electronics

MDPI

Article

PMHR: Path-Based Multi-Hop Reasoning Incorporating Rule-Enhanced Reinforcement Learning and KG Embeddings

Ang Ma [1], Yanhua Yu [1,*], Chuan Shi [1], Shuai Zhen [1], Liang Pang [2] and Tat-Seng Chua [3]

[1] College of Computer Science, Beijing University of Posts and Telecommunications, Beijing 100876, China; ang@bupt.edu.cn (A.M.); shichuan@bupt.edu.cn (C.S.); zhenshuai@bupt.edu.cn (S.Z.)
[2] Institute of Computing Technology, University of Chinese Academy of Sciences, Beijing 100190, China; pangliang@ict.ac.cn
[3] Sea-NExT Joint Lab, National University of Singapore, Singapore 119077, Singapore; dcscts@nus.edu.sg
* Correspondence: yuyanhua@bupt.edu.cn

Abstract: Multi-hop reasoning provides a means for inferring indirect relationships and missing information from knowledge graphs (KGs). Reinforcement learning (RL) was recently employed for multi-hop reasoning. Although RL-based methods provide explainability, they face challenges such as sparse rewards, spurious paths, large action spaces, and long training and running times. In this study, we present a novel approach that combines KG embeddings and RL strategies for multi-hop reasoning called path-based multi-hop reasoning (PMHR). We address the issues of sparse rewards and spurious paths by incorporating a well-designed reward function that combines soft rewards with rule-based rewards. The rewards are adjusted based on the target entity and the path to it. Furthermore, we perform action filtering and utilize the vectors of entities and relations acquired through KG embeddings to initialize the environment, thereby significantly reducing the runtime. Experiments involving a comprehensive performance evaluation, efficiency analysis, ablation studies, and a case study were performed. The experimental results on benchmark datasets demonstrate the effectiveness of PMHR in improving KG reasoning accuracy while preserving interpretability. Compared to existing state-of-the-art models, PMHR achieved Hit@1 improvements of 0.63%, 2.02%, and 3.17% on the UMLS, Kinship, and NELL-995 datasets, respectively. PMHR provides not only improved reasoning accuracy and explainability but also optimized computational efficiency, thereby offering a robust solution for multi-hop reasoning.

Keywords: knowledge graphs; knowledge graph reasoning; reinforcement learning; multi-hop reasoning

Citation: Ma, A.; Yu, Y.; Shi, C.; Zhen, S.; Pang, L.; Chua, T.-S. PMHR: Path-Based Multi-Hop Reasoning Incorporating Rule-Enhanced Reinforcement Learning and KG Embeddings. *Electronics* **2024**, *13*, 4847. https://doi.org/10.3390/electronics13234847

Academic Editor: Duc Tai Le

Received: 16 October 2024
Revised: 26 November 2024
Accepted: 4 December 2024
Published: 9 December 2024

1. Introduction

Knowledge graphs (KGs) [1] are powerful representations of structured information that capture complex semantic relationships between entities. They have become fundamental components in various applications such as natural language understanding [2,3], recommendation systems [4], and semantic web searches [5]. However, most extant KGs suffer from incompleteness and inaccuracy because they are usually built automatically from a variety of data sources. Incompleteness and inaccuracy are manifested as a significant dearth of crucial relations between the entities, which significantly constrains the applicability and utility of the KGs. Knowledge graph reasoning (KGR) involves inferring new knowledge or relations within a KG by applying various computational methods and techniques. The aim in KGR is to enhance the completeness, accuracy, and comprehensibility of KGs by revealing hidden information and predicting missing links.

Traditional KGR methods, particularly embedding-based and rule-based methods, have limitations in dealing with noisy or incomplete data and capturing the potential complexity of real-world knowledge. Multi-hop KGR [6–8] is an important KGR technique used to discover complex and multi-level relations between entities in a KG to infer

hidden information. Numerous reinforcement learning (RL)-based methods for multi-hop reasoning have emerged recently. Reinforcement methods were first employed to learn reasoning paths in KGs in DeepPath [9], where path finding is treated as a Markov decision process (MDP) and a sophisticated reward function is employed in which accuracy, efficiency, and path diversity are considered simultaneously. This approach provides enhanced control over the path finding process. DeepPath was followed by series of other techniques such as MINERVA [10], AttenPath [11], Reasoning Like Human (RLH) [12], DAPath [13], etc. These techniques offer high efficacy and the valuable characteristic of providing interpretable reasoning paths for KGR. For example, as shown in Figure 1, given a query (Albert Einstein, Nationality, ?), these methods can provide a chain of evidence to draw the conclusion (Albert Einstein, Nationality, Germany), e.g., Albert Einstein $\xrightarrow{\text{Born in}}$ Ulm $\xrightarrow{\text{Located in}}$ Baden-Württemberg $\xrightarrow{\text{Located in}}$ Germany.

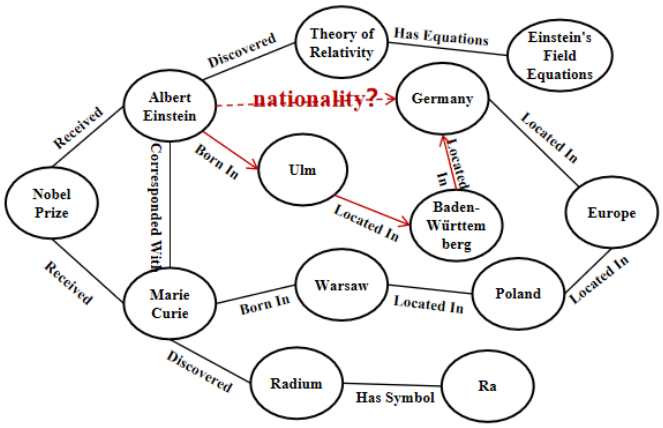


Figure 1. An example of a KG. Solid edges are observed and the dashed edge is a query.

However, the existing RL-based multi-hop reasoning methods have certain drawbacks. First, multi-hop reasoning suffers from the problem of sparse rewards, in which no feedback is provided for all the actions during the decision making process except for the final action. Although the problem of sparse rewards can be mitigated by using soft rewards, the majority of RL-based reasoning algorithms still face the problem of spurious paths, which significantly undermines the explainability of RL-based models. Second, the state representation of the agent plays a pivotal role in its ability to make informed decisions and learn effective policies. The state space should contain all the useful variables and information available to the agent in the environment. Third, the efficiency of RL algorithms is a major concern, as most RL algorithms require 10–20 h or more to complete training [14].

In this paper, we present an innovative framework that combines rule-enhanced RL and embeddings for multi-hop reasoning (path based multi-hop reasoning, PMHR). We address how reinforcement learning can effectively combine with embeddings and rules to perform accurate, interpretable multi-hop reasoning. We further explore how this approach leverages both learned representations and logical rules to enhance reasoning efficiency and scalability across complex knowledge graphs. Specifically, PMHR employs the REINFORCE algorithm. To address the issue of sparse rewards and spurious paths, PMHR combines soft rewards and rule-based rewards, which alleviates the problem of sparse rewards and prompts the agent to discover more reasonable paths. The expressive state space provides the agent with a comprehensive perspective of its environment. To enrich the state representation and provide more guided states for the agent, the state representation is redefined to include not only the immediate inputs but also relevant historical information, graph structure information and query-related guiding information. In addition, PMHR leverages the embedding of entities and relations learned through KG representation learning to initialize the environment and prune the action space, thereby improving the efficiency of RL and reducing the run time.

The key contributions of our work are as follows:

- We propose the PMHR model for KGs in which soft rewards and rule-based rewards are combined in a novel reward function. The use of soft rewards mitigates the sparse rewards problem, whereas rule-based rewards reshaping further addresses the problem of spurious paths.
- We redefine the state representation to provide an informative state, which equips the agent with the necessary tools to make effective decisions. The rich states allow the agent to recognize complex patterns, dependencies, and correlations within the environment. This facilitates more accurate predictions of the consequences of various actions.
- We perform action filtering for large action spaces by removing irrelevant actions. The efficiency of RL is enhanced by leveraging the embeddings of entities and relations learned from the KG representation for environment initialization.

2. Related Works

In this section, we present relevant works pertaining to the study. In addition, we provide a brief comparison of PMHR with previous models to highlight both their differences and connections.

