of the instruction reasoning module includes a query embedding $\boldsymbol{q}$ and the instruction-guided embedding $\boldsymbol{h}_{ins}^{(t-1)}$ generated from the previous reasoning step. The instruction-guided embedding $\boldsymbol{h}_{ins}^{(0)}$ is initialized as a zero vector at the beginning of reasoning. Given a question $q = (q_1, q_2, q_3, \ldots, q_n)$, where each token q_i is initialized using pre-trained GloVe word embeddings. To capture richer semantic information, we employ a Transformer encoder to encode the GloVe-initialized question and obtain the hidden states of the query. After that, in order to reduce the dimension, we use LSTM to encode the hidden states of the query again to get a set of hidden states $\{\boldsymbol{h}_i\}_{i=1}^{l}$, where $\boldsymbol{h}_i \in \mathbb{R}^d$ and l is the length of the query. At the same time, we regard the last hidden state as the query representation, i.e., $\boldsymbol{q} = \boldsymbol{h}_l$. Let $\boldsymbol{h}_{ins}^{(t)} \in \mathbb{R}^d$ denote the instruction-guided embedding at the tth step of reasoning. The $\boldsymbol{h}_{ins}^{(t)}$ is learned using the following method:

$$\boldsymbol{h}_{ins}^{(t)} = \sum_{i=1}^{l} \alpha_i^{(t)} \boldsymbol{h}_i \tag{5}$$

$$\alpha_i^{(t)} = \mathrm{softmax}_i \left(\boldsymbol{W}_q \left(\boldsymbol{q}^{(t)} \odot \boldsymbol{h}_i \right) + \boldsymbol{b}_q \right) \tag{6}$$

$$\boldsymbol{q}^{(t)} = \boldsymbol{W}^{(t)} \left[\boldsymbol{h}_{ins}^{(t-1)}; \boldsymbol{q} \right] + \boldsymbol{b}^{(t)}, \quad \boldsymbol{h}_{ins}^{(0)} = \boldsymbol{0}, \quad t > 0 \tag{7}$$

where $\boldsymbol{W}^{(t)} \in \mathbb{R}^{d\times 2d}$ and $\boldsymbol{W}_q \in \mathbb{R}^{d\times d}$ are parameters to learn. By repeating the above process, after n steps of reasoning, we can obtain a series of instruction-guided embedding $\left\{ \boldsymbol{h}_{ins}^{(t)} \right\}_{t=1}^{n}$.

We can treat the instruction-guided embedding as a guiding signal to track the state of multi-hop reasoning by dynamically capturing query representations at different steps and combining it with the RL framework to provide clues for the agent's decision-making.

3.4.3. Path-aware embedding

In RL, the agent only receives delayed and sparse rewards from the environment upon reaching terminal states, which not only reduces the efficiency of exploration but also increases the risk of biased reasoning. Inspired by the bidirectional search paths in graphs, we have devised path supervision information. The agent can make wiser decisions at each step of the reasoning process by integrating global path supervision information, rather than relying solely on sparse rewards at the end of the reasoning process.

Specifically, inspired by a Bi-directional search for path verification (Xiong et al., 2017), we apply a Depth-First Search (DFS) on the KG $\mathcal{G}$ to find the path from the topic entity to the answer entity. Generally, assuming $Path = (e_h, r_1, e_1, r_2, e_2, \ldots, r_T, e_a)$, we extract relations from $Path$ to get $r_{path} = (r_1, r_2, r_3, \ldots, r_T)$. Then, we use pre-trained KG embeddings obtained in Section 3.3 to encode these relations to obtain $\boldsymbol{r}_{path} = \{\boldsymbol{r}_i\}_{i=1}^{T}$, where $\boldsymbol{r}_i \in \mathbb{R}^d$ and T is max number of hops. Due to the max number of hops for multi-hop reasoning in this paper does not exceed 3, we concat these relation embeddings to form $\boldsymbol{r}'_{path} = [\boldsymbol{r}_1; \boldsymbol{r}_2; \ldots; \boldsymbol{r}_T] \in \mathbb{R}^{T\times d}$. In order to be consistent with the dimensions of $\boldsymbol{h}_{exp}$ and $\boldsymbol{h}_{ins}$, we stack the corresponding elements in $\boldsymbol{r}'_{path}$ along the column axis and add them up to obtain Path-aware embedding $\boldsymbol{h}_{path}$:

$$\boldsymbol{h}_{path} = \sum_{k=0}^{T} r'_{path}[k, :] \in \mathbb{R}^d \tag{8}$$

Through the above operation, $\boldsymbol{h}_{path}$ now contains path encoding information. Treating $\boldsymbol{h}_{path}$ as global supervision and integrating it with the RL framework can enhance the path selection ability of the intelligent agent and mitigate biased reasoning.

3.5. Reinforcement learning formulation

The Markov Decision Process (MDP) provides powerful support for modeling and solving sequential decision problems. Starting from the source entity, the agent selects potential relations from the KG according to the strategy and traverses to a new entity until reaching the target entity. Naturally, path search is transformed into a reinforcement learning process. Specifically, MDP is mainly composed of the following parts.

States. The agent is currently in state $s_t = (e_t, (e_s, r_q)) \in \mathcal{S}$, where e_t denotes the entity visited at step t, and (e_s, r_q) are the source entity and query relation. In multi-hop question answering, e_s is also viewed as a topic entity. The above simple state information is not enough, we need to introduce more information to describe the state of the agent. Due to the rich state information, it can help the agent to make more reliable decisions. In order to provide more reliable information for the agent, we also construct multi-perspective information, including $\boldsymbol{h}_{exp}$, $\boldsymbol{h}_{ins}$, and $\boldsymbol{h}_{path}$, which are described in detail in Sections 3.4.1, 3.4.2, and 3.4.3, respectively. Meanwhile, we also track the search history h_t which is derived from the historical sequence of decisions made by the agent. The details of h_t are elaborated in Section 3.6. Then, the state s_t is expanded to $(e_t, (e_s, r_q), h_t, \boldsymbol{h}_{exp}, \boldsymbol{h}_{ins}^{(t)}, \boldsymbol{h}_{path})$. We will integrate multi-perspective state information with the subsequent policy network and value network in RL framework to address the issues of biased reasoning and low exploration efficiency in multi-hop KGQA.

Actions. When the agent arrives at state s_t, it may choose the set of actions $A \in \mathcal{A}$ that consists of the outgoing edges of e_t. Concretely, action spaces $A_t = \{(r', e') \mid (e_t, r', e') \in \mathcal{G}\}$. To terminate the agent's exploration within a fixed time step T, we consider adding a self-loop relation (r_{loop}, e_t) viewed as a "stop" action to each set A_t. If the agent selects the self-loop relation at time step t, it will remain at the current entity e_t and consider e_t as the predicted answer, while the process of path expansion terminates.

Transition. In RL, "State Transition" describes how the agent moves from one state to another in the process of continuously taking actions in the environment. A state transition function $\delta : S \times \mathcal{A} \rightarrow S$ is defined as $\delta(s_t, A_t) = \delta(e_t, (e_s, r_q), h_t, \boldsymbol{h}_{exp}, \boldsymbol{h}_{ins}^{(t)}, \boldsymbol{h}_{path}, A_t)$. Specifically, in state s_t, after the agent selects an action $a_t = (r_t, e_t) \in A_t$, according to the current optimal strategy, it arrives at the state $s_{t+1} = (e_{t+1}, (e_s, r_q), h_{t+1}, \boldsymbol{h}_{exp}, \boldsymbol{h}_{ins}^{(t+1)}, \boldsymbol{h}_{path}, A_{t+1})$, where $h_{t+1} = h_t \cap \{A_t\}$. Note that, $e_s, r_q, \boldsymbol{h}_{exp}$ and $\boldsymbol{h}_{path}$ are global information shared by all states, and will not change when a state transition occurs.

Rewards. If the agent arrives at a correct target entity at the end of an episode, it will get a terminal reward of 1 and 0 otherwise.

$$R_b(s_T) = \mathbb{1}(\boldsymbol{e}_s, \boldsymbol{r}_q, \boldsymbol{e}_T) \in \mathcal{G} \tag{9}$$

This type of reward is delayed and sparse, thus significantly slowing down the convergence speed of RL. In this paper, we still follow previous work using a reward-shaping method (Lin et al., 2018) to give the agent a soft reward based on a pre-trained KGE model. Formally, the soft reward is defined as follows:

$$R(s_T) = R_b(s_T) + (1 - R_b(s_T))\phi(\boldsymbol{e}_s, \boldsymbol{r}_q, \boldsymbol{e}_T) \tag{10}$$

where $\boldsymbol{e}_s$, $\boldsymbol{r}_q$ and $\boldsymbol{e}_T$ refer to the embeddings of the topic entity, query relation, and predicted entity, respectively, and the state $s_T = (e_T, (e_s, r_q), h_T, \boldsymbol{h}_{exp}, \boldsymbol{h}_{ins}^{(T)}, \boldsymbol{h}_{path}, A_T)$ denotes the terminal state. If the predicted entity e_T matches the answer entity, the agent will receive a reward of 1, otherwise, it will be rewarded $\phi(\boldsymbol{e}_s, \boldsymbol{r}_q, \boldsymbol{e}_T)$ according to a scoring function.

