# A Collaborative Reasoning Framework Powered by Reinforcement Learning and Large Language Models for Complex Questions Answering over Knowledge Graph

Anonymous ACL submission

## Abstract

Knowledge Graph Question Answering (KGQA) aims to automatically answer natural language questions by reasoning across multiple triples in knowledge graphs (KGs). Reinforcement learning (RL)-based methods are introduced to enhance model interpretability. Nevertheless, when addressing complex questions requiring long-term reasoning, the RL agent is usually misled by aimless exploration, as it lacks common learning practices with prior knowledge. Recently, large language models (LLMs) have been proven to encode vast amounts of knowledge about the world and possess remarkable reasoning capabilities. However, they often encounter challenges with hallucination issues, failing to address complex questions that demand deep and deliberate reasoning. In this paper, we propose a collaborative reasoning framework (CRF) powered by RL and LLMs to answer complex questions based on the knowledge graph. Our approach leverages the common sense priors contained in LLMs while utilizing RL to provide learning from the environment, resulting in a hierarchical agent that uses LLMs to solve the complex KGQA task. By combining LLMs and the RL policy, the high-level agent accurately identifies constraints encountered during reasoning, while the low-level agent conducts efficient path reasoning by selecting the most promising relations in KG. Extensive experiments conducted on four benchmark datasets clearly demonstrate the effectiveness of the proposed model, which surpasses state-of-the-art approaches.

## 1 Introduction

Knowledge Graph Question Answering (KGQA) is a classical NLP task to automatically answer natural language questions by reasoning across multiple triples in a given knowledge graph (KG). The KGQA system uses data from knowledge graphs to accurately answer user queries. It has significant applications in various fields, making it a key focus of academic research and industry innovation. As user demands become increasingly intricate, recent research has attempted to build KGQA systems capable of answering complex questions to better adapt to real-world scenarios.

Complex questions often involve constraints that necessitate logical, quantitative, and aggregation reasoning across a series of KG triples. To answer these complex questions effectively, some methods (Chen et al., 2019; Han et al., 2020) introduce a hop-by-hop inference process to select the multi-hop relation paths. They are trained in strong supervision through pre-annotated intermediate golden relations, thus achieving promising performance for complex KGQA. Unfortunately, due to the high cost of data annotation, complex questions are only annotated with final answers, resulting in weak supervision. To tackle the issue, several studies (Cui et al., 2023; Zhang et al., 2022; Qiu et al., 2020) train a reinforcement learning (RL) agent to sequentially extend its path of reasoning within the KG by iteratively selecting the most promising actions until the target entity is reached. RL-based methods exhibit strong performance in both effectiveness and interpretability, as they establish an interpretable inference chain throughout the sequential decision-making process. However, when tackling complex questions that require long-term reasoning, the RL agent is usually misled by aimless exploration, due to its lack of common learning practices with prior knowledge. This issue prevents the rapid convergence of the RL agent, thereby diminishing its exploration efficiency (Lv et al., 2020). Furthermore, the majority of existing methods encounter challenge in performing effective reasoning in the knowledge graph for complex questions with constraints.

The emergence of large language models (LLMs) in recent times, such as GPT4 (Achiam et al., 2023) and Llama (Touvron et al., 2023b), have been shown to encode a tremendous amount of knowledge about the world by virtue of being trained on massive amounts of text. LLMs have achieved significant success in various tasks, which encourages their application in KGQA research (Luo et al., 2023; Li et al., 2023; Jiang et al., 2023). While these studies have significantly enhanced the performance of KGQA systems, they often encounter challenges with hallucination issues, failing to provide stable and responsible answers when faced with complex questions requiring deliberate reasoning (Ye et al., 2023). This limitation is expected due to the training methodology employed by LLMs, where they are trained to predict the next token in sequence based on the context provided, without an interval for deliberate thoughts. As explained in (Kahneman, 2011), our cognition comprises two systems: System 1 is an intuitive and unconscious thinking system that relies on experience, while System 2 employs knowledge for deliberate and reliable logical reasoning. Currently, LLMs exhibit characteristics that are more in line with System 1 thinking, which may account for their shortfall in addressing complex questions.