2.1. Embedding-Based Knowledge Graph Reasoning

Embedding-based KGR aims to represent entities and relations in a KG as embeddings in a low-dimensional continuous vector space to perform various reasoning tasks, such as triple classification, relation prediction, and entity prediction. The early models such as TransE [15] and TransD [16] primarily focused on distance-based methods. In bilinear models such as RESCAL [17], DisMult [18], ComplEx [19], it is assumed that there are latent semantics between entities and relations and that triples can be measured by matching the latent semantic spatial similarity between entities and relations. In neural network-based approaches, such as the neural tensor network (NTN) [20], ConvE [21], and ConvKB [22], entities and relations are fed into convolutional neural networks to compute a semantic matching scores. The use of large language models for KGR has also been explored recently. The large language models, which utilize extensive textual information, function as encoders of entities and relations for triplet score calculations (e.g., PKGC [23], SimKGC [24]), generators for the direct generation of tail entities (e.g., GenKGC [25], AutoKG [26]), or rankers for reordering entities returned by traditional embedding-based models (e.g., KICGPT [27]). Despite the powerful representational ability of embedding-based models, their performance is limited by the complexity of multi-hop reasoning because of their difficulty in capturing long-distance or highly complex paths. In addition, it can be challenging for these models to explain their reasoning process, which can be problematic in situations where transparency and interpretability are essential.

2.2. Graph-Based Knowledge Graph Reasoning

Graph neural networks [28] have been widely used in recent years. They aggregate and propagate information by performing convolution on graph structures and perform well in many downstream tasks, such as node classification, graph classification, and link prediction. Graph-based KGR models typically follow the encoder–decoder structure. RGCN [29] is a graph convolutional network (GCN) model for multi-relational graphs. It acts as a graph encoder in which relationship-specific weight matrices are used to model different relationships and convolution operations are used to learn the embeddings of entities. SACN [30] consists of a WGCN encoder [31] and a Conv-TransE decoder. The weight is defined as the strength between two adjacent nodes with the same relation type, and the information in the KG is captured using the node structure, node attributes, and relation types. In order to maintain translation features, ConvE is employed in the Conv-TransE decoder as the semantic matching metric and the reshape step in ConvE 2D convolution is skipped. Differing from the above works, CompGCN [32] is a novel

framework for GCNs that simultaneously learns the embeddings of nodes and relations in multi-relational graphs while maintaining effective scalability with the number of relations. Graph-based methods consider the structural information of the graph and, through techniques like graph convolution or graph neural networks, leverage global information to improve reasoning performance. While they capture global structural information well, they often lack interpretability and may miss the specific path information crucial for transparent reasoning.

2.3. Path-Based Knowledge Graph Reasoning

The ability of the path-finding process between entities to be naturally modeled as a sequential decision problem has resulted in the emergence of RL-based multi-hop reasoning methods. DeepPath [9] is the first model in which RL is applied for relational reasoning by using the REINFORCE algorithm to find paths between entities. In MINERVA [10], the query answer is formulated as an RL problem for determining the path that provides an answer based on the input query. Strategies for navigating from the query entity to the answer node are learned by selecting relationships. MINERVA provides interpretable reasoning paths for its predictions. In MultiHopKG [33], pretrained representation learning models such as ConvE are utilized to provide soft rewards for unobserved triples, thereby mitigating the impact of sparse rewards. In AttnPath [11], LSTM and graph-attention mechanism are used as memory components. Unlike the aforementioned methods in which the rewards are manually designed, the reward functions for different datasets in DIVINE [34] are automatically tuned for optimal performance. In GaussianPath [35], a Gaussian distribution is employed to represent entities and relations and to capture reasoning path uncertainty. In the PAAR model [36], multi-hop reasoning is enhanced by leveraging hyperbolic embeddings to capture hierarchical information and using soft rewards to address reward sparsity. A novel reward function was introduced in Path Spuriousness (PS) [14] to estimate the extent of path spuriousness and balance answer accuracy with path reasonableness.

2.4. Differences

Our approach differs significantly from those in previous studies in several key aspects. First, the existing RL-based multi-hop reasoning methods are hindered by the problem of sparse rewards, in which feedback provided only for the final action in the decision making process. Although this issue is partially alleviated by soft rewards, many RL algorithms continue to face the challenge of spurious paths, which undermines the explainability of RL-based models. Our approach combines soft rewards and rule-based rewards to address both the sparse rewards and spurious paths problems effectively by prompting the agent to discover more reasonable paths. Second, in contrast to the often limited state representation in current algorithms, the importance of comprehensive state representation is emphasized in our approach. We redefine the state representation to include not only the immediate inputs but also the path history, graph structure, and query-related guidance information. This enriched state space equips the agent with a more detailed understanding of its environment to enable better decision making. Third, the efficiency of RL algorithms is a significant concern because most RL algorithms require extensive training. Our framework leverages the embeddings of entities and relations learned through KG representation learning to initialize the environment and prune the action space, thereby significantly improving the efficiency and reducing the run time. In summary, our innovative framework addresses key limitations of existing RL-based methods by effectively handling sparse rewards and spurious paths, enhancing state representation, and improving algorithm efficiency.

3. Preliminaries

We begin this section by presenting the definitions of the key concepts. We subsequently detail the fundamentals of our model by introducing the RL formula.

3.1. Path Based Multi-Hop Reasoning

Knowledge graph: A KG is represented as a directed graph $G = (E, R)$, where E denotes the set of entities and R represents the set of relations. A fact within the knowledge graph G is defined by a triple (e_s, r, e_o), where e_s and e_o are entities belonging to E, and r is a relation in R. The edges within the KG are bidirectional. This enables the graph to be further enriched through the addition of reverse edges.

Path-based Multi-Hop Reasoning: Given query $q = (e_s, r_q, ?)$ and a KG, the tail entity is predicted in path-based multi-hop reasoning by performing effective path searches and discovering one or multiple l-hop paths on the KG (e.g., $e_s \xrightarrow{r_1} e_1 \xrightarrow{r_2} \cdots \xrightarrow{r_l} e_T$) that lead to a potential set of target entities. We let P be a set of paths, where each path p comprises a start entity e_s and passes through a series of relations to reach the target entity e_{T_p}. The set of target entities can be represented as $E_{\text{target}} = \left\{ e_{T_p} \mid p \in P \right\}$, where $\left(e_s, r_q, e_{T_p}\right) \notin G$. The potential reasoning paths serve to elucidate the reasoning process and enhance its interpretability.

For example, we consider the query (Albert Einstein, nationality, ?) shown in Figure 1. One possible three-hop reasoning path is Albert Einstein $\xrightarrow{\text{Born in}}$ Ulm $\xrightarrow{\text{Located in}}$ Baden-Württemberg $\xrightarrow{\text{Located in}}$ Germany.

3.2. RL Formula

The reasoning process can be modeled as an MDP with an agent and a KG environment. The agent selects actions according to the policy network, whereas the KG environment provides the states and rewards. The primary components in MDP are as follows:

States: A state concatenates the current situation with the original query. The state in Step t is represented as a tuple $s_t = \left(e_t, r_q, h_t, d_q\right)$, where e_t denotes the current entity, r_q denotes the query relation, h_t denotes the decision history information, and d_q is the query-related guiding information. The state is thoroughly explained in Section 4.3.

Actions: The possible actions at Step t comprise the outgoing edges at the current entity e_t and their associated entities. The action space denotes $A_t = \{(r_{t+1}, e_{t+1}) \mid (e_t, r_{t+1}, e_{t+1}) \in G\}$. The valid actions therefore represent the neighborhoods of the current entity e_t. In line with previous studies, we augment the action space by introducing the self-loop relations r_{sl}. Specifically, at Step t, we add the action (r_{sl}, e_t) to the action space A_t.

Transitions: The agent performs action $a_t = (r_{t+1}, e_{t+1})$, which converts the current state s_t into s_{t+1}. Formally, this transition is denoted as $s_{t+1} = \delta(s_t, A_t) = \left(e_{t+1}, r_q, h_{t+1}, d_q\right)$. Note that r_q and d_q represent global information that remains unchanged during the transition.

Rewards: In a KG environment, rewards provide feedback to the agent based on its actions. Typically, the default binary reward is a terminal reward [33], defined as $R_T(e_T) = \mathbb{I}\left(\left(e_s, r_q, e_T\right) \in G\right)$, where $\mathbb{I}(\cdot)$ is an indicator function. If the agent succeeds in locating the target entity e_T at the final step T, i.e., the triple $\left(e_s, r_q, e_T\right)$ exists in the KG, the indicator function returns one; otherwise, it returns zero. However, relying solely on terminal rewards often leads to sparse reward scenarios. Even if the path reaches a correct entity, it can be purely coincidental and lack logical relevance to the query. To address this problem, we propose a novel reward function, which is described in detail in Section 4.3.

4. Methodology

In this section, the framework of the PMHR model is described. Following this, we detail the KG representation learning approach for pre-training the entities and relations in the KG and obtaining their representations. The query policy network, which incorporates state representation and action space pruning, is subsequently introduced. Finally, a mixed rules and embedded reward function is proposed.