3.6. Policy network

In this section, we will elaborate in detail the integration of multi-perspective information with the policy network in RL. The policy network takes the state information of the agent as input and outputs a probability distribution over candidate actions. Concretely, a question $q = (q_1, q_2, \dots, q_n)$ is sequentially processed through pre-trained GloVe word embeddings initialization and Transformer encoder encoding to obtain the hidden states of q which denotes as $\boldsymbol{q} = (\boldsymbol{q}_1, \boldsymbol{q}_2, \dots, \boldsymbol{q}_n)$, where $\boldsymbol{q}_i \in \mathbb{R}^d$. Since we need to focus on different parts of q at different reasoning steps, a simple linear network is applied to dynamically update q to generate $\boldsymbol{q}^{(t)} \in \mathbb{R}^{d\times n}$:

$$\boldsymbol{q}^{(t)} = \mathbf{Tanh}(\boldsymbol{W}_q^{(t)} \cdot \boldsymbol{q} + \boldsymbol{b}^{(t)}) \tag{11}$$

where $\boldsymbol{W}_q^{(t)} \in R^{d\times d}$ and $\boldsymbol{b}^{(t)} \in R^{d\times 1}$ are parameters to learn. $\mathbf{Tanh}(\cdot)$ is the activation function.

Relation-augmented question representation At time step t, action spaces $A_t = \{(r', e') \mid (e_t, r', e') \in \mathcal{G}\}$. To get relation-augmented question representation $\hat{\boldsymbol{q}}^{(t)}$, we use an attention mechanism to align the relation derived from each action $a_t = (r_t, e_t) \in A_t$ with question as follows:

$$\beta_i' = \boldsymbol{W}_{rq} \cdot (\boldsymbol{r}_t \odot \boldsymbol{q}_i^{(t)}) \tag{12}$$

$$\alpha_i' = \frac{exp^{(\beta_i')}}{\sum_{j=1}^{n} exp^{(\beta_j')}} \tag{13}$$

$$\hat{\boldsymbol{q}}^{(t)} = \sum_{i=1}^{n} \alpha_i' \cdot \boldsymbol{q}_i^{(t)} \tag{14}$$

where $\boldsymbol{r}_t$ is the vector embedding of relation r_t through KGE, $\boldsymbol{q}_i^{(t)}$ is a dynamic update representation of the query token q_i, and $\boldsymbol{W}_{rq}$ are parameters to learn.

Integrating Multi-Perspective Information into Policy Network Every entity e_t and relation r_t in action $a_t = (r_t, e_t) \in A_t$ is assigned a dense vector embedding $\boldsymbol{e}_t \in \mathbb{R}^d$ and $\boldsymbol{r}_t \in \mathbb{R}^d$ through KGE. As a result, the action $a_t = (r_t, e_t)$ is encoded into $\boldsymbol{a}_t = [\boldsymbol{r}_t; \boldsymbol{e}_t] \in \mathbb{R}^{2d}$ that denotes the concatenation of the relation embedding and the entity embedding. In addition, LSTM is utilized by us to encode the search history $h_t = (e_s, r_1, e_1, \dots, r_t, e_t)$ containing the sequence of observations and actions taken up to step t to get search history embedding $\boldsymbol{h}_t$:

$$\boldsymbol{h}_t = \mathbf{LSTM}(\boldsymbol{h}_{t-1}, \boldsymbol{a}_{t-1}), \quad t > 0 \tag{15}$$

$$\boldsymbol{h}_0 = \mathbf{LSTM}(\mathbf{0}, \boldsymbol{a}_0) \tag{16}$$

$$\boldsymbol{a}_0 = [\boldsymbol{r}_0; \boldsymbol{e}_s] \tag{17}$$

where $\boldsymbol{e}_s$ is the vector embedding of the topic entity and $\boldsymbol{r}_0$ is a special start relation connected with the topic entity. To further improve the performance of multi-hop question answering, we introduce a cross attention module to enhance the interaction among multi-perspective information. First, the cross attention between $\boldsymbol{h}_{exp}$ and $\boldsymbol{h}_{ins}$ is computed as:

$$\boldsymbol{Q} = \boldsymbol{h}_{exp}\boldsymbol{W}^{Q} \tag{18}$$

$$\boldsymbol{K} = \boldsymbol{h}_{ins}\boldsymbol{W}^{K} \tag{19}$$

$$\boldsymbol{V} = \boldsymbol{h}_{ins}\boldsymbol{W}^{V} \tag{20}$$

$$CrossAttention(\boldsymbol{h}_{exp}, \boldsymbol{h}_{ins}) = Attention(\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V}) \tag{21}$$

$$Attention(\boldsymbol{Q}, \boldsymbol{K}, \boldsymbol{V}) = softmax(\frac{\boldsymbol{Q}\boldsymbol{K}^{\boldsymbol{T}}}{\sqrt{d_k}})\boldsymbol{V} \tag{22}$$

Second, similarly, we calculate the cross attention among $\boldsymbol{h}_{exp}$, $\boldsymbol{h}_{ins}$, and $\boldsymbol{h}_{path}$ to get $\boldsymbol{h}_{mixed}$. Finally, we obtain $\boldsymbol{A}_t \in \mathbb{R}^{|A_t|\times 2d}$ by stacking the embeddings of all actions in the action spaces A_t. Therefore, the policy network is defined as:

$$\pi_\theta(a_t|s_t) = \sigma(\boldsymbol{A}_t \cdot \boldsymbol{W}_2 \cdot ReLU(\boldsymbol{W}_1 \cdot [\hat{\boldsymbol{q}}^{(t)}; \boldsymbol{h}_t; \boldsymbol{h}_{exp}; \boldsymbol{h}_{ins}; \boldsymbol{h}_{path}; \boldsymbol{h}_{mixed}])) \tag{23}$$

$$\boldsymbol{h}_{mixed} = CrossAttention(CrossAttention(\boldsymbol{h}_{exp}, \boldsymbol{h}_{ins}), \boldsymbol{h}_{path}) \tag{24}$$

where $\boldsymbol{A}_t$ denotes the encoding of action spaces, $[\hat{\boldsymbol{q}}^{(t)}; \boldsymbol{h}_t; \boldsymbol{h}_{exp}; \boldsymbol{h}_{ins}; \boldsymbol{h}_{path}; \boldsymbol{h}_{mixed}]$ denotes the vector concatenation of relation-augmented question representation, search history embedding, expectation embedding, instruction-guided embedding, path-aware embedding, cross attention among $\boldsymbol{h}_{exp}$, $\boldsymbol{h}_{ins}$, and $\boldsymbol{h}_{path}$. $\boldsymbol{W}_1$ and $\boldsymbol{W}_2$ are parameters to learn, σ is the softmax operator.

3.7. Optimization

Due to the difficulty of convergence in RL, the Asynchronous Advantage Actor-Critic (A3C) algorithm is introduced. By utilizing Actor-Critic architecture and integrating policy gradient and value function optimization, A3C helps to reduce variance during training and improve stability. Furthermore, the parallel architecture of A3C enables multiple agents or threads to interact with the environment simultaneously, thereby accelerating training. Hence, we employ A3C to optimize both the policy network $\pi(a_t|s_t; \theta)$ and the value network $V(s_t; \theta_v)$. The optimization procedure is summarized in Algorithm Mnih et al. (2016). First, we initialize the parameter vectors θ' and θ_v' for each thread. Then, we use the global shared parameter vectors θ and θ_v to synchronize the thread-specific parameters. Subsequently, the agent observes the current state s_t and selects an action a_t based on the policy function $\pi(a_t|s_t; \theta')$. After executing action a_t, the agent receives a reward $r(s_t)$ from the environment and updates the state to s_{t+1}. The above process is repeated until the agent reaches a terminal state s_t or the maximum number of hops T_{max}. The agent calculates the cumulative reward $R(s_t)$ based on the delayed reward $R(s_{T_{max}})$ according to Eqs. (26), (27). Next, we introduce the advantage function $A(s_t, a_t)$ to estimate the advantage of action a_t in state s_t based on the cumulative reward $R(s_t)$. Then, we calculate the accumulated gradients of the policy network and the value network according to Eqs. (29), (30), and (31). Finally, we perform asynchronous updates on the global shared parameter vectors θ and θ_v. The optimization process terminates when all threads have executed C_{max} epochs.