To this end, in this paper, we propose a collaborative reasoning framework (CRF) that integrates large language models (LLMs) and hierarchical reinforcement learning (HRL) to mimic human cognitive processes, thereby enhancing the ability to answer complex questions based on the knowledge graph. Our approach leverages the common sense priors contained in LLMs while utilizing RL to provide learning from the environment, resulting in a hierarchical agent that uses LLMs to solve the complex KGQA task. Specifically, the proposed model dismantles the KGQA task into a two-level hierarchical decision process. In the high-level process, the agent employs RL policy to identify constraints (options) encountered during reasoning. Furthermore, LLMs output the probability of each option based on the current state and in-context demonstration, serving as intermediate rewards to address the challenge of delayed and sparse rewards due to weak supervision. In the low-level process, the agent combining LLMs and the RL policy conducts efficient path reasoning by selecting the most promising relations (actions) in KG. More concretely, we use LLMs to inject common sense priors into the agent. The LLMs guide the agent by suggesting the most likely courses of action to avoid aimless exploration, significantly improving learning efficiency. Additionally, the trained policy-based agent can provide deliberate and reliable logical reasoning as verification for LLMs to eliminate hallucinations. In summary, the main contributions of this paper can be summarized as follows:

- We propose a collaborative reasoning framework powered by hierarchical RL and LLMs to mimic human cognitive processes. Our approach leverages the common sense priors contained in LLMs while utilizing RL to provide learning from the environment, resulting in a hierarchical agent that uses LLMs to solve the complex KGQA task.

- We dismantle the KGQA task into a high-level process for constrain detection and a low-level process for path reasoning, respectively. By combining LLMs and the RL policy, the hierarchical agents can tackle the challenges of aimless exploration and hallucination for complex question answering based on the knowledge graph.

- We validate the efficacy of the proposed framework through comprehensive experiments and meticulous ablation studies on widely-used benchmark datasets. Empirical results demonstrate that our method achieves state-of-the-art performance for complex KGQA.

## 2 Related Work

Our KGQA method is closely related to the studies on Reinforcement Learning and Large Language Models.

**Reinforcement Learning (RL) for KGQA.** The RL-based methods are proposed to frame complex KGQA as a sequential decision-making process to extend its reasoning path within the KG by iteratively selecting promising actions until reaching the target entity. SRN (Qiu et al., 2020) performs an effective path search over KG to infer answer entities based on RL. ARL (Zhang et al., 2022) proposes a new adaptive reinforcement learning framework and introduces three atomic operations to adaptively extend the relation paths. ARN (Cui et al., 2023) incorporates KG embeddings as anticipation information into RL framework to capture the potential target information for multihop reasoning. Moreover, Zhu et al. (2022) applies a hierarchical reinforcement learning framework to tackle the challenge of one-to-many relation-entity in knowledge graph reasoning task. While these methods exhibit strong performance in both effectiveness and interpretability, they struggle to perform effective reasoning in the knowledge graph for complex questions with constraints and are influenced by aimless exploration. Our work extends the line of RL-based models, utilizing hierarchical RL to formulate complex KGQA as a hierarchical decision problem. It introduces a novel perspective to handle complex questions in knowledge graph reasoning