4.1. Overview

Figure 2 shows an overview of PMHR. PMHR consists of five parts: KGE pre-training, state representation, policy network, action space pruning, and rule-based reward reshaping. Specifically, a KG embedding technique such as ConvE is first utilized for pre-training to obtain representations of the entities and relations in the KG, which are then used to initialize the environment for RL. The state representation is subsequently enriched to better guide the agent in learning and the action space is pruned using the obtained representation to obtain the available candidate actions. Finally, a reward function that combines binary rewards, soft rewards, and rule-based rewards is used to guide the agent in finding a correct and reasonable path.

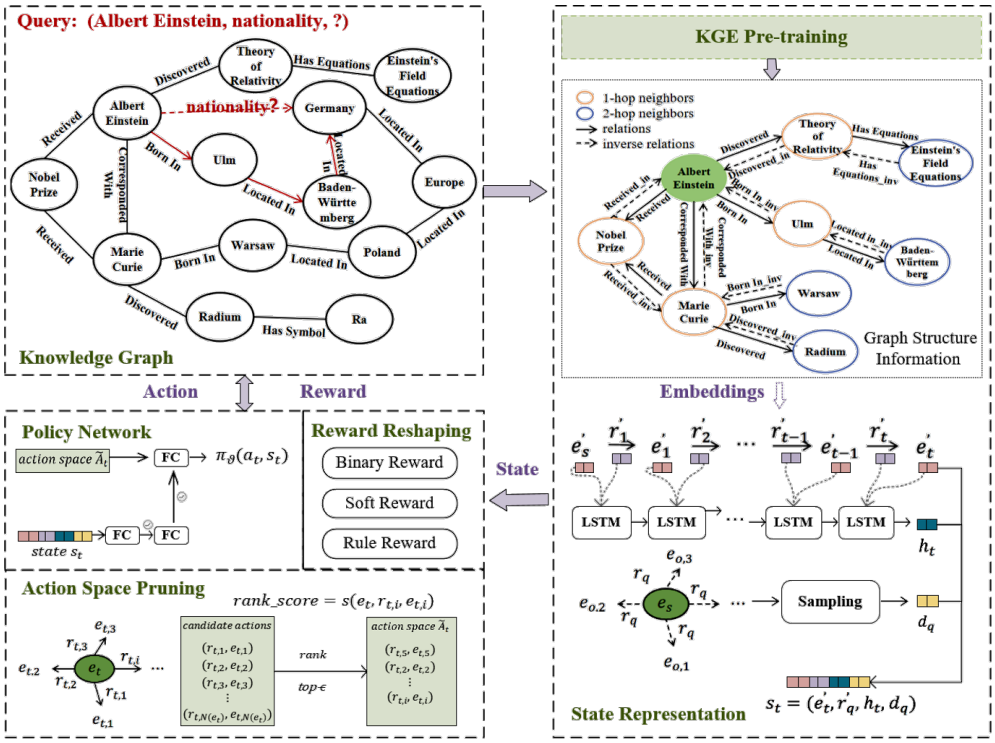


Figure 2. An overview of PMHR model.

At each step, the agent evaluates the current state along with the available candidate actions to determine the move to the next state. This process continues iteratively until the agent reaches the target entity or the maximum exploration length is reached. The exploration trajectory of the agent forms a reasoning path, which is used not only to infer new knowledge but also to provide an explanatory foundation for the reasoning results. The reasoning path aids in elucidating and validating the reasoning results, thereby enhancing their credibility and practical utility.

4.2. KGE Pre-Training

Despite their lack of interpretability, knowledge graph embedding (KGE) models often outperform multi-hop reasoning models across many KGs. KGE models are more resilient due to their independence from KG connectivity [37].

Motivated by this observation, we utilize ConvE [21], a classical KGE method, to perform representation learning of the entities and relations in the KG. The first step in the method involves reshaping and concatenating the entity and relation embeddings. The combined matrix is then fed into a convolutional layer and the resulting feature map tensor vectorized and projected into a k-dimensional space. This projected representation is finally

compared with all the candidate object embeddings to obtain the scores of the triples. The score of a triple (e_s, r, e_o) is formally calculated as

$$\psi_r(e_s, e_o) = f(\text{vec}(f([\overline{\mathbf{e}_s}; \overline{\mathbf{r}}] * \omega))\mathbf{W}_1)\mathbf{e}_o, \tag{1}$$

where $\mathbf{e}_s$ and $\mathbf{e}_o \in \mathbb{R}^d$ are the embeddings of entities e_s and e_o, respectively. $\mathbf{r} \in \mathbb{R}^d$ is the embedding of the relation r. $\overline{\mathbf{e}_s}$ and $\overline{\mathbf{r}}$ denote a 2D reshaping of $\mathbf{e}_s$ and $\mathbf{r}$, and $\overline{\mathbf{e}_s}, \overline{\mathbf{r}} \in \mathbb{R}^{d_w \times d_h}$, where $d = d_w d_h$. $\mathbf{W_1}$ is a learnable weight matrix. $*$ represents the convolution operation. Here, the concatenated $[\overline{\mathbf{e}_s}; \overline{\mathbf{r}}]$ is used as an input to a 2D convolutional layer with filters ω. $\text{vec}(\cdot)$ is a vectorization operation that transforms the feature map tensor produced by the convolutional layer into a 1D vector. f denotes the rectified linear unit (ReLU) activation function.

The embeddings of the entities and relations in G are obtained by training with binary cross-entropy loss. These embeddings from the KGE stage are subsequently used as the initial embeddings in the RL stage.

4.3. State Representation

The state representation quality has a significant impact on the RL performance of the agent. A well-defined state representation can capture essential features of the environment, thereby enabling the agent to make more informed and effective decisions. In this section, we explain the various types of information for refining the state representation, comprising the graph structure information, path history information, and query-related guiding information. By improving the state representation, we aim to enhance the agent's ability to understand and navigate complex environments, which ultimately leads to better learning outcomes and more robust policies.

Graph structure information: We explicitly encode the local structure information of the graph to enrich the state representation. Inspired by CompGCN [32], we use GCNs to aggregate the neighbor information of the entities. To be specific, let us denote the current entity as e_t, where $\mathcal{N}(e_t)$ represents the set of neighbors of e_t along its outgoing edges. The representation of e_t after k layers, denoted as $\mathbf{e}_t^{k+1}$, is defined as follows:

$$\mathbf{e}_t^{(k+1)} = f\left(\sum_{(r_{t+1}, e_{t+1}) \in \mathcal{N}(e_t)} \mathbf{W_2}\, \phi\left(\mathbf{r}_{t+1}^k, \mathbf{e}_{t+1}^k\right)\right), \tag{2}$$

where $\mathbf{W}_2 \in \mathbb{R}^{d_o \times d}$ is a learnable weight matrix, ϕ is a non-parametric composition function, such as concatenation $\phi(\mathbf{r}, \mathbf{e}) = [\mathbf{r}; \mathbf{e}]$, addition $\phi(\mathbf{r}, \mathbf{e}) = \mathbf{r} + \mathbf{e}$, or multiplication $\phi(\mathbf{r}, \mathbf{e}) = \mathbf{r} * \mathbf{e}$. The representation of a relation r_t after k layers, denoted as $\mathbf{r}_t^{k+1}$, is defined as follows:

$$\mathbf{r}_t^{k+1} = \mathbf{W}_{\text{rel}}^k \mathbf{r}_t^k, \tag{3}$$

where $\mathbf{W}_{\text{rel}} \in \mathbb{R}^{d \times d}$ is a learnable transformation matrix that maps relations into the entity embedding space so that they can be used in the subsequent GCN layer. We use the entity and relation representations after the kth layer as the final entity and relation representations, which we denote as $\mathbf{e}'$ and $\mathbf{r}'$, respectively.

Path history information: We employ a long short-term memory (LSTM) network to encode the path history information h_t at Step t so that the agent can memorize the path history [14,33]. The LSTM receives the previous history h_{t-1} and the last action $a_t = (r_{t+1}, e_{t+1})$ as input in the following way:

$$\mathbf{h}_t = \text{LSTM}(\mathbf{h}_{t-1}, \mathbf{a}_t). \tag{4}$$

where $\mathbf{a}_t = [\mathbf{r}'_{t+1}; \mathbf{e}'_{t+1}]$.

Query-related guiding information: Given that KGE models outperform multi-hop reasoning models across many KGs, we integrate guiding information from embedding-based models into the state representation to facilitate model learning. Specifically, for the

query $(e_s, r_q, ?)$, we utilize the KGE model in Section 4.2 and derive a probability vector denoted as $\mathbf{q} \in \mathbb{R}^{|E|}$. Each dimension i of $\mathbf{q}$ denotes the probability that e_i corresponds to the correct tail entity. We then sample an entity e_i according to the probability distribution $\mathbf{q}$ and denote its vector as $\mathbf{d}_q = \mathbf{e}'_i$.