Specifically, we apply a simple fully connected layer to implement the value network:

$$V(s_t;\theta_v) = Sigmoid(\boldsymbol{W}_v \cdot [\boldsymbol{h}_t;\boldsymbol{h}_{exp};\boldsymbol{h}_{ins};\boldsymbol{h}_{path};\boldsymbol{h}_{mixed}]) \tag{25}$$

where $\boldsymbol{W}_v$ is a parameter to learn and $[\boldsymbol{h}_t;\boldsymbol{h}_{exp};\boldsymbol{h}_{ins};\boldsymbol{h}_{path};\boldsymbol{h}_{mixed}]$ denotes the vector concatenation of search history embedding, expectation embedding, instruction-guided embedding, path-aware embedding and cross attention among $\boldsymbol{h}_{exp}$, $\boldsymbol{h}_{ins}$ and $\boldsymbol{h}_{path}$. Since the agent only receives a reward when it reaches the end of an episode, for intermediate states, the agent can only obtain a reward of 0. Delayed and sparse rewards can significantly slow down the convergence speed of reinforcement learning. To address the above, we provide the agent with a soft reward $r(s_t)$ in intermediate state s_t. Thus, in state s_t, the cumulative reward $R(s_t)$ the agent gets is calculated as follows:

$$r(s_t) = \phi(\boldsymbol{e}_s, \boldsymbol{r}_q, \boldsymbol{e}_t) \tag{26}$$

$$R(s_t) = r(s_t) + \gamma^{T-t} R(s_T) \tag{27}$$

where $\boldsymbol{e}_s, \boldsymbol{r}_q, \boldsymbol{e}_t$ refer to embeddings of the topic entity, query relation, and intermediate entity, respectively. $r(s_t)$ is the intermediate and soft reward obtained by the agent in state s_t through a scoring function described in Section 3.5, $R(s_T)$ denotes the soft reward obtained by the agent in the terminal state s_T, γ is a discount factor, and T is the max number of hops. Moreover, an advantage function is introduced to estimate the additional benefit of taking action a_t relative to the average reward that would be obtained by executing the default policy:

$$A(s_t, a_t) = R(s_t) - V(s_t) \tag{28}$$

In the A3C algorithm, the loss function is designed to simultaneously optimize the policy network and the value function network. As a result, the loss function usually consists of two parts: policy loss and value function loss. The policy loss applies a policy gradient-based method to encourage the policy network to generate a better action distribution:

$$\mathcal{L}(\theta) = \mathbb{E}_{\tau\sim\pi_\theta} \sum_{t=0}^{T} A(s_t, a_t) \cdot log\pi(a_t|s_t;\theta) \tag{29}$$

where $\pi(a_t|s_t;\theta)$ is the probability that the policy network generates action a_t in state s_t, and $A(s_t, a_t)$ is the advantage function that denotes the advantage of taking action a_t in state s_t. As for the value network, the mean squared error (MSE) is usually used to optimize the value network loss:

$$\mathcal{L}(\theta_v) = \mathbb{E}_{\tau\sim\pi_{\theta_v}} \left[\frac{1}{2} \sum_{t=0}^{T} (R(s_t) - V(s_t;\theta_v))^2 \right] \tag{30}$$

Finally, the total loss is defined as:

$$\mathcal{L}(\theta,\theta_v) = \mathcal{L}(\theta) + \lambda\mathcal{L}(\theta_v) + \alpha \cdot H(\pi(\cdot|s_t;\theta)) \tag{31}$$

where λ, α are hyperparameters, and $H()$ denotes the cross entropy of strategy $\pi(\cdot|s_t;\theta)$ which can improve exploration and avoid converging to a suboptimal deterministic strategy.

4. Experiments

In this section, we outline the datasets used to evaluate our method, then describe the experimental setup in detail and present the obtained results.

Algorithm 1 Application of the Asynchronous Advantage Actor-Critic in Optimization Procedure

```
Input: Global shared parameter vectors $\theta$ and $\theta_v$; Thread-specific parameter vectors $\theta'$ and $\theta'_v$; Global shared counter $C = 0$; Max number of hops $T_{max}$; Maximum Iteration epochs $C_{max}$.
Output: Optimized parameters $\theta$ and $\theta_v$.
Initialize thread step counter $t \leftarrow 1$
repeat
    Reset gradients $d\theta \leftarrow 0$ and $d\theta_v \leftarrow 0$
    Synchronize thread-specific parameters $\theta' \leftarrow \theta$ and $\theta'_v \leftarrow \theta_v$
    $t_{start} = t$
    Get state $s_t$
    repeat
        Perform $a_t$ according to policy $\pi(a_t|s_t;\theta')$
        Receive reward $r(s_t)$ and new state $s_{t+1}$
        $t \leftarrow t+1$
    until terminal $s_t$ or $t - t_{start} == T_{max}$
    Get the delayed reward $R(s_{T_{max}})$
    for $i \in \{t-1, ..., t_{start}\}$ do
        Calculate the cumulative reward
        $R(s_i) \leftarrow r(s_i) + \gamma^{T_{max}-i} R(s_{T_{max}})$
        Accumulate gradients w.r.t. $\theta'$:
        $d\theta \leftarrow d\theta + \nabla_{\theta'}(R(s_i) - V(s_i;\theta'_v)) \cdot log\pi(a_i|s_i;\theta') + \beta \cdot \nabla_{\theta'} H(\pi(\cdot|s_i;\theta')$
        Accumulate gradients w.r.t. $\theta'_v$:
        $d\theta_v \leftarrow d\theta_v + \lambda \cdot \nabla_{\theta'_v}(R(s_i) - V(s_i;\theta'_v))^2$
    end for
    Perform asynchronous updates of $\theta$ using $d\theta$ and of $\theta_v$ using $d\theta_v$.
    $C \leftarrow C+1$
until $C > C_{max}$
```

4.1. Datasets

To evaluate the effectiveness of the proposed method, we conduct a series of experiments on three benchmark datasets for the multi-hop KBQA task, and the detailed statistics of the datasets are shown in Table 1. PathQuestion (PQ) (Zhou et al., 2018) is composed of selecting a set of entity pairs from two subsets of Freebase, and then using relations in the knowledge graph to extract paths connecting these entity pairs through the path extraction algorithm. Different question types are included in the PQ dataset, namely 2-hop (PQ-2H) and 3-hop (PQ-3H) questions. PQ-Mix is a mixed dataset that contains the mix of all questions in the PQ dataset, which aims to provide a more comprehensive and diverse dataset for evaluating the performance of models on multi-hop KGQA tasks.

PathQuestion-Large (PQL) is an extended version of the PQ dataset, which uses a larger knowledge graph but provides few training instances. Likewise, the PQL dataset also contains two types of questions, PQL-2H and PQL-3H. Among them, PQL-2H denotes the 2-hop questions, and PQL-3H denotes the 3-hop questions. PQL-Mix is a mixed dataset of PQL.

As a comprehensive extension of WikiMovies, MetaQA (Zhang et al., 2018) is an extensive KGQA dataset focusing on the movie domain. It contains over 400k single and multi-hop questions covering 1-hop, 2-hop, and 3-hop.

4.2. Baselines

In order to better verify the effectiveness of the proposed method, we extensively consider the following baselines for performance comparison: (1) reinforcement learning-based method: *SRN* (Qiu et al., 2020); (2) graph neural network-based methods: *GraftNet* (Sun et al., 2018), *SGReader* (Xiong et al., 2019), *PullNet* (Sun et al., 2019),

Table 1
Information about three benchmark datasets.

	PathQuestion			PathQuestion-Large			MetaQA		
	2H	3H	Mix	2H	3H	Mix	1H	2H	3H
Train	1528	4163	5691	1276	825	2101	96 106	118 980	114 196
Dev	189	515	704	158	102	260	9992	14 872	14 274
Test	191	520	711	160	104	264	9947	14 872	14 274
Entities	1056	1836	2256	5034	6505	6505	43 234	43 234	43 234
Relations	13	13	13	363	411	411	18	18	18
Triples	1211	2839	3377	4247	5597	5597	134 741	134 741	134 741

2HR-DR (Han et al., 2020), *HyperTransformer* (Heo et al., 2022); (3) embedding-based methods: *KVMemNN* (Miller et al., 2016), *EmbedKGQA* (Saxena et al., 2020); (4) comprehensive method that combines reinforcement learning-based and embedding-base: *ARN* (Cui, Peng, Xiao et al., 2023). We present a detailed description of these baselines as follows:

- **KVMemNN** A dedicated retrieval memory table is applied by KVMemNN to store important knowledge base facts in the form of key–value pairs, allowing for the rapid access and utilization of the stored information when confronted with tasks such as question–answering, reasoning, or complex reasoning.
- **SGReader** proposes a QA model based on incomplete knowledge base (KB) and text documents. It obtains question-related knowledge-assisted entity encoding through the graph attention mechanism and designs a gate mechanism to integrate entity knowledge in KB when encoding text.
- **GraftNet** employs adaptive graph convolution technology to effectively link different pieces of information by traversing a variety of complex dependencies within a heterogeneous graph, enabling the accomplishment of intricate multi-hop logical reasoning tasks.
- **PullNet** strategically utilizes the shortest path as a form of supervision during the training phase to identify and select the most direct and relevant connections within the graph. It employs its newly acquired domain knowledge to perform complex multi-hop reasoning on the selected subgraph.
- **SRN** Through a well-designed reinforcement learning framework, SRN effectively extends its cognitive scope into a KB and addresses multi-hop question answering tasks by dynamically expanding reasoning paths within the knowledge base, thereby achieving more comprehensive and context-rich answers.
- **EmbedKGQA** resolves complex problems through multi-hop reasoning, which aligns question embeddings extracted from RoBERTa with entity embeddings pre-trained on an extensive knowledge base.
- **2HR-DR** guides the subsequent reasoning process by iteratively updating relationship representations and entity states, achieved through the construction of a directed hypergraph convolutional network specifically designed for knowledge retrieval.
- **HyperTransformer** systematically constructs a question hypergraph and a knowledge hypergraph that is query-aware, achieving reasoning for questions by encoding and understanding complex associations spanning these two hypergraphs.
- **ARN** incorporates KGE as prior information into the RL framework, which not only enhances model interpretability but also improves the effect of multi-hop KGQA.

By comparing the above models, we are able to comprehensively evaluate the performance of our proposed approach on multi-hop KGQA tasks. During the validation process, we will consider ablation studies to ensure a thorough assessment of the method.

4.3. Implementation details

Similar to our proposed model, SRN and ARN also incorporate RL. Therefore, we refer to their experimental parameter settings throughout the experiment. Firstly, the pre-trained 300-dimensional Glove word embeddings are applied to initialize the Word Embedding layer, which also serves as the initial input of the question representation. Secondly, we use a Transformer encoder with 4 layers and 4 attention heads to further encode the question. For the entities and relations in the KG, we employ off-the-shelf KGE methods to obtain the corresponding vector representations.

In this paper, we set the dimension of the knowledge graph embeddings to 300. In addition, we also introduce a three-layer unidirectional LSTM with a hidden dimension of 200 to encode the search history information. In the experiment, some hyperparameters are vital and need to be set. We set the discount factor γ to 0.98 and the entropy coefficient α to 0.02. The value loss coefficient λ is tuned according to the performance of the model on the validation set, which can be selected amongst $\{0.2, 0.5, 1.0\}$. The number of threads in the A3C will be dynamically adjusted based on available GPU memory and could be chosen from amongst $\{3, 4, 5\}$. We use the Adam optimizer to optimize the model, where the batch size is set to 32 and the learning rate is set to 0.0001. To avoid overfitting, we evaluate the Hits@1 score on the validation set and stop training early accordingly. In beam search inference, we set the beam size to 5 to generate multiple paths of reasoning, aiming to achieve better results.

4.4. Experimental results and analysis

We use the Hits@1 score to evaluate the performance of the model. Hits@1 score refers to the proportion of answers predicted by the model that ranks highest among all queries. Table 2 summarizes experimental results on three benchmarks. From the overall experimental results, our proposed MPIFRN achieves good performance on the three datasets. Even when faced with the challenging PQ-mix, PQL, and MetaQA-3H, MPIFRN also shows expected results, which are attributed to the well-designed multi-perspective information. We observe that for Graph neural network-based methods, i.e., GraftNet and SGReader, these models do not perform well on MetaQA-3H compared to MetaQA-1H and MetaQA-2H. We hypothesize that this is due to the exponential increase in the number of candidate relations and entities with the increase in reasoning hops, leading to a drastic expansion of the search space and making it difficult to find answer entities. However, our proposed MPIFRN can keep a promising performance on the MetaQA dataset, with the Hits@1 score even improving from MetaQA-2H to MetaQA-3H. MPIFRN integrates information from multiple perspectives in RL, enabling the agent to avoid the noise or erroneous information introduced at path expansion and prevent decision failures due to the increase in reasoning hops.

Furthermore, as for PullNet, it constructs heterogeneous subgraphs related to a specific problem through a heuristic iterative process and extracts answers from the constructed subgraphs using convolutional networks. While achieving competitive results on MetaQA, this approach heavily relies on heuristic algorithms. Our method models multi-hop KGQA into a sequential decision-making task through the introduction of an RL framework, achieving controllable reasoning, thus yielding better results on MetaQA-3H compared to PullNet. In addition, when it comes to HyperTransformer and 2HR-DR, they each introduce hypergraphs and encode entities and relations using convolutional networks to facilitate complex problem reasoning. By comparison, except

Table 2
Results (%Hits@1) on the test set of three benchmarks. The best result is emphasized in bold.

	PathQuestion			PathQuestion-Large			MetaQA		
	2H	3H	Mix	2H	3H	Mix	1H	2H	3H
KVMemNN	91.50	79.40	85.20	70.50	63.40	68.60	93.50	84.30	53.80
SGReader	–	–	–	71.90	89.30	–	96.70	80.70	61.00
GraftNet	–	–	–	70.70	91.00	–	97.00	94.80	77.70
PullNet	–	–	–	–	–	–	97.00	**99.90**	91.40
ARN	98.95	90.58	93.67	97.50	97.12	98.48	97.12	94.92	97.06
EmbedKGQA	–	–	–	–	–	–	97.50	98.80	94.80
SRN	96.30	89.20	89.30	78.60	77.50	78.30	97.00	95.10	75.20
2HR-DR	–	–	–	75.50	92.10	–	**98.80**	93.70	–
HyperTransformer	96.40	90.30	89.50	90.50	95.40	94.50	–	–	–
MPIFRN (ours)	**99.47**	**92.50**	**96.91**	**98.12**	**98.08**	**99.62**	97.13	96.19	**97.20**

for MetaQA-1H, our method outperforms the aforementioned two models, indicating the effectiveness of path-based reasoning approaches. Graph neural network-based methods are only focused on finding predicted answers and lack interpretability, whereas our approach incorporates RL, enhancing interpretability.

We also observed that the performances of the model based on the embedding method, i.e., KVMemNN, dropped particularly significantly on PQL compared to PQ. Due to capacity limitations, key–value memories in KVMemNN cannot handle more triples that are derived from PQL containing more complex relations and entities.

Compared to RL-based models, i.e., SRN, our model also achieves better performance, especially in PQL and MetaQA-3H. This indicates that integrating additional environmental information with RL can significantly enhances the exploration efficiency of the agent, thereby improving model performance.

As for the comprehensive model that combines reinforcement learning-based and embedding-based approaches, i.e., ARN, our model's overall performance is also superior to ARN, which highlights the effectiveness of multi-perspective information. Due to its utilization of only anticipation information, ARN cannot effectively address the issue of useless exploration. We combine rich multi-perspective information with RL to enable the agent to better perceive the environment and make more reasonable decisions, thereby avoiding getting trapped in locally biased reasoning.

In short, the above comparative analysis of different methods demonstrates that our method integrating multi-perspective information with RL is an effective strategy for solving multi-hop KGQA.

4.5. Effectiveness of multi-perspective information

Since expectation embedding obtained by using **ConvE** (Dettmers et al., 2018) as the KGE method has achieved better results than DistMult (Yang, Yih et al., 2015), ComplEx (Trouillon et al., 2016), and TuckER (Balazevic et al., 2019), our experiments mainly focus on ConvE. In order to explore the effectiveness of multi-perspective information, we add multi-perspective information to the MPIFRN base model step by step.