**Large Language Models (LLMs) for KGQA.** LLMs have achieved significant success in various tasks, which encourages their application in KGQA research. The most intuitive idea is to use LLMs as parsers to generate logical forms for questions. KB-BINDER (Li et al., 2023) uses LLMs to create preliminary logical forms through demonstration imitation and then binds the draft to an executable version through knowledge base integration. ChatKBQA (Luo et al., 2023) proposes generating the logical form with fine-tuned LLMs first, then retrieving and replacing entities and relations through an unsupervised retrieval method. Furthermore, FlexKBQA uses LLMs to generate synthetic data for the KGQA task, then use the data to fine-tune a smaller light-weight KGQA model. In addition, novel methods have been proposed to fully leverage the reasoning capability of LLMs. ToG (Sun et al., 2023) enables the LLM agent to iteratively execute beam search on KG, discover the most promising reasoning paths, and return the most likely reasoning results. StructGPT (Jiang et al., 2023) gathers relevant evidence from structured data, allowing LLMs to focus on the reasoning task using the acquired information. However, they often encounter challenges with hallucination issues, failing to address complex questions.

## 3 Methodology

### 3.1 Problem Formulation

Our study focuses on factoid question answering over a knowledge graph (KG). The KG can be formally represented as $G = (E, R)$, where $E$ is the set of entities and $R$ is the set of relations. Given a natural language question $q$ and a KG $G$, our goal is to take an optimal reasoning process to predict the answer entities $A_q \in E$. In this paper, we focus on handling complex questions, in which the corresponding answer entities are multi-hop away from the topic entities, and the questions may contain constraints, such as entity constraint, numerical constraint, etc.

### 3.2 Framework Overview

Our methodology utilizes the inherent common sense priors in LLMs and incorporates RL for environmental learning, leading to the construction of a hierarchical agent for solving the complex KGQA task. Figure 1 shows the overall architecture of our framework, which mainly consists of four parts: a high-level policy-based agent for constraint detection, a low-level policy-based agent for path reasoning, LLMs as reward function and LLMs as guider. The high-level policy-based agent identifies constraints (options) encountered during reasoning, while the low-level policy-based agent conducts efficient path reasoning by selecting the most promising relations (actions) in KG. The LLMs-based reward function generates the probability of each option as intermediate rewards for the high-level policy-based agent. Moreover, the LLMs guide the low-level policy-based agent by suggesting the most likely courses of action.

### 3.3 High-level Process for Constraint Detection

In the high-level process, the agent is designed to detect the constraint type of current timestep $t$, which guides the corresponding path reasoning.

**State.** At timestep $t$, the state of the high-level process is defined as $S_t^h = (e_0, q, e_t, h_t)$, where $e_0$ is the topic entity of the given question $q$; $e_t$ represents the current entity at timestep $t$ during reasoning process; and $h_t$ refers the representation of the historical relation path selected, we apply LSTM to represent the sequential path information as following: $h_t = LSTM(h_{t-1}, r_{t-1})$. Note that $h_0$ and $r_0$ are both set to zero vectors.

**Option.** To handle different types of complex questions, we define six types of options corresponding to the constraints: $Basic$, $Bridge$, $Union$, $Filter$, $Ordinal$, and $Aggregation$. Specifically, $Basic$ indicates that the current option is generating the reasoning path by adding relations hop-by-hop. In $Bridge$, the option can incorporate the path from different topic entities to handle the questions with multiple topic entities. The option $Union$ aims to solve questions containing multiple relations from the same topic entity. The option of $Filter$ represents a numerical or temporal comparison, including $<, \leq, >, \geq, =, \neq$. In $Ordinal$, the option involves sorting the current