State representation refinement: After learning the representations of the graph structure information, path history information, and query-related guiding information in the previous stage, the refined state representation is obtained as

$$\mathbf{s}_t = [\mathbf{e}'_t; \mathbf{r}'_q; \mathbf{h}_t; \mathbf{d}_q] \tag{5}$$

where $[;]$ denotes the concatenation operation.

4.4. Policy Network

The parameterized policy network receives the state information and outputs the probability distribution π of the action space.

Action space pruning: The pre-trained KGE model is used to predict the probability of the actions $a_t = (r_{t+1}, e_{t+1})$ for the current entity e_t via the triple scores $\psi_{r_{t+1}}(e_t, e_{t+1})$. To simplify the presentation, we let $A_t = \{a_1, a_2, \ldots, a_m\}$ be the action space at time t, which contains m actions. These actions are sorted based on the values of the scoring function ψ. The sorting can be represented by a permutation ρ such that $f\left(a_{\rho(1)}\right) \geq f\left(a_{\rho(2)}\right) \geq \cdots \geq f\left(a_{\rho(m)}\right)$. The top ϵ actions after sorting are retained as the available candidate actions, which can be represented as the set $\widetilde{A}_t$.

$$\widetilde{A}_t = \left\{a_{\rho(1)}, a_{\rho(2)}, \ldots, a_{\rho(\epsilon)}\right\}. \tag{6}$$

Policy network: Based on the modified state enriched with query-related guiding information, the policy network π_θ outputs the probability distribution $\pi(a_t \mid s_t)$ over all the possible actions in $\widetilde{A}_t$. This process can be expressed as

$$\pi_\theta(a_t \mid s_t) = \sigma\left(\widetilde{\mathbf{A}}_t(\mathbf{W}_3 \operatorname{ReLU}(\mathbf{W}_4 \mathbf{s}_t))\right), \tag{7}$$

where σ denotes the softmax operator, $\mathbf{W}_3$ and $\mathbf{W}_4$ are two learnable weight matrices, and $\widetilde{\mathbf{A}}_t \in \mathbb{R}^{\epsilon \times 2d}$ serves as a representation of the action space $\widetilde{A}_t$, formed by concatenating the embeddings of all the actions within it.

4.5. Rule-Based Reward Reshaping

To mitigate the incompleteness of KGs, the reward shaping function R_s was introduced in MultiHopKG [33]. Unlike R_T, which provides only a score of 0 when the answer entity is not present in the KG, R_s provides a score between 0 and 1 as a soft reward in such cases. The score is computed using the pre-trained embedding-based model described in Section 4.2. $R_S(e_T)$ is defined as

$$R_S(e_T) = R_T(e_T) + (1 - R_T(e_T)) \cdot \psi_{r_q}(e_s, e_o). \tag{8}$$

If the answer entity e_T corresponds to a triple (e_s, r_q, e_T) in the KG, and the agent receives a reward, $R_S(e_T) = R_T(e_T) = 1$. If (e_s, r_q, e_T) is not present in the KG, the agent receives a reward $R_S(e_T) = \psi_{r_q}(e_s, e_o)$ computed by the pre-trained embedding-based model. This approach enables the agent to receive informative feedback even when the ground truth is not directly available in the KG.

Although this reward design extends beyond traditional binary rewards and incorporates soft rewards, it does not completely address the issue of spurious paths. Spurious paths can still occur when the agent stumbles upon the correct answer by chance. These paths lack interpretability and fail to provide meaningful insights into the underlying reasoning process.

To mitigate the spurious paths problem and enhance interpretability, additional mechanisms may need to be incorporated into the model. We utilize the KG association rule mining algorithm AnyBURL [38], which is a algorithm for knowledge graph completion within the symbolic space by sampling paths and generalizing them into Horn rules.

The rules mined by AnyBURL are typically Horn clauses in the form of implications. A rule in AnyBURL has the general form of

$$\text{Body} \Rightarrow \text{Head}, \tag{9}$$

where Head is the consequent of the rule, which is typically a single triple pattern, and Body is the antecedent of the rule, which is typically a conjunction of triple patterns. Each rule denoted as H mined by AnyBURL comes with an associated confidence $conf(H)$. The confidence represents the fraction of instances where both the body and head of the rule hold true. We consider a rule that states 'If John is born in Paris, and Paris is located in France, then John's nationality is France'.

$$H_1 : (X, \text{ Born in }, Y) \wedge (Y, \text{ Located in }, Z) \Rightarrow (X, \text{ Nationality}, Z). \tag{10}$$

Here, X, Y, and Z are variables. The resulting path is converted into the form of a Horn rule. For example, the corresponding Horn rule for the path Albert Einstein $\xrightarrow{\text{Born in}}$ Ulm $\xrightarrow{\text{Located in}}$ Baden-Wurttemberg $\xrightarrow{\text{Located in}}$ Germany is

$$H_2 : (X, \text{ Born in }, F) \wedge (F, \text{ Located in }, Y) \wedge (Y, \text{ Located in }, Z) \Rightarrow (X, \text{ Nationality}, Z). \tag{11}$$

If the target entity is reached in a path p, we extract the Horn rule for the path, denoted as H_p, and use the confidence of the rule as the rule reward $R_R(H_p) = conf(H_p)$. In this paper, we use the confidence provided by AnyBURL as the rule reward. The final reward is formally given by

$$R(e_T) = R_T(e_T) \cdot \big(1 + R_R\big(H_p\big)\big) + (1 - R_T(e_T)) \cdot \psi_{r_q}(e_s, e_o). \tag{12}$$

If the answer entity e_T corresponds to a triple $\big(e_s, r_q, e_T\big)$ in the KG, i.e., $R_T(e_T) = 1$, the agent receives a reward, $R(e_T) = 1 + R_R\big(H_p\big)$. If $\big(e_s, r_q, e_T\big)$ is not present in the KG, i.e., $R_T(e_T) = 0$, the agent receives a reward $R(e_T) = \psi_{r_q}(e_s, e_o)$ computed by the pre-trained embedding-based model.

The rule reward scheme can help ensure that the paths generated by the model are logically relevant to the query triple. The interpretability of the model is therefore enhanced by considering the logical relevance of the reasoning paths to the queries during its decision making process.

4.6. Optimization

The policy network is trained to obtain the optimal parameters by maximizing the expected cumulative reward. The objective function can be expressed as

$$J(\theta) = \mathbb{E}_{\left(e_s, r_q, e_T\right) \in G} \mathbb{E}_{(a_1, a_2, \cdots, a_T) \sim \pi_\theta} R(e_T), \tag{13}$$

where $J(\theta)$ represents the expected cumulative reward for all the training data following policy π_θ. REINFORCE is utilized to address the optimization problem by iteratively updating θ as follows:

$$\nabla_\theta J(\theta) \approx \sum_{t=1}^{T} R(e_T) \nabla_\theta \log \pi_\theta(a_t \mid s_t), \tag{14}$$

$$\theta = \theta + \eta \nabla_\theta J(\theta), \tag{15}$$

where η is the learning rate. A detailed pseudocode of PMHR is provided in Algorithm 1. Assuming N is the number of episodes, the training set contains M triples, and T represents the maximum number of steps, the time complexity of PMHR simplifies to $O(N \cdot M \cdot T)$.

Algorithm 1: Training Algorithm of the proposed PMHR

Input: KG G, pre-trained embedding model, AnyBURL rules confidence scores, Training Set
Output: parameters of PMHR θ

```
initialize LSTM, Policy network; for episode ← 1 to N do
    for each triple (e_s, r_q, e_T) in Training Set do
        initialize num steps to 0; while num steps < max steps do
            compute the representation of current state with Equation (2),
             Equation (3), Equation (4) and Equation (5); compute the action space
             with Equation (6); compute π_θ(a_t | s_t) with Equation (7) and sample;
             Update the state and increment num steps; if success or num steps =
             max steps then
                break;
            end
        end
        compute the reward with Equation (12); update the parameters with
         Equation (13), Equation (14) and Equation (15);
    end
end
```

5. Experiment

We assess the performance of PMHR for knowledge reasoning tasks using four KG datasets. We first outline the experimental datasets, baseline models, evaluation metrics, and implementation specifics in the experimental setup. We then conduct a comparative analysis between our model and the baseline models. This is followed by an ablation study aimed at elucidating the effectiveness of the PMHR components. Finally, we present a case study to demonstrate the resulting interpretable reasoning paths.

5.1. Datasets

We utilized four open-access datasets commonly used in KG reasoning comprising UMLS [39], Kinship [33], WN18RR [21], and NELL-995 [9]. UMLS [39] is a comprehensive knowledge base that incorporates general knowledge about various health and biomedical vocabularies. Kinship [33] contains entities and relations representing kinship relationships between family members. WN18RR [21] is a relational dataset derived from WordNet. NELL-995 [9] was obtained from NELL (Never-Ending Language Learning), which is a knowledge base extracted from Web texts for machine learning and natural language processing research. The four datasets contain various types of structural information. The statistics of the datasets are presented in Table 1.