As shown in Table 3, Exp-e, Ig-e, and Path-e refer to expectation embedding $\boldsymbol{h}_{exp}$, instruction-guided embedding $\boldsymbol{h}_{ins}$, and path-aware embedding $\boldsymbol{h}_{path}$, respectively. For the MPIFRN base, we only use search history embedding $\boldsymbol{h}_t$ which is reflected in Eqs. (23) and (25), thus the policy network and value network can be respectively denoted as $\pi_\theta(a_t|s_t) = \sigma(\boldsymbol{A}_t \cdot \boldsymbol{W}'_2 \cdot ReLU(\boldsymbol{W}'_1 \cdot [\hat{\boldsymbol{q}}^{(t)}; \boldsymbol{h}_t]))$ and $V(s_t; \theta_v) = Sigmoid(\boldsymbol{W}'_v \cdot [\boldsymbol{h}_t])$. Similarly, "w/ Exp-e" denotes adding expectation embedding $\boldsymbol{h}_{exp}$ to the policy network and value network to get $[\boldsymbol{h}_t; \boldsymbol{h}_{exp}]$. "w/ Exp-e, w/Ig-e" denotes continuing to add instruction-guided embedding $\boldsymbol{h}_{ins}$ to the policy network and value network to obtain $[\boldsymbol{h}_t; \boldsymbol{h}_{exp}; \boldsymbol{h}_{ins}]$. "w/ Exp-e, w/ Ig-e, w/ Path-e" means continuing to add path-aware embedding $\boldsymbol{h}_{path}$ to the policy and value network to get $[\boldsymbol{h}_t; \boldsymbol{h}_{exp}; \boldsymbol{h}_{ins}]$. Due to the concatenation of these embeddings into different dimensions, it is necessary to introduce learnable parameters for each type of joint encoding. Therefore, the policy network and value network can be uniformly denoted as $\pi_\theta(a_t|s_t) = \sigma(\boldsymbol{A}_t \cdot \boldsymbol{W}'_2 \cdot ReLU(\boldsymbol{W}'_1 \cdot [;]))$ and $V(s_t; \theta_v) = Sigmoid(\boldsymbol{W}'_v \cdot [;])$, respectively, where $\boldsymbol{W}'_1$ and $\boldsymbol{W}'_2$ denote the learnable parameters, and [;] denotes vector concatenation. As shown in Table 3, the model with Exp-e improves the Hits@1 score over the plain MPIFRN base model by an average of 0.55 points on all datasets. This reveals that it is useful to incorporate the global KGE as expectation embedding into RL framework, which can reduce the agent's aimless exploration to a certain extent. By continuing to add instruction-guided embedding, i.e., "w/ Exp-e, w/ Ig-e", the Hits@1 score also increases by an average of 0.95 points. This illustrates that it is essential to utilize instruction-guided embedding as a guiding signal to track the state of multi-hop reasoning. The improvement in model performance benefits from the agent easing biased reasoning by dynamically capturing query representations based on attention mechanism at different steps.

By further including path-aware embedding, i.e., "w/ Exp-e, w/ Ig-e", w/ Path-e, the Hits@1 score increases by 0.42 points. This indicates that the addition of path-aware embedding is helpful. However, as the number of hops increases, noise may also be introduced, i.e., erroneous relations will be further accumulated and propagated along with path expansion, thus only leading to limited improvement.

4.6. Effectiveness of different KGE methods

To integrate KGE information into RL, we design the KEQA framework. By measuring the plausibility of the "triple" $\langle topic\ entity, question, candidate\ entities \rangle$ in the KGE space, the final weighted candidate entity vector is obtained as expectation embedding. This process is described in Section 3.4.1. In order to study the impact of different KGE methods, we choose four off-the-shelf KGE models, i.e., ComplEx (Trouillon et al., 2016), DistMult (Yang, Yih et al., 2015), TuckER (Balazevic et al., 2019), and ConvE (Dettmers et al., 2018) to apply to the KEQA framework, and evaluate them on three datasets. For simplicity, we choose the setting of MPIFRN-base with "w/ Exp-e, w/ Ig-e, w/ path-e" in Table 3.

As shown in Table 4, our proposed KEQA framework achieves good performance on multiple datasets, e.g., the Hits@1 scores of ConvE are 97.10%, 98.67%, and 99.21% on PQL-3H, PQL-mix, and PQ-2H, respectively, indicating the effectiveness of our framework. Meanwhile, we also find that different KGE methods are suitable for different datasets, e.g., DistMult performs relatively well on PQ-2H and MetaQA, but does not meet expectations on PQ-3H and PQ-mix. For the PQ datasets, we notice a significant decline in the model's performance with an increase in the number of reasoning hops, e.g., the Hits@1 score of ComplEx drops from 96.73% to 88.27%, and the Hits@1 score of DistMult drops from 97.78% to 83.66%. This indicates that KGE methods are sensitive to the number of reasoning hops. We hypothesize that this is due to the fact that as the number of hops increases, the number of candidate relations and entities increases exponentially, and the search space expands dramatically, making answer prediction difficult. Despite the above shortcomings, expectation embedding applied in the KEQA framework, as a form of global supervision information, can fully take into account the distribution of candidate entities and provide more reliable exploration for agent in RL.

Table 3
Ablation studies (%Hits@1) on the effectiveness of multi-perspective information.

Model	PathQuestion			PathQuestion-Large			MetaQA			Avg.
	2H	3H	Mix	2H	3H	Mix	1H	2H	3H	
MPIFRN base	97.88	89.49	93.68	94.38	94.87	97.22	95.67	93.86	95.69	94.75
w/ Exp-e	98.43	90.19	94.45	95.32	95.19	97.73	96.07	94.24	96.12	95.30
w/ Exp-e w/ Ig-e	98.95	90.64	94.80	96.57	96.15	98.23	96.42	95.33	96.78	96.25
w/ Exp-e w/ Ig-e w/ Path-e	99.21	92.05	95.92	97.03	97.10	98.67	97.04	96.03	96.96	96.67

Table 4
Experimental results (%Hits@1) on different KGE methods utilized in KEQA framework. The best score is in bold.

	PathQuestion			PathQuestion-Large			MetaQA		
	2H	3H	Mix	2H	3H	Mix	1H	2H	3H
DistMult	97.78	83.66	86.64	92.08	91.35	91.92	96.96	93.31	95.48
ComplEx	96.73	88.27	92.26	93.63	93.27	95.46	96.98	94.59	94.75
TuckER	98.43	90.51	91.98	96.25	96.64	97.85	96.99	93.26	96.70
ConvE	**99.21**	**92.05**	**95.92**	**97.03**	**97.10**	**98.67**	**97.04**	**96.03**	**96.96**

Table 5
Experimental results (%Hits@1) about cross attention on multi-perspective information.

Model	PathQuestion			PathQuestion-Large			MetaQA			Avg.
	2H	3H	Mix	2H	3H	Mix	1H	2H	3H	
MPIFRN base	97.88	89.49	93.68	94.38	94.87	97.22	95.67	93.86	95.69	94.75
w/ Exp-e, Ig-e, Path-e	99.21	92.05	95.92	97.03	97.10	98.67	97.04	96.03	96.96	96.67
w/ Exp-e, Ig-e, Path-e, w/ Cross attention	99.47	92.50	96.91	98.12	98.08	99.62	97.13	96.19	97.20	97.16

4.7. The impact of multi-perspective information on reinforcement learning

Due to the integration of multi-perspective information in RL to guide the decision-making of the agent, we conduct a series of experiments to observe the impact of multi-perspective information on RL. We randomly select four datasets, i.e., PQ-3H, PQ-mix, PQL-mix, and MetaQA-3H, and continuously add multi-perspective information to the base model. We observe the change in the Hits@1 score on the validation set as the training episodes increase. An episode denotes that one of the sub-threads has completed training on a batch of data.

Through observation, the results are shown in Fig. 2, we find that, firstly, for all the datasets, models with the integration of multi-perspective information are able to rapidly achieve Hits@1 score that surpasses the model without multi-perspective information in the early stages of training. Secondly, models with the incorporation of all perspectives can converge to the highest Hits@1 score on the validation set. This indicates that the introduction of multi-perspective information can enhance the exploration efficiency and performance of the agent. We believe that the main reason is that this multi-perspective information can provide the agent with a certain prior environmental understanding. When combined with policy learning, it enables the agent to avoid getting stuck in local optima and reduces ineffective exploration.