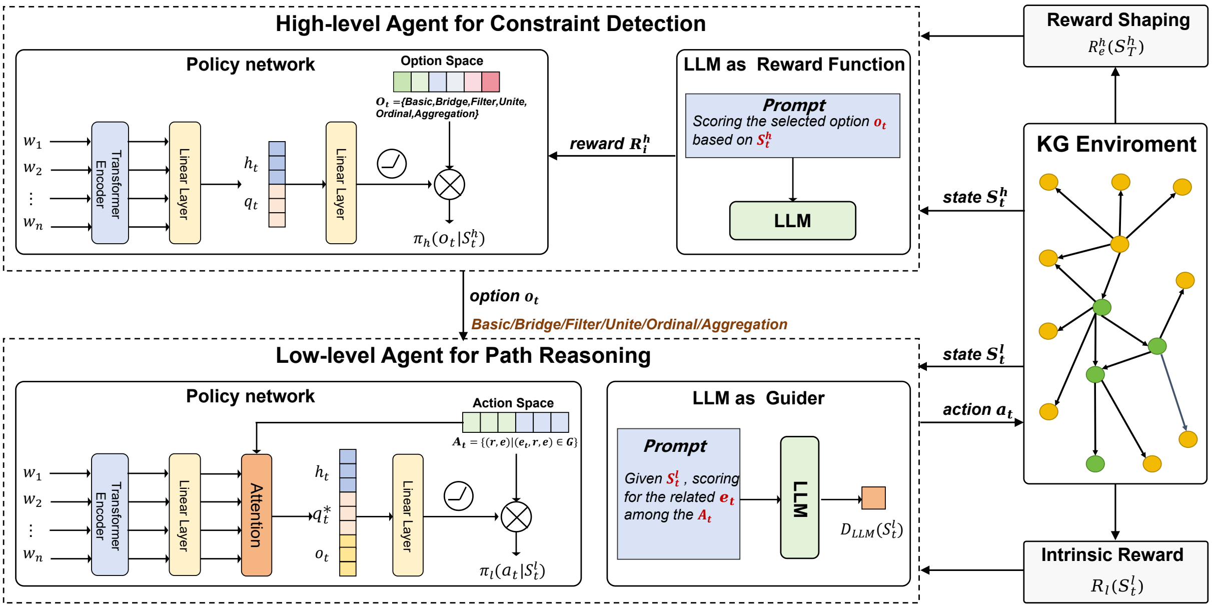


Figure 1: The overview of our proposed collaborative reasoning framework (CRF).

entity set in either ascending or descending order and selecting entities based on the ordinal number. Within $Aggregation$, the option signifies the use of aggregation functions on the current entities, such as Count, Limit, etc. At the timestep $t$, the option $o_t$ is selected from option space $O_t = \{Basic, Bridge, Union, Filter, Aggregation, Ordinal\}$.

**Reward.** The rewards received by the high-level agent are primarily divided into two parts: extrinsic rewards provided by the environment KG and the likelihood of options output by LLMs. Regarding extrinsic rewards, since only the final answers are labeled as weak supervision, we utilize the scoring function of the KGE model to calculate soft rewards for candidate entities (Cui et al., 2023). Formally, the extrinsic reward function $R_e^h(S_T^h)$ is defined as follows:

$$R_e^h(S_T^h) = \begin{cases} 1, & e_T = \hat{e} \\ f_s(e_0, q, e_T), & \text{otherwise} \end{cases} \tag{1}$$

where $e_T$ is the predicted entity and $\hat{e}$ is the golden answer. Intuitively, if $e_T$ matches $\hat{e}$, the agent gets a reward of 1; otherwise, it receives a soft reward between 0 and 1 from the scoring function $f_s()$.

In addition, the extensive training data of LLMs enables them to excel as in-context learners and also equips them to comprehend significant common-sense priors to assess the reasonableness of the selected constraint types. Therefore, LLMs can output the probability of each option based on the current state and in-context demonstration, serving as intermediate rewards. Formally, the intermediate reward function $R_i^h(S_t^h)$ is defined as follows:

$$R_i^h(o_t, S_t^h) = P_{LLM}(\boldsymbol{o_t}, \boldsymbol{S_t^h}, \boldsymbol{\rho}) \tag{2}$$

where $\boldsymbol{o_t}$ and $\boldsymbol{S_t^h}$ represent the textual descriptions of the option and state at the timestep $t$. The details of the prompt $\rho$ are described in Appendix C. Finally, the high-level reward function can be defined as follows:

$$R^h(o_t, S_t^h, S_T^h) = R_i^h(o_t, S_t^h) + R_e^h(S_T^h) \tag{3}$$

**Policy.** The policy for the high-level process takes state information as input and outputs the probabilities over candidate options at each step. Specifically, we use Transformer encoder to obtain the question representation denoted as $\boldsymbol{q} = [\boldsymbol{w_1}, \boldsymbol{w_2}, \boldsymbol{...w_n}]$. To make the agent aware of the current step, a linear network is explored to generate step-aware question representation $\boldsymbol{q_t} \in \mathbb{R}^{d \times n}$:

$$\boldsymbol{q_t} = Tanh(W_t \cdot \boldsymbol{q} + b_t) \tag{4}$$

Where $W_t \in \mathbb{R}^{d \times n}$ and $b_t \in \mathbb{R}^{d \times 1}$ are learnable parameters. At timestep $t$, the high-level agent receives state $S_t^h$, and selects an option according to the calculated probability distribution. Additionally, a "Loop" option is added into the option space to signify when the reasoning process should stop. The option space is encoded by stacking the embedding of all valid options in $O_t$ : $\boldsymbol{O_t} \in \mathbb{R}^{|O_t| \times d}$.

And the high-level policy network $\pi_h$ is defined as:

$$\pi_h(o_t|S_t^h) = \boldsymbol{O_t} \cdot W_{h2} \cdot ReLu(W_{h1} \cdot [\boldsymbol{h_t}; \boldsymbol{q_t}]) \quad (5)$$

where $W_{h1}$ and $W_{h2}$ are parameter matrices. $[\boldsymbol{h_t}; \boldsymbol{q_t}]$ denotes the concatenation of encoded decision history and step-aware question vector.

### 3.4 Low-level Process for Path Reasoning

Once the high-level agent has detected a constraint type, the low-level agent will execute path reasoning to select the most promising relation. To make the detected constraint type accessible in the low-level process, the option $o_t$ is taken as additional input for guiding the low-level path reasoning process.

**State.** Similar to the policy for constraint detection, the low-level intra-option state includes topic entity $e_0$, given question $q$, historical information $h_t$. In addition, it also contains the high-level option $o_t$, which can affect the learning of low-level strategies. Formally, the low-level process state can be defined as $S_t^l = (e_0, q, e_t, h_t, o_t)$.

**Action.** For the option $o_t$, the low-level agent takes action $a_t$ to select the most promising relation. The action space for the state $S_t^l$ is the set of outgoing edges of the current entity $e_t$, where $A_t = \{(r, e)|(e_t, r, e) \in G\}$. Meanwhile, each option possesses its own list of valid actions for every step. Based the option $Basic$, the agent adds a single-hop relation $r_t$ to $h_t$. Note that some relations in the KG are 1-to-many. Therefore, the target $e_{t+1}$ may be a set of entities. For the options $Bridge$, $Union$, and $Filter$, the relation $r_t$ is selected based on constraints including entity, relation, or numerical constraints. In addition, for the options $Ordinal$ and $Aggregation$, the low-level RL policy struggles to select the correct actions. Here, we enable the agent to directly perform correct reasoning with the assistance of LLMs.

**Reward.** The low-level agent receives an immediate reward by intrinsic motivation. Since a correct decision contains a KG relation which covers part of the semantic information of the question, we measure the semantic similarity between the given question and the selected relation as the low-level reward, defined as follows:

$$R^l(S_t^l) = ReLU(cos[\boldsymbol{r_t}; \boldsymbol{q_t^*}]) \quad (6)$$

where $\boldsymbol{r_t}$ is the representation of the selected relation; $\boldsymbol{q_t^*}$ is produced as the result of the interaction between relation $r_t$ and question $q$ according to the attention weights.