Table 1. Dataset statistics.

Dataset	#Fact			#Ent	#Rel	#Degree	
	#Train	#Valid	#Test			Avg.	Median
UMLS	5216	652	661	135	48	38.6	28
Kinship	8544	1068	1074	104	25	82.2	82
WN18RR	86,835	3034	3134	40,943	11	2.2	2
NELL-995	8747	543	3992	10,105	12	1.6	1

5.2. Baseline Models and Evaluation

We conducted a comprehensive comparison of PMHR with several baseline models comprising embedding-based models, rule-based models, and path-based models.

- The embedding-based models comprise DistMult [18], ComplEx [19], and ConvE [21].
- The rule-based models comprise the following: Neural Logical Programming (NeuralLP) [40] is an end-to-end knowledge base reasoning framework inspired by TensorLog for learning probabilistic first-order logical rules. A neural controller is used in NeuralLP to compose differentiable reasoning operations. AnyBURL [38] is a model for efficiently mining logical rules on KGs.
- The path-based models comprise the following: MINERVA [10] is an automated reasoning model for multi-hop reasoning designed to teach an agent to navigate to the answer entity based on the query. MultiHopKG [33] incorporates embedding-based models as the reward shaping function. MetaKGR [41] uses meta-learning to derive meta parameters from high-frequency relations so that few-shot relations can be quickly adapted to and performance drops for rare relations mitigated. RARL [42] integrates symbolic rules and uses partially random beam search to mitigate sparse reward issues and reduce bias from spurious paths. PAAR [36] is a hyperbolic KGE model with an expanded action space in which KGE scores are utilized as soft rewards, and hyperbolic space metrics are employed as a penalty strategy to address reward sparsity. In Path Spuriousness (PS) [14], Path Spuriousness is introduced to calculate path quality in a model that balances reasoning accuracy and path reasonableness.

The baseline performance metrics shown in Table 2 were obtained directly from the respective studies. We assessed performance using the Hits@*k* and mean reciprocal rank (MRR) standard metrics. Hits@*k* represents the proportion of correct tail entities ranked in the top *k* positions, reflecting the model's ability to prioritize correct answers. MRR provides the average reciprocal rank of correct answers, indicating how highly the model ranks them overall.

Table 2. Performance comparison of PMHR with baseline models.

	UMLS			**Kinship**			**WN18RR**			**NELL-995**		
	Hit@1	**Hit@10**	**MRR**	**Hit@1**	**Hit@10**	**MRR**	**Hit@1**	**Hit@10**	**MRR**	**Hit@1**	**Hit@10**	**MRR**
DisMult	0.829	0.971	0.877	0.454	0.903	0.593	0.406	0.495	0.434	0.642	0.862	0.730
ComplEx	0.911	0.991	0.943	0.798	0.985	0.873	0.416	0.476	0.436	0.646	0.856	0.729
ConvE	0.926	0.997	0.954	0.815	0.984	0.882	0.398	0.539	0.446	0.673	0.879	0.755
NeuralLP	0.629	0.952	0.766	0.481	0.901	0.620	0.362	0.551	0.423	0.059	0.682	0.349
AnyBURL	0.905	0.986	0.944	0.524	0.922	0.662	0.448	0.558	0.484	0.360	0.545	0.427
MINERVA	0.702	0.924	0.803	0.432	0.920	0.600	0.431	0.528	0.462	0.638	0.825	0.706
MultiHopKG	0.903	0.992	0.940	0.789	0.982	0.866	0.408	0.511	0.442	0.647	0.840	0.720
MetaKGR	0.909	0.994	0.945	0.790	0.986	0.871	0.364	0.502	0.418	0.588	0.834	0.686
RARL	0.762	0.956	0.842	0.613	0.944	0.733	**0.442**	0.533	**0.469**	0.628	0.822	0.704
PAAR	0.895	0.995	0.940	0.727	0.963	0.814	0.406	**0.538**	0.449	0.652	0.839	0.722
Path Spuriousness	0.868	0.992	0.913	0.798	0.982	0.872	0.415	0.513	0.449	0.635	0.840	0.704
PMHR	**0.911**	0.994	**0.947**	**0.807**	**0.987**	**0.878**	0.426	0.525	0.457	**0.684**	**0.849**	**0.747**

Note: The best scores for path-based models are in bold, while those for embedding-based models are underscored.

5.3. Implementation Details

All the experiments were performed using Python 3.7, PyTorch 1.8.1, and the CUDA 11.1. We employed a grid search strategy to identify optimal hyperparameter values. We set the embedding sizes of entities and relations for the four datasets to 200 and utilized a 1-layer GCN as the graph structure encoder and a 3-layer LSTM as the path history encoder. The batch size was set to 512 for the WN18RR dataset, 128 for the Kinship and NELL-995 datasets, and 64 for the UMLS dataset. The embedding dropout rate was set to 0.1 for UMLS, WN18RR, and NELL-995, and to 0.3 for Kinship. The action dropout rate

was set to 0.95 for UMLS, 0.9 for Kinship, 0.1 for WN18RR, and NELL-995. In the KGE-pretrained ConvE model, convolutions were performed with a kernel size of 3, and the number of output channels in the convolution layer was set to 32. Adam optimization [43] was employed for parameter refinement. The learning rate was set to 0.001 for UMLS, Kinship, WN18RR, and 0.003 for NELL-995.

To determine the optimal number of hops for multi-hop reasoning, we conducted a statistical analysis of the dataset. Specifically, we utilized the triples in the training dataset to construct a graph and then analyzed the triples in the validation and test datasets. We employed the Breadth-First Search (BFS) algorithm to identify the shortest paths between the head and tail entities and recorded the distribution of the path lengths. Figure 3 shows the distribution of the shortest path lengths in the four datasets. The symbol ∞ indicates that there is no path between the head and tail entities, which means that they are unreachable from one another. As shown in Figure 3, the entities in the UMLS and Kinship datasets can be obtained within two hops, while most of entities in the WN18RR and NELL-995 datasets can be found within three hops. However, some entities in the WN18RR and NELL-995 datasets are unreachable. Therefore, the maximum reasoning hop was set to two for the UMLS and Kinship datasets and three for WN18RR and NELL-995. The values for PMHR in Table 2 were obtained as the averages of the results for three runs.

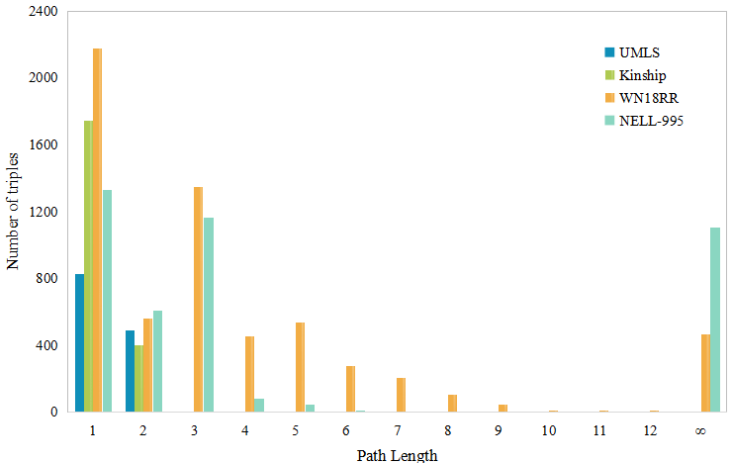


Figure 3. The distribution of the shortest path lengths.

5.4. Results and Analysis

Table 2 shows a comparison between the link prediction results of PMHR and the baseline models on UMLS, Kinship, WN18RR, and NELL-995. The top part shows embedding-based models, the second part shows rule-based models, the third part shows path-based multi-hop reasoning models, and the bottom part shows our proposed method PMHR. The results for these baseline models were obtained by re-running their publicly available open-source code with the original configurations. All experiments were conducted on the same testbed under identical conditions, and we report the mean performance across three independent runs to ensure reliability. For RARL [42] and PAAR [36], as their open-source code was unavailable, we directly cited the results reported in Path Spuriousness [14]. The best scores of the path-based multi-hop reasoning models are highlighted in bold, whereas those of the embedding-based models are underscored.

The performance evaluation metrics in Table 2 indicate that the KGE models outperformed the rule-based models, which exhibited the lowest performance. However, this study emphasizes path-based reasoning, which offers an advantage beyond accuracy: it enables interpretable reasoning paths—a crucial feature often lacking in embedding-based models, especially in applications where understanding the reasoning process is as important as the prediction itself. As shown in Table 2, our model consistently outperformed all the path-based baseline models on three datasets UMLS, Kinship, and NELL-995, and was second only to the state-of-the-art (SOTA) model on WN18RR. These results show

that the effectiveness of our model is significantly enhanced by RL through the integration of query-related guiding information into the representation vectors. Although adding a soft reward function helps alleviate the sparse rewards problem in RL-based multi-hop reasoning, it cannot easily resolve the issue of spurious paths. The problem of spurious paths is alleviated in PMHR through rule-based reward reshaping by further modifying the rewards for the paths that correctly reach the target entity.