4.8. Cross attention on multi-perspective information

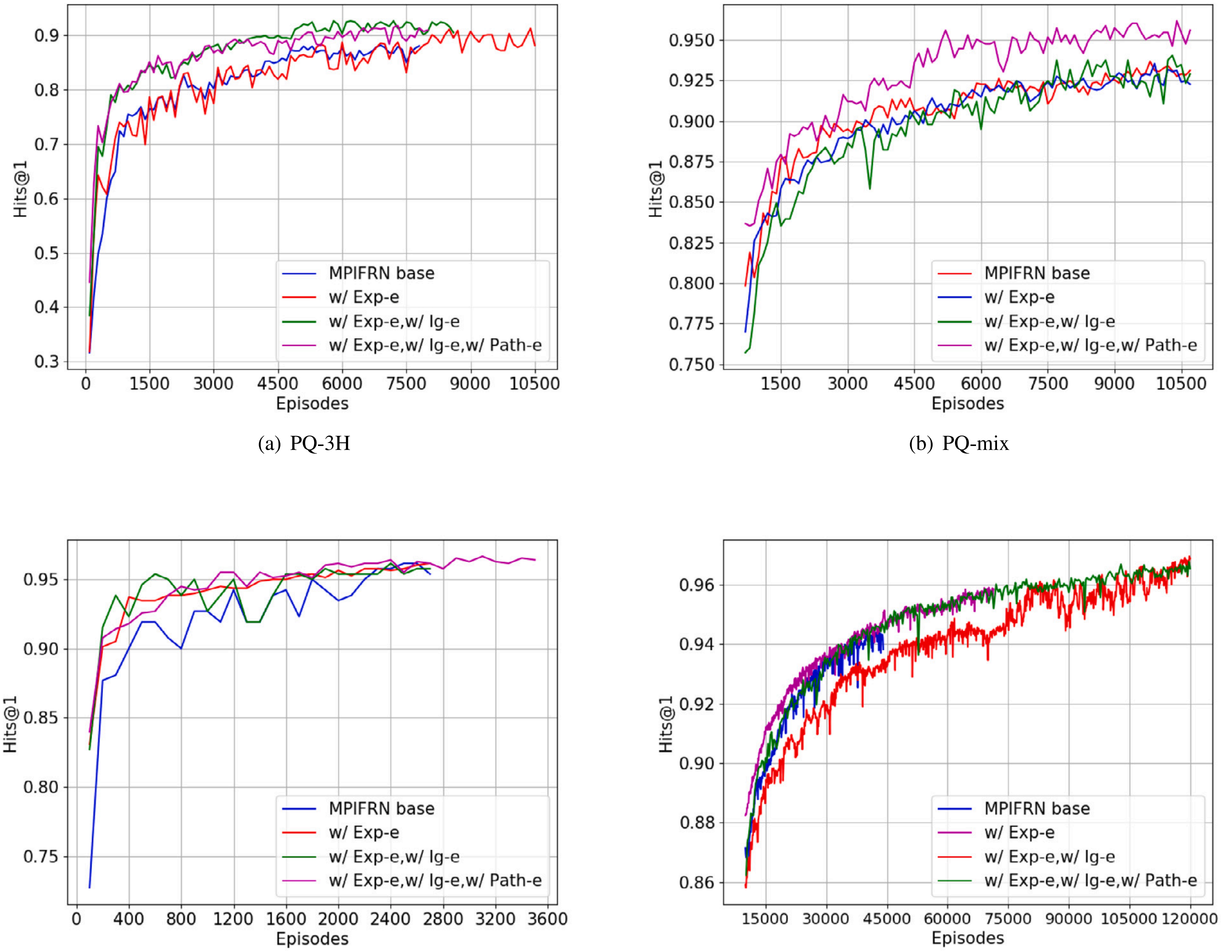
In this section, we will explore the impact of interactions between multi-perspective information on model performance. We calculate the cross-attention on multi-perspective information to obtain $\boldsymbol{h}_{mixed}$ according to Eq. (24). As shown in Table 5, compared to without cross attention, the model with cross attention, i.e., w/ Cross attention, improves by an average of 0.5 percentage points across all datasets, which demonstrates the effectiveness of cross attention. Specifically, compared with PQ and MetaQA, despite PQL-large providing fewer training instances, the cross-attention mechanism enables the model to achieve an average improvement of 1.0 on PQL-large, whereas PQ and MetaQA only saw increases of 0.3 and 0.2, respectively. This also indicates from another view that the interaction of information from different perspectives can enhance the model's performance.

4.9. Effect of the hyper-parameter

In the experiment, we find that the hyperparameter λ, as described in Eq. (31), which adjusts the ratio between policy loss and value loss, has a different impact on the experimental results. Consequently, we fine-tune λ within the range of $\{0.2, 0.5, 1.0\}$ based on the model's performance on the validation set. We randomly select two datasets, i.e., PQL-mix and MetaQA-2H, to study the impact of different λ on the model's performance. As shown in Fig. 3, as the training episodes increase, the Hits@1 score of the model on the validation set changes. An episode denotes that one of the sub-threads has completed training on a batch of data. As for PQL-mix, we observe that its Hits@1 score improves rapidly with an increase in training episodes, owing to the limited number of training instances it contains. When the value of λ is set to 1.0, the training process tends to be stable, and the Hits@1 score on the PQL-mix is significantly higher than the value of other hyperparameters. Therefore, the optimal value of λ should be set to 1.0. For MetaQA-2H, when λ is set to 0.5 and the training process stabilizes, it converges to the highest Hits@1 score.

4.10. Case study

As illustrated in Fig. 4, we utilize beam search on the KG to generate multiple reasoning paths at the testing stage, e.g., given the question "What is the film casting director of Cult Comedies's titles?", the topic entity is Cult Comedies. Starting from the topic entity Cult Comedies, through different linked relations, the agent reaches different entities, forming an initial set of action spaces, i.e., (self-loop, Cult Comedies), (genre_titles, Heathers), (genre_titles, Jackass_Number_Two) and (genre_titles, Cry-Baby). The policy network outputs the probabilities for each action in the action space, and then the top-N actions (N is the beam size and is set to 3) are selected based on their scores. Subsequently, the agent executes the top-N actions and reaches new entities, i.e., Heathers, Jackass_Number_Two,



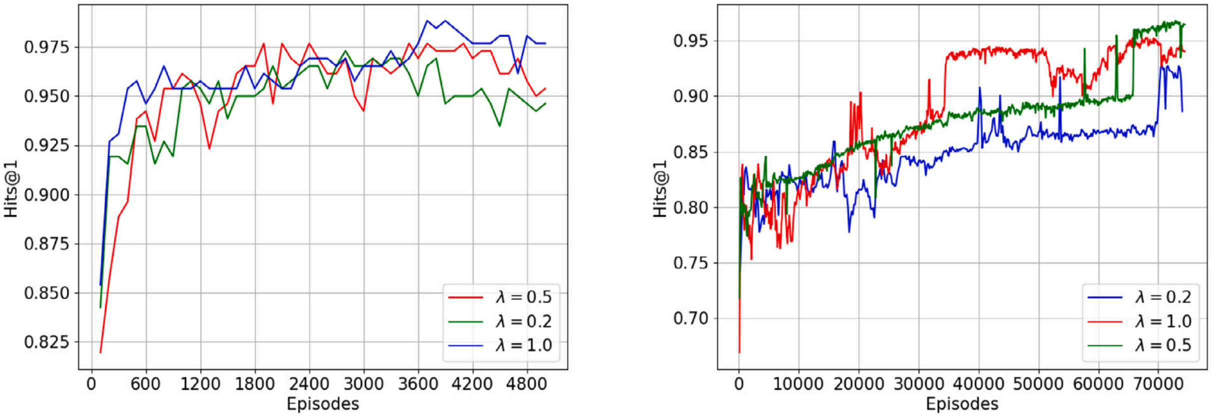
(a) PQ-3H

(b) PQ-mix

(c) PQL-mix

(d) MetaQA-3H

Fig. 2. The impact of multi-perspective information on the convergence rate of reinforcement learning, i.e., the change of validation Hits@1 score with the increase of training episodes.



(a) PQL-mix

(b) MetaQA-2H

Fig. 3. The impact of the hyper-parameter, i.e., the change of Hits@1 scores obtained by MPIFRN as the number of training episodes increases, with respect to different value loss coefficient λ values, evaluated on the validation set.

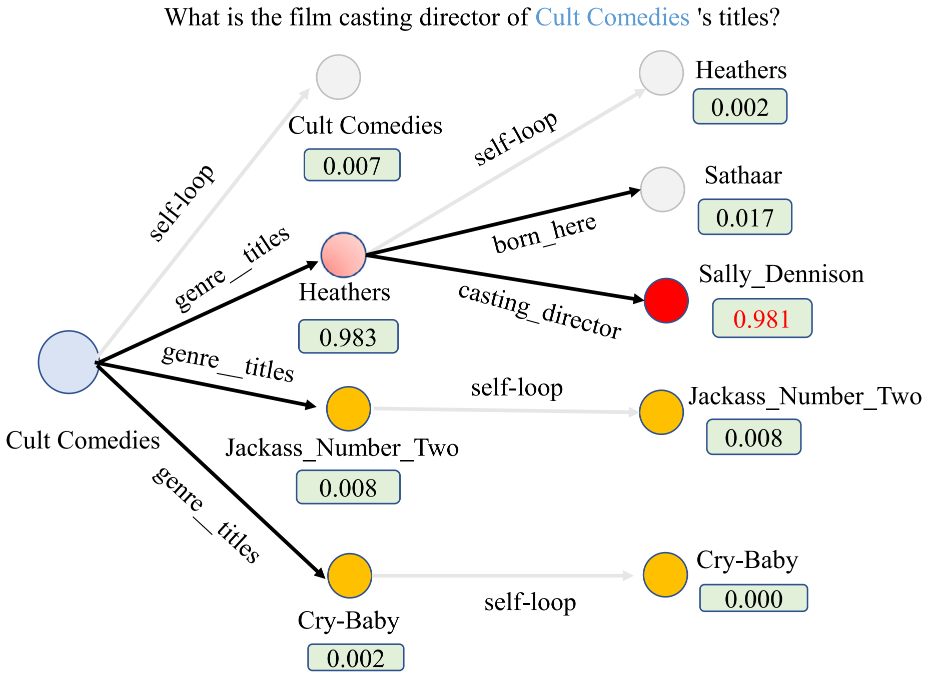


Fig. 4. A case from PQL-2H dataset.

and Cry-Baby. Similarly, in the second step, the current action space contains the relations derived from the top-N candidate entities in the previous step. The policy network again outputs the probability distribution of the actions and selects the top-N actions based on scores to execution. Finally, the tail entity with the highest score, Sally_Dennison, is considered as the predicted answer.