**Policy.** Once an option $o_t$ is selected, the policy for the low-level process takes action $a_t$ to conduct path reasoning. Given the low-level state $S_t^l$ and action space $A_t$, a relation-aware question representation can be calculated for each action $a_t = (r^*, e^*) \in A_t$:

$$\boldsymbol{q_t^*} = \sum_{i=1}^{n} \alpha_i^* \cdot \boldsymbol{w_{t,i}} \quad (7)$$

$$\alpha_i^* = \sigma(W_a \cdot (\boldsymbol{r^*} \odot \boldsymbol{w_{t,i}}) + b) \quad (8)$$

where $\sigma$ is the SoftMax operator; $\boldsymbol{q_t^*}$ is the result of the interaction between the relation and the question according to the attention weights; $\boldsymbol{r^*}$ is the vector of relation $r^*$; $\boldsymbol{w_{t,i}}$ is the step-aware representation of token $\boldsymbol{w_i}$; $W_a$ and $b$ are learnable parameters. Moreover, the action space is encoded by stacking the embeddings of all valid actions in $A_t$ : $\boldsymbol{A_t} \in \mathbb{R}^{|A_t| \times d}$. Therefor, the low-level policy network $\pi_l$ is defined as:

$$\pi_l(a_t|S_t^l) = \boldsymbol{A_t} \cdot W_{l2} \cdot ReLu(W_{l1} \cdot [\boldsymbol{h_t}; \boldsymbol{q_t^*}; \boldsymbol{o_t}]) \quad (9)$$

where $W_{l1}$ and $W_{l2}$ are parameter matrices; $[\boldsymbol{h_t}; \boldsymbol{q_t^*}; \boldsymbol{o_t}]$ represents the concatenation of encoded decision history, relation-aware question vector and option embedding. In addition, utilizing RL for exploration without relying on commonsense intuition may be inefficient when addressing complex queries that require long-term reasoning. Therefore, the section 3.5 introduces how to utilize LLMs to improve exploration in the low-level policy.

### 3.5 Using LLMs to Guide Low-level Policy

In the low-level process, the agent can conduct efficient exploration in the KG through the combination of LLMs and the RL policy. The common sense priors and planning capabilities of LLMs can be injected into the policy-based agent to improve low-level action selection in the form of language. The core idea is to use LLMs to obtain a value that approximates the probability that each candidate action is relevant to answer the question. Specifically, the LLMs are used to evaluate the function $f_{LLM}(e_t^*, a_t^i, o_t^*, q, h_t^*)$ for each candidate action at timestep $t$, where $e_t^*$, $o_t^*$ and $h_t^*$ are the language description of the current entity, the current option and the historical selected relation path; $a_t^m$

represents the language description of the corresponding relation and tail entity. Essentially, the LLM answers the following question: given the task of answering question $q$, based on the current entity $e_t^*$, the current option $o_t^*$ and the historical selected relation path $h_t^*$, should we choose the candidate action $a_t^m$ ? The output of the LLM, 'yes' or 'no', can easily be converted to an int ("0" or "1"). Further details are listed in Appendix D. Through these question-answering prompts, we can acquire common-sense priors from the LLMs. After evaluating this for each of the k candidate actions in the action space, we utilize the SoftMax function for normalization. The formula is represented as follows.

$$D_{LLM} = SoftMax([f_{LLM_1}, f_{LLM_2}, ..., f_{LLM_k}]) \tag{10}$$

where $D_{LLM}$ represents the probability of each candidate action evaluated through common sense priors from LLMs. Since the RL agent encounters the issue of aimless exploration due to a lack of common sense. We use the LLMs $D_{LLM}$ to guide exploration by suggesting the most likely courses of action. In the low-level process, the probability distribution of candidate actions is calculated through low-level policy $\pi_l$ and LLMs $D_{LLM}$. The action selection is formalized as follows.

$$S(a_t|S_t^l) = \pi_l(a_t|S_t^l) + D_{LLM}(S_t^l) \tag{11}$$

where the prompts inputted into the LLM are obtained by the current state $S_t^l$.

### 3.6 Optimization and Inference

**Optimization.** During the model training, we exploit the REINFORCE