5.5. Efficiency Analysis

The training time efficiency of our proposed method PMHR was compared against four baseline models (MINERVA [10], MultiHopKG [33], MetaKGR [41], and Path Spuriousness [14]) across four datasets (UMLS, Kinship, WN18RR, and NELL-995). Training time refers to the time required for training the RL agent, excluding the pretraining time for knowledge graph embeddings, which is shared by MultiHopKG [33], MetaKGR [41], Path Spuriousness [14], and PMHR. MINERVA [10] does not include the pretraining process for knowledge graph embedding, which results in a shorter overall running time. Since our analysis focuses on the training efficiency of the RL agent, the pretraining time is excluded for all models. In Figure 4, the horizontal axis represents the datasets while the vertical axis shows the training time in minutes. Due to the significant differences in training time across datasets, we use a logarithmic scale of base 10 for the vertical axis to better visualize the results.

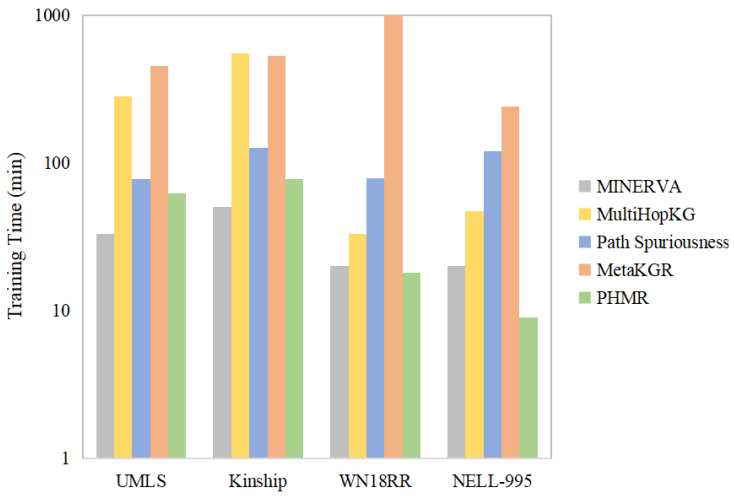


Figure 4. The efficiency comparison between PMHR and path-based models.

The results, as illustrated in Figure 4, highlight the balance our method achieves between training efficiency and effectiveness, as it consistently demonstrates competitive or superior performance in reducing RL agent training time while maintaining better overall results compared to MINERVA [10]. On UMLS, PMHR achieved the second shortest training time, only marginally slower than MINERVA, while delivering superior model performance. On Kinship, PMHR again ranked second in training efficiency, with a slight gap from MINERVA but significantly outperforming MultiHopKG [33], MetaKGR [41], and Path Spuriousness [14], with the latter two showing notably longer training times. On WN18RR, PMHR achieved the shortest training time, narrowly edging out MINERVA [10], while significantly outperforming the other baselines. The RL agent training time of MetaKGR [41] exceeded the vertical axis limit, underscoring its inefficiency in handling larger or more complex datasets. On NELL-995, PMHR demonstrated a more pronounced advantage, achieving the shortest RL agent training time among all models.

In summary, these results highlight the superiority of PMHR, which not only maintains RL agent training efficiency but also delivers better effectiveness compared to baseline models.

5.6. Ablation Study

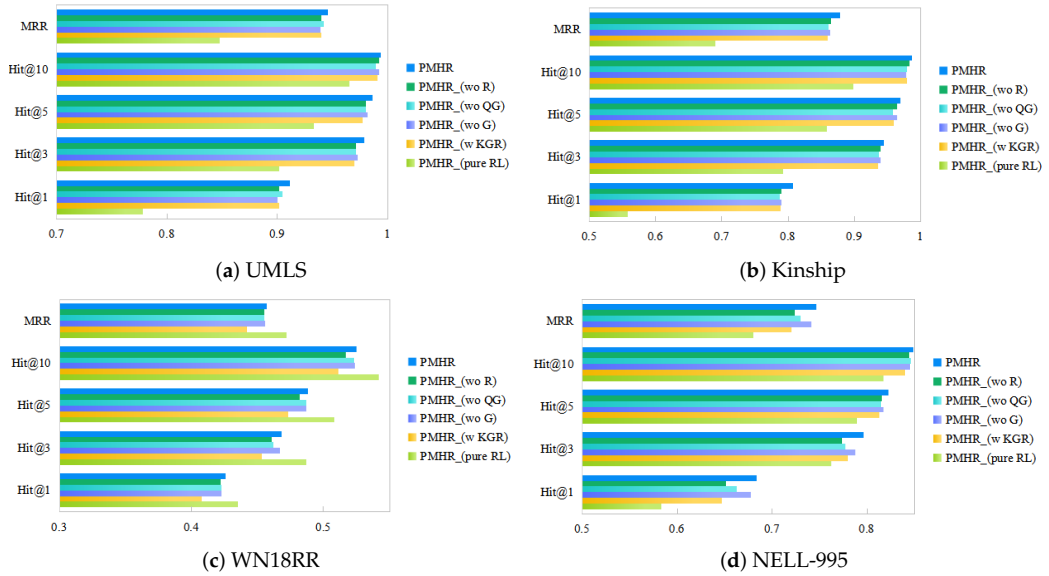
We performed ablation experiments on the four datasets to study the influence of each model part on the results using the following variants of PMHR:

- PMHR$_{(pure\ RL)}$: A pure RL method is used for KG reasoning without incorporating any enhancements in this variant.
- PMHR$_{(w\ KGR)}$: The ConvE KG representation learning method is used for soft rewards in this variant, which corresponds to MultiHopKG.
- PMHR$_{(wo\ G)}$: The graph structure information learning component is omitted in this variant.
- PMHR$_{(wo\ QG)}$: The query-related guiding information learning component is omitted in this variant.
- PMHR$_{(wo\ R)}$: The rule-based reward component is omitted in this variant.

The results shown in Figure 5 show that removing any component of the model reduced its effectiveness in most cases. This demonstrates that all parts contribute to the overall performance. Specifically, PMHR$_{(pure\ RL)}$ is a pure RL method without enhancement. It utilizes the randomly initialized vectors instead of vectors learned by KG representation learning. Consequently, PMHR$_{(pure\ RL)}$ achieved the worst performance on the UMLS, Kinship, and NELL-995 datasets. An interesting finding is that on the WN18RR dataset, PMHR$_{(pure\ RL)}$ achieved the best performance, which exceeds those of the various variant models, PMHR, and even the SOTA ConvE [21] and DisMult [18] models. Under the PMHR$_{(pure\ RL)}$ method, Hit@1 is 0.435, 0.542 for Hit@10, and 0.472 for MRR, which are better than those of the SOTA models. This is because the reward function is a crucial factor that affects the effectiveness of RL. It directly influences the behavior and decision making process of the agent during the learning process. As shown in Table 2, while the KG representation learning models performed well on other datasets with MRR values above 0.75, their performance on the WN18RR dataset was comparatively poor, with MRR values below 0.5. Thus, representation learning helped mitigate reward sparsity and provided positive guidance on the UMLS, Kinship, and NELL-995 datasets, whereas it interfered with the RL process on the WN18RR dataset. This interference occurred because a high positive reward (close to 1) was assigned to some paths that did not reach the target entity. Consequently, directly using the obtained representations did not have a positive impact on the RL environment and rewards. PMHR$_{(w\ KGR)}$ uses ConvE for KGR and soft rewards. The use of soft rewards helped alleviate the sparse rewards problem in RL and resulted in noticeable improvements on most datasets. A comparison between PMHR and PMHR$_{(wo\ G)}$ shows that the graph structure information helped enrich the state representations. Graph structures offer natural representations of the complex relations between entities. The explicit modeling of these structures allowed the complex interactions within the environment to be captured, thereby enhancing the state representation. A comparison between PMHR and PMHR$_{(wo\ QG)}$ shows that query-related guiding information allowed the agent to make correct choices and improved model performance. A comparison between PMHR and PMHR$_{(wo\ R)}$ shows that adding rule-based rewards improved the model. The agent was further constrained by the rules to find more reasonable paths that lead to the target entity and conform to the rules when it encountered spurious paths that led to the target entity but were unrelated to the query.

5.7. Case Study

In Table 3, we present one triple, two KG reasoning paths generated by PMHR, and three mined rules for the relation 'organizationhiredperson' on the NELL-995 dataset. The inverse of a directional relation r is represented by r^{-1}.