5. Conclusion and future work

In this work, we introduce Multi-Perspective Information Fusion Reasoning Network (MPIFRN), a new approach for multi-hop KGQA. Given that the agent in multi-hop KGQA based on RL cannot obtain sufficient information from the environment during the reasoning process, it hinders the agent's effective exploration of the state space to a certain extent and reduces the reliability of the strategy. Our MPIFRN model constructs three types of state information from different perspectives, i.e., expectation embedding, instruction-guided embedding, and path-aware embedding. These general multi-perspective pieces of information are model-agnostic and have been integrated into a carefully designed KEQA framework, which is combined with policy learning in A3C. Detailed experimental results show that our proposed MPIFRN outperforms most KGQA models in Hits@1 scores. In particular, the combination of multi-perspective information with A3C policy learning not only accelerates the convergence speed of reinforcement learning but also enhances the agent's efficiency in exploring critical paths, providing clearer interpretability of reasoning paths. At present, the KEQA framework and the policy selection module are trained independently, and future work includes studying the interaction between the KGQA framework and policy selection module, as well as joint training, to further enhance the performance of multi-hop KGQA.

CRediT authorship contribution statement

Chuanyang Gong: Conceptualization, Investigation, Methodology, Software, Validation, Writing – original draft, Writing – review & editing. **Zhihua Wei:** Project administration, Supervision, Conceptualization. **Rui Wang:** Supervision, Methodology, Validation. **Ping Zhu:** Investigation. **Jing Chen:** Investigation. **Hongyun Zhang:** Supervision. **Duoqian Miao:** Supervision.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

Acknowledgments

The work is partially supported by the National Nature Science Foundation of China (No. 61976160, 61906137, 61976158, 62076184, 62076182) and Shanghai Science and Technology Plan Project (No. 21DZ1204800) and Technology research plan project of the Ministry of Public and Security (Grant No. 2020JSYJD01).

References

Balazevic, I., Allen, C., & Hospedales, T. (2019). TuckER: Tensor factorization for knowledge graph completion. In *Proceedings of the 2019 conference on empirical methods in natural language processing and the 9th international joint conference on natural language processing* EMNLP-IJCNLP, (pp. 5185–5194). Association for Computational Linguistics, http://dx.doi.org/10.18653/v1/D19-1522, URL: https://aclanthology.org/D19-1522.

Bast, H., & Haussmann, E. (2015). More accurate question answering on freebase. In J. Bailey, A. Moffat, C. C. Aggarwal, M. de Rijke, R. Kumar, V. Murdock, T. K. Sellis, & J. X. Yu (Eds.), *Proceedings of the 24th ACM international conference on information and knowledge management, CIKM 2015, melbourne, VIC, Australia, October 19 - 23, 2015* (pp. 1431–1440). ACM, http://dx.doi.org/10.1145/2806416.2806472.

Berant, J., Chou, A., Frostig, R., & Liang, P. (2013). Semantic parsing on freebase from question-answer pairs. In *Proceedings of the 2013 conference on empirical methods in natural language processing* (pp. 1533–1544). Seattle, Washington, USA: Association for Computational Linguistics, URL: https://aclanthology.org/D13-1160.

Chen, Z.-Y., Chang, C.-H., Chen, Y.-P., Nayak, J., & Ku, L.-W. (2019). UHop: An unrestricted-hop relation extraction framework for knowledge-based question answering. In *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers)* (pp. 345–356). Minneapolis, Minnesota: Association for Computational Linguistics, http://dx.doi.org/10.18653/v1/N19-1031, URL: https://aclanthology.org/N19-1031.

Chen, Y., Li, H., Qi, G., Wu, T., & Wang, T. (2023). Outlining and filling: Hierarchical query graph generation for answering complex questions over knowledge graphs. *IEEE Transactions on Knowledge and Data Engineering*, *35*(8), 8343–8357. http://dx.doi.org/10.1109/TKDE.2022.3207477.

Christmann, P., Saha Roy, R., & Weikum, G. (2022). Beyond NED: Fast and effective search space reduction for complex question answering over knowledge bases. In *Proceedings of the fifteenth ACM international conference on web search and data mining* (pp. 172–180). http://dx.doi.org/10.1145/3488560.3498488.

Cui, H., Peng, T., Bao, T., Han, R., Han, J., & Liu, L. (2023). Stepwise relation prediction with dynamic reasoning network for multi-hop knowledge graph question answering. *Applied Intelligence: The International Journal of Artificial Intelligence, Neural Networks, and Complex Problem-Solving Technologies*, *53*(10), 12340–12354.

Cui, H., Peng, T., Feng, L., Bao, T., & Liu, L. (2021). Simple question answering over knowledge graph enhanced by question pattern classification. *Knowledge and Information Systems*, *63*(10), 2741–2761.

Cui, H., Peng, T., Xiao, F., Han, J., Han, R., & Liu, L. (2023). Incorporating anticipation embedding into reinforcement learning framework for multi-hop knowledge graph question answering. *Information Sciences*, *619*, 745–761. http://dx.doi.org/10.1016/j.ins.2022.11.042, URL: https://www.sciencedirect.com/science/article/pii/S0020025522013317.

Das, R., Dhuliawala, S., Zaheer, M., Vilnis, L., Durugkar, I., Krishnamurthy, A., Smola, A., & McCallum, A. (2018). 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, URL: https://openreview.net/forum?id=Syg-YfWCW.

Dettmers, T., Minervini, P., Stenetorp, P., & Riedel, S. (2018). Convolutional 2D knowledge graph embeddings. In *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. 1811–1818). AAAI Press, URL: https://www.aaai.org/ocs/index.php/AAAI/AAAI18/paper/view/17366.

Gardner, M., Talukdar, P. P., Kisiel, B., & Mitchell, T. (2013). Improving learning and inference in a large knowledge-base using latent syntactic cues. In *Proceedings of the 2013 conference on empirical methods in natural language processing* (pp. 833–838). Association for Computational Linguistics, URL: https://aclanthology.org/D13-1080.

Han, J., Cheng, B., & Wang, X. (2020). Two-phase hypergraph based reasoning with dynamic relations for multi-hop KBQA. In *Proceedings of the twenty-ninth international joint conference on artificial intelligence* IJCAI 2020, (pp. 3615–3621). ijcai.org, http://dx.doi.org/10.24963/ijcai.2020/500.

Hao, Y., Zhang, Y., Liu, K., He, S., Liu, Z., Wu, H., & Zhao, J. (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)* (pp. 221–231). Vancouver, Canada: Association for Computational Linguistics, http://dx.doi.org/10.18653/v1/P17-1021, URL: https://aclanthology.org/P17-1021.

He, G., Lan, Y., Jiang, J., Zhao, W. X., & Wen, J.-R. (2021). Improving multi-hop knowledge base question answering by learning intermediate supervision signals. In *Proceedings of the 14th ACM international conference on web search and data mining* (pp. 553–561).

Heo, Y.-J., Kim, E.-S., Choi, W. S., & Zhang, B.-T. (2022). Hypergraph transformer: Weakly-supervised multi-hop reasoning for knowledge-based visual question answering. In *Proceedings of the 60th annual meeting of the association for computational linguistics (volume 1: long papers)* (pp. 373–390). Dublin, Ireland: Association for Computational Linguistics, http://dx.doi.org/10.18653/v1/2022.acl-long.29, URL: https://aclanthology.org/2022.acl-long.29.

Jin, W., Zhao, B., Yu, H., Tao, X., Yin, R., & Liu, G. (2023). Improving embedded knowledge graph multi-hop question answering by introducing relational chain reasoning. *Data Mining and Knowledge Discovery*, *37*(1), 255–288.

Kaiser, M., Saha Roy, R., & Weikum, G. (2021). Reinforcement learning from reformulations in conversational question answering over knowledge graphs. In *Proceedings of the 44th international ACM SIGIR conference on research and development in information retrieval* (pp. 459–469).

Lee, W.-K., Shin, W.-C., Jagvaral, B., Roh, J.-S., Kim, M.-S., Lee, M.-H., Park, H.-K., & Park, Y.-T. (2021). A path-based relation networks model for knowledge graph completion. *Expert Systems with Applications*, *182*, Article 115273. http://dx.doi.org/10.1016/j.eswa.2021.115273, URL: https://www.sciencedirect.com/science/article/pii/S0957417421007041.

Lin, X. V., Socher, R., & Xiong, C. (2018). Multi-hop knowledge graph reasoning with reward shaping. In *Proceedings of the 2018 conference on empirical methods in natural language processing* (pp. 3243–3253). Association for Computational Linguistics, http://dx.doi.org/10.18653/v1/D18-1362, URL: https://aclanthology.org/D18-1362.

Lv, X., Han, X., Hou, L., Li, J., Liu, Z., Zhang, W., Zhang, Y., Kong, H., & Wu, S. (2020). Dynamic anticipation and completion for multi-hop reasoning over sparse knowledge graph. In *Proceedings of the 2020 conference on empirical methods in natural language processing* EMNLP, (pp. 5694–5703). Online: Association for Computational Linguistics, http://dx.doi.org/10.18653/v1/2020.emnlp-main.459, URL: https://aclanthology.org/2020.emnlp-main.459.

Miller, A., Fisch, A., Dodge, J., Karimi, A.-H., Bordes, A., & Weston, J. (2016). Key-value memory networks for directly reading documents. In *Proceedings of the 2016 conference on empirical methods in natural language processing* (pp. 1400–1409). Austin, Texas: Association for Computational Linguistics, http://dx.doi.org/10.18653/v1/D16-1147, URL: https://aclanthology.org/D16-1147.

Mnih, V., Badia, A. P., Mirza, M., Graves, A., Lillicrap, T. P., Harley, T., Silver, D., & Kavukcuoglu, K. (2016). Asynchronous methods for deep reinforcement learning. *vol. 48*, In *Proceedings of the 33nd international conference on machine learning, ICML 2016, new york city, NY, USA, June 19-24, 2016* (pp. 1928–1937). JMLR.org, URL: http://proceedings.mlr.press/v48/mniha16.html.

Niu, G., Li, Y., Tang, C., Hu, Z., Yang, S., Li, P., Wang, C., Wang, H., & Sun, J. (2021). Path-enhanced multi-relational question answering with knowledge graph embeddings. ArXiv preprint abs/2110.15622, URL: https://arxiv.org/abs/2110.15622.

Qin, K., Wang, Y., Li, C., Gunaratna, K., Jin, H., Pavlu, V., & Aslam, J. A. (2020). A complex kbqa system using multiple reasoning paths. ArXiv preprint abs/2005.10970, URL: https://arxiv.org/abs/2005.10970.

Qiu, Y., Wang, Y., Jin, X., & Zhang, K. (2020). Stepwise reasoning for multi-relation question answering over knowledge graph with weak supervision. In *WSDM '20: the thirteenth ACM international conference on web search and data mining, houston, TX, USA, February 3-7, 2020* (pp. 474–482). ACM, http://dx.doi.org/10.1145/3336191.3371812.

Saxena, A., Tripathi, A., & Talukdar, P. (2020). Improving multi-hop question answering over knowledge graphs using knowledge base embeddings. In *Proceedings of the 58th annual meeting of the association for computational linguistics* (pp. 4498–4507). Online: Association for Computational Linguistics, http://dx.doi.org/10.18653/v1/2020.acl-main.412, URL: https://aclanthology.org/2020.acl-main.412.

Shi, J., Cao, S., Hou, L., Li, J., & Zhang, H. (2021). 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* (pp. 4149–4158). Online and Punta Cana, Dominican Republic: Association for Computational Linguistics, http://dx.doi.org/10.18653/v1/2021.emnlp-main.341, URL: https://aclanthology.org/2021.emnlp-main.341.

Shin, S., & Lee, K. H. (2020). Processing knowledge graph-based complex questions through question decomposition and recomposition. *Information Sciences*, *523*, 234–244.

Sukhbaatar, S., Szlam, A., Weston, J., & Fergus, R. (2015). End-to-end memory networks. In *Advances in neural information processing systems 28: annual conference on neural information processing systems 2015, December 7-12, 2015, montreal, quebec, Canada* (pp. 2440–2448). URL: https://proceedings.neurips.cc/paper/2015/hash/8fb21ee7a2207526da55a679f0332de2-Abstract.html.

Sun, H., Bedrax-Weiss, T., & Cohen, W. (2019). PullNet: Open domain question answering with iterative retrieval on knowledge bases and text. In *Proceedings of the 2019 conference on empirical methods in natural language processing and the 9th international joint conference on natural language processing* EMNLP-IJCNLP, (pp. 2380–2390). Hong Kong, China: Association for Computational Linguistics, http://dx.doi.org/10.18653/v1/D19-1242, URL: https://aclanthology.org/D19-1242.

Sun, H., Dhingra, B., Zaheer, M., Mazaitis, K., Salakhutdinov, R., & Cohen, W. (2018). Open domain question answering using early fusion of knowledge bases and text. In *Proceedings of the 2018 conference on empirical methods in natural language processing* (pp. 4231–4242). Brussels, Belgium: Association for Computational Linguistics, http://dx.doi.org/10.18653/v1/D18-1455, URL: https://aclanthology.org/D18-1455.

Trouillon, T., Welbl, J., Riedel, S., Gaussier, É., & Bouchard, G. (2016). Complex embeddings for simple link prediction. *vol. 48*, In *Proceedings of the 33nd international conference on machine learning, ICML 2016, new york city, NY, USA, June 19-24, 2016* (pp. 2071–2080). PMLR, URL: http://proceedings.mlr.press/v48/trouillon16.html.

Vakulenko, S., Garcia, J. D. F., Polleres, A., de Rijke, M., & Cochez, M. (2019). Message passing for complex question answering over knowledge graphs. In *Proceedings of the 28th ACM international conference on information and knowledge management, CIKM 2019, Beijing, China, November 3-7, 2019* (pp. 1431–1440). ACM, http://dx.doi.org/10.1145/3357384.3358026.

Wu, W., Zhu, Z., Zhang, G., Kang, S., & Liu, P. (2021). A reasoning enhance network for muti-relation question answering. *Applied Intelligence: The International Journal of Artificial Intelligence, Neural Networks, and Complex Problem-Solving Technologies*, *51*(7), 4515–4524. http://dx.doi.org/10.1007/s10489-020-02111-6.

Xiong, W., Hoang, T., & Wang, W. Y. (2017). DeepPath: A reinforcement learning method for knowledge graph reasoning. In *Proceedings of the 2017 conference on empirical methods in natural language processing* (pp. 564–573). Association for Computational Linguistics, http://dx.doi.org/10.18653/v1/D17-1060, URL: https://aclanthology.org/D17-1060.

Xiong, W., Yu, M., Chang, S., Guo, X., & Wang, W. Y. (2019). Improving question answering over incomplete KBs with knowledge-aware reader. In *Proceedings of the 57th annual meeting of the association for computational linguistics* (pp. 4258–4264). Florence, Italy: Association for Computational Linguistics, http://dx.doi.org/10.18653/v1/P19-1417, URL: https://aclanthology.org/P19-1417.

Yang, M.-C., Lee, D.-G., Park, S.-Y., & Rim, H.-C. (2015). Knowledge-based question answering using the semantic embedding space. *Expert Systems with Applications*, *42*(23), 9086–9104. http://dx.doi.org/10.1016/j.eswa.2015.07.009, URL: https://www.sciencedirect.com/science/article/pii/S0957417415004698.

Yang, B., Yih, W., He, X., Gao, J., & Deng, L. (2015). Embedding entities and relations for learning and inference in knowledge bases. In *3rd international conference on learning representations, ICLR 2015, san diego, CA, USA, May 7-9, 2015, conference track proceedings*. URL: http://arxiv.org/abs/1412.6575.

Zhang, Y., Dai, H., Kozareva, Z., Smola, A. J., & Song, L. (2018). Variational reasoning for question answering with knowledge graph. In *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, URL: https://www.aaai.org/ocs/index.php/AAAI/AAAI18/paper/view/16983.

Zhou, M., Huang, M., & Zhu, X. (2018). An interpretable reasoning network for multi-relation question answering. In *Proceedings of the 27th international conference on computational linguistics* (pp. 2010–2022). Association for Computational Linguistics, URL: https://aclanthology.org/C18-1171.

Zhou, G., Xie, Z., Yu, Z., & Huang, J. X. (2021). DFM: A parameter-shared deep fused model for knowledge base question answering. *Information Sciences*, *547*, 103–118.

Zhu, A., Ouyang, D., Liang, S., & Shao, J. (2022). Step by step: A hierarchical framework for multi-hop knowledge graph reasoning with reinforcement learning. *Knowledge-Based Systems*, *248*, Article 108843. http://dx.doi.org/10.1016/j.knosys.2022.108843, URL: https://www.sciencedirect.com/science/article/pii/S0950705122004026.

Zhihua Wei is currently a Professor at Tongji University. She received a Ph.D. degree pattern recognition and intelligent system from Tongji University in China, a Ph.D. degree in Information from Lyon2 University in France, and B.S. and M.S. degrees both in Computer Science from Tongji University in China. Her cur- rent research interests include machine learning, natural language processing and speech processing. she is a member of the granular computing and knowledge discovery Professional Committee of Chinese Artificial Intelligence Society, and a member of the natural understanding professional committee of Chinese Artificial Intelligence Society.