(**a**) UMLS (**b**) Kinship

(**c**) WN18RR (**d**) NELL-995

Figure 5. Ablation study on key parts of PMHR.

Table 3. Case study of PMHR on NELL-995.

Query Triple : (canadian jewish congress, organizationhiredperson, ?) Answer: sylvain_abitbol	
Path	
p_1: canadian jewish congress-organizationhiredperson->E_1-personleadsorganization->E_2-organizationhiredperson->sylvain_abitbol p_2: canadian jewish congress-organizationhiredperson->E_3-personbelongstoorganization->E_4-worksfor^{-1}->sylvain_abitbol	
Rules	Confidence
H_1: organizationhiredperson (X, A), personbelongstoorganization (A, B), worksfor^{-1} (B, Y) =>organizationhiredperson (X, Y)}	0.73
H_2: organizationhiredperson (X, A), coachesteam (A, B), worksfor^{-1} (B, Y) =>organizationhiredperson (X, Y)}	0.38
H_3: organizationhiredperson (X, A), personleadsorganization (A, B), organizationhiredperson (B, Y)=>organizationhiredperson (X, Y)}	0.23

The triple denotes the query triple. The middle section of the table shows the relevant reasoning paths found by the PMHR for the query (canadian jewish congress, organizationhiredperson, ?). The placeholders like 'E_i' are used in replace of the entities to make the presentation simpler, and only the relations are shown. The bottom of the table shows the rules associated with the relation 'organizationhiredperson' and their corresponding confidence values discovered through rule mining with AnyBURL.

As shown in Table 3, for the given query triple (canadian jewish congress, organizationhiredperson, ?), two reasoning paths p_1 and p_2 were identified by PMHR. Because both paths lead to the final target entity, they are assigned equal rewards of 1 by models such as MultiHopKG. However, from a human intuition perspective, the relations in p_1, such as 'personleadsorganization' and 'organizationhiredperson', introduce more intermediate entities. These relations are less clear and less reliable than those in p_2. Consequently, PMHR leverages rule-mining algorithms to obtain query-related rules and their corresponding confidence. The final rewards for each path are adjusted using the confidence of the rules associated with these paths as the rule rewards in such scenarios. For example, the final reward of p_1 is $1 + 0.23$ and that of p_2 is $1 + 0.73$. A more reasonable path has a higher confidence, which results in a larger reward value. This approach encourages the agent to identify logical paths, thereby mitigating the issue of spurious paths.

6. Implications and Conclusions

6.1. Theoretical and Practical Implications

Path-based multi-hop reasoning can be used not only to discover hidden information but also to provide explainable reasoning paths. With the help of KGE and logical rule mining

techniques, PMHR addresses the key limitations of existing path-based multi-hop reasoning methods through effective handling of sparse rewards and spurious paths, enhanced state representation, and improved algorithm efficiency. Because of the complexity of multi-hop reasoning, the performance of embedding-based and neural network-based models on complex reasoning tasks is limited by their difficulty in capturing long-distance or highly complex paths. Additionally, it can be challenging for these models to explain their reasoning process. The use of these models in prediction applications that require a high degree of trust such as drug design, disease diagnosis, and fault diagnosis becomes more challenging when users have low confidence in the models. Path-based reasoning can trace relationships between symptoms, diseases, and treatments, leading to more reliable medical diagnoses. Path-based multi-hop reasoning enhances the reliability, interpretability, and overall credibility of KGs by providing clear, traceable reasoning paths for the generated results.

6.2. Conclusions

In this paper, we propose the PMHR framework, which combines KG embeddings and logically rules with RL strategies. To address the issues of sparse rewards and spurious paths, soft rewards and rule-based rewards are integrated into PMHR to mitigate sparse rewards and encourage the agent to explore more reasonable paths to avoid spurious paths. To enhance state representation and provide more guidance to the agent, the state representation is redefined to incorporate current information, historical information, graph structure information, and query-related guiding information. Furthermore, the embeddings of entities and relations learned through KG representation are leveraged to initialize the environment and prune action space, thereby improving the efficiency of the RL algorithm and reducing the run time. The experimental results on benchmark datasets demonstrate how well PMHR works to increase the accuracy of KG reasoning while maintaining interpretability. On the UMLS, Kinship, and NELL-995 datasets, PMHR obtained Hit@1 increases of 0.63%, 2.02%, and 3.17%, respectively, compared to the state-of-the-art path-based models. In future work, we aim to investigate the alleviation of spurious paths using large language models and explore this area in greater depth.

Author Contributions: Investigation, A.M. and S.Z.; conceptualization, A.M. and Y.Y.; methodology, A.M.; writing—original draft preparation, A.M.; writing—review and editing, Y.Y., C.S. and L.P.; supervision, C.S., T.-S.C. and Y.Y.; funding acquisition, Y.Y. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the National Natural Science Foundation of China (grant ID U22B2019); National Key Research and Development Program(grant ID 2020YFB2104503).

Data Availability Statement: The data used in this study come from publicly available knowledge graph datasets. For more data information, please visit https://github.com/salesforce/MultiHopKG (accessed on 10 December 2022).

Conflicts of Interest: The authors declare no conflicts of interest.

Abbreviations

The following abbreviations are used in this manuscript:

KGs	Knowledge graphs
RL	Reinforcement learning
KGR	Knowledge graph reasoning
MDP	Markov decision process
GCN	Graph convolutional network
KGE	Knowledge graph embedding
LSTM	Long short-term memory

References

1. Ji, S.; Pan, S.; Cambria, E.; Marttinen, P.; Philip, S.Y. A survey on knowledge graphs: Representation, acquisition, and applications. *IEEE Trans. Neural Netw. Learn. Syst.* **2022**, *33*, 494–514. [CrossRef] [PubMed]
2. Xu, W.; Deng, Y.; Zhang, H.; Cai, D.; Lam, W. Exploiting reasoning chains for multi-hop science question answering. In Proceedings of the Findings of the Association for Computational Linguistics: EMNLP 2021, Virtual Event, 16–20 November 2021; pp. 1143–1156. [CrossRef]
3. Cui, H.; Peng, T.; Han, R.; Han, J.; Liu, L. Path-based multi-hop reasoning over knowledge graph for answering questions via adversarial reinforcement learning. *Knowl.-Based Syst.* **2023**, *276*, 110760. [CrossRef]
4. Heo, Y.J.; Kim, E.S.; Choi, W.S.; Zhang, B.T. 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, Dublin, Ireland, 22–27 May 2022; pp. 373–390. [CrossRef]
5. Ma, R.; Wang, X.; Cao, C.; Bu, X.; Wu, H.; Zhao, L. GLSEC: Global and local semantic-enhanced contrastive framework for knowledge graph completion. *Expert Syst. Appl.* **2024**, *250*, 123793. [CrossRef]
6. Wang, H.; Song, D.; Wu, Z.; Li, J.; Zhou, Y.; Xu, J. A collaborative learning framework for knowledge graph embedding and reasoning. *Knowl.-Based Syst.* **2024**, *289*, 111505. [CrossRef]
7. Meng, X.; Bai, L.; Hu, J.; Zhu, L. Multi-hop path reasoning over sparse temporal knowledge graphs based on path completion and reward shaping. *Inf. Process. Manag.* **2024**, *61*, 103605. [CrossRef]
8. Shang, B.; Zhao, Y.; Liu, Y.; Wang, C. Attention-based exploitation and exploration strategy for multi-hop knowledge graph reasoning. *Inf. Sci.* **2024**, *653*, 119787. [CrossRef]
9. Xiong, W.; Hoang, T.; Wang, W.Y. Deeppath: A reinforcement learning method for knowledge graph reasoning. In Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017, Copenhagen, Denmark, 9–11 September 2017; pp. 564–573. [CrossRef]
10. Das, R.; Dhuliawala, S.; Zaheer, M.; Vilnis, L.; Durugkar, I.; Krishnamurthy, A.; Smola, A.; McCallum, A. Go for a walk and arrive at the answer: Reasoning over paths in knowledge bases using reinforcement learning. In Proceedings of the 6th International Conference on Learning Representations, Vancouver, BC, Canada, 30 April–3 May 2018.
11. Wang, H.; Li, S.; Pan, R.; Mao, M. Incorporating graph attention mechanism into knowledge graph reasoning based on deep reinforcement learning. In Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, Hong Kong, China, 3–7 November 2019; pp. 2623–2631. [CrossRef]
12. Wan, G.; Pan, S.; Gong, C.; Zhou, C.; Haffari, G. Reasoning like human: Hierarchical reinforcement learning for knowledge graph reasoning. In Proceedings of the 29th International Joint Conference on Artificial Intelligence, Yokohama, Japan 7–15 January 2021. [CrossRef]
13. Tiwari, P.; Zhu, H.; Pandey, H.M. DAPath: Distance-aware knowledge graph reasoning based on deep reinforcement learning. *Neural Netw.* **2021**, *135*, 1–12. [CrossRef] [PubMed]
14. Jiang, C.; Zhu, T.; Zhou, H.; Liu, C.; Deng, T.; Hu, C.; Li, J. Path Spuriousness-aware Reinforcement Learning for Multi-Hop Knowledge Graph Reasoning. In Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics, Dubrovnik, Croatia, 2–6 May 2023; pp. 3173–3184. [CrossRef]
15. Bordes, A.; Usunier, N.; Garcia-Duran, A.; Weston, J.; Yakhnenko, O. Translating embeddings for modeling multi-relational data. In Proceedings of the Advances in Neural Information Processing Systems, Lake Tahoe, NA, USA, 5–8 December 2013; Volume 26.
16. Ji, G.; He, S.; Xu, L.; Liu, K.; Zhao, J. Knowledge graph embedding via dynamic mapping matrix. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing*; Association for Computational Linguistics: Beijing, China, 2015; pp. 687–696. [CrossRef]
17. Nickel, M.; Tresp, V.; Kriegel, H.P. A three-way model for collective learning on multi-relational data. In Proceedings of the 28th International Conference on Machine Learning, Bellevue, DC, USA, 28 June–2 July 2011; pp. 3104482–3104584.
18. Yang, B.; Yih, W.t.; He, X.; Gao, J.; Deng, L. Embedding entities and relations for learning and inference in knowledge bases. In Proceedings of the 3rd International Conference on Learning Representations, San Diego, CA, USA, 7–9 May 2015.
19. Trouillon, T.; Welbl, J.; Riedel, S.; Gaussier, É.; Bouchard, G. Complex embeddings for simple link prediction. In Proceedings of the 33nd International Conference on Machine Learning, New York, NY, USA, 19–24 June 2016; pp. 2071–2080.
20. Socher, R.; Chen, D.; Manning, C.D.; Ng, A. Reasoning with neural tensor networks for knowledge base completion. In Proceedings of the Advances in Neural Information Processing Systems, Lake Tahoe, NA, USA, 5–8 December 2013; Volume 26.
21. Dettmers, T.; Minervini, P.; Stenetorp, P.; Riedel, S. Convolutional 2d knowledge graph embeddings. In Proceedings of the 32nd AAAI Conference on Artificial Intelligence, New Orleans, LA, USA, 2–7 February 2018; Volume 32, pp. 1811–1818. [CrossRef]
22. Nguyen, D.Q.; Nguyen, T.D.; Nguyen, D.Q.; Phung, D. A novel embedding model for knowledge base completion based on convolutional neural network. In Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, New Orleans, LA, USA, 1–6 June 2018; pp. 327–333. [CrossRef]
23. Lv, X.; Lin, Y.; Cao, Y.; Hou, L.; Li, J.; Liu, Z.; Li, P.; Zhou, J. Do Pre-trained Models Benefit Knowledge Graph Completion? A Reliable Evaluation and a Reasonable Approach. In Proceedings of the Findings of the Association for Computational Linguistics, Dublin, Ireland, 22–27 May 2022; pp. 3570–3581. [CrossRef]

24. Wang, L.; Zhao, W.; Wei, Z.; Liu, J. SimKGC: Simple contrastive knowledge graph completion with pre-trained language models. In Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics, Dublin, Ireland, 22–27 May 2022; pp. 4281–4294. [CrossRef]
25. Xie, X.; Zhang, N.; Li, Z.; Deng, S.; Chen, H.; Xiong, F.; Chen, M.; Chen, H. From discrimination to generation: Knowledge graph completion with generative transformer. In Proceedings of the Companion Proceedings of the Web Conference 2022, Lyon, France, 25–29 April 2022; pp. 162–165. [CrossRef]
26. Zhu, Y.; Wang, X.; Chen, J.; Qiao, S.; Ou, Y.; Yao, Y.; Deng, S.; Chen, H.; Zhang, N. Llms for knowledge graph construction and reasoning: Recent capabilities and future opportunities. *arXiv* **2023**, arXiv:2305.13168. [CrossRef]
27. Wei, Y.; Huang, Q.; Kwok, J.T.; Zhang, Y. KICGPT: Large Language Model with Knowledge in Context for Knowledge Graph Completion. In Proceedings of the Findings of the Association for Computational Linguistics: EMNLP 2023, Singapore, 6–10 December 2023; pp. 8667–8683. [CrossRef]
28. Zhou, J.; Cui, G.; Hu, S.; Zhang, Z.; Yang, C.; Liu, Z.; Wang, L.; Li, C.; Sun, M. Graph neural networks: A review of methods and applications. *AI Open* **2020**, *1*, 57–81. [CrossRef]
29. Chen, J.; Hou, H.; Gao, J.; Ji, Y.; Bai, T. RGCN: Recurrent graph convolutional networks for target-dependent sentiment analysis. In *Knowledge Science, Engineering and Management*; Springer: Berlin/Heidelberg, Germany, 2019; pp. 667–675. [CrossRef]
30. Shang, C.; Tang, Y.; Huang, J.; Bi, J.; He, X.; Zhou, B. End-to-end structure-aware convolutional networks for knowledge base completion. In Proceedings of the 33rd AAAI Conference on Artificial Intelligence, Honolulu, HI, USA, 27 January–1 February 2019; pp. 3060–3067. [CrossRef]
31. Zhou, L.; Wang, T.; Qu, H.; Huang, L.; Liu, Y. A weighted GCN with logical adjacency matrix for relation extraction. In Proceedings of the 24th European Conference on Artificial Intelligence, Santiago de Compostela, Spain, 29 August–8 September; IOS Press: Amsterdam, The Netherlands, 2020; Volume 325, pp. 2314–2321. [CrossRef]
32. Vashishth, S.; Sanyal, S.; Nitin, V.; Talukdar, P.P. Composition-based Multi-Relational Graph Convolutional Networks. In Proceedings of the 8th International Conference on Learning Representations, Addis Ababa, Ethiopia, 26–30 April 2020.
33. Lin, X.V.; Socher, R.; Xiong, C. Multi-hop knowledge graph reasoning with reward shaping. In Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, 31 October–4 November 2018; pp. 3243–3253. [CrossRef]
34. Li, R.; Cheng, X. DIVINE: a generative adversarial imitation learning framework for knowledge graph reasoning. In Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, Hong Kong, China, 3–7 November 2019; pp. 2642–2651. [CrossRef]
35. Wan, G.; Du, B. Gaussianpath: A bayesian multi-hop reasoning framework for knowledge graph reasoning. In Proceedings of the 35th AAAI Conference on Artificial Intelligence, Virtual, 2–9 February 2021; pp. 4393–4401. [CrossRef]
36. Zhou, X.; Wang, P.; Luo, Q.; Pan, Z. Multi-hop knowledge graph reasoning based on hyperbolic knowledge graph embedding and reinforcement learning. In Proceedings of the 10th International Joint Conference on Knowledge Graphs, Virtual, 6–8 December 2021; pp. 1–9. [CrossRef]
37. Lv, X.; Han, X.; Hou, L.; Li, J.; Liu, Z.; Zhang, W.; Zhang, Y.; Kong, H.; Wu, S. 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, Online, 16–20 November 2020; pp. 5694–5703. [CrossRef]
38. Meilicke, C.; Chekol, M.W.; Fink, M.; Stuckenschmidt, H. Reinforced anytime bottom up rule learning for knowledge graph completion. *arXiv* **2020**, arXiv:2004.04412.
39. Kok, S.; Domingos, P. Statistical predicate invention. In Proceedings of the 24th International Conference on Machine Learning, Corvallis, OR, USA, 20–24 June 2007; pp. 433–440.
40. Yang, F.; Yang, Z.; Cohen, W.W. Differentiable learning of logical rules for knowledge base reasoning. In Proceedings of the Advances in Neural Information Processing Systems, Long Beach, CA, USA, 4–9 December 2017; Volume 30, pp. 2319–2328.
41. Lv, X.; Gu, Y.; Han, X.; Hou, L.; Li, J.; Liu, Z. Adapting meta knowledge graph information for multi-hop reasoning over few-shot relations. In Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, Hong Kong, China, 3–7 November 2019; pp. 3374–3379. [CrossRef]
42. Hou, Z.; Jin, X.; Li, Z.; Bai, L. Rule-aware reinforcement learning for knowledge graph reasoning. In Proceedings of the Findings of the Association for Computational Linguistics, Online, 1–6 August 2021; pp. 4687–4692. [CrossRef]
43. Kingma, D.P.; Ba, J. Adam: A method for stochastic optimization. In Proceedings of the 3rd International Conference on Learning Representations, San Diego, CA, USA, 7–9 May 2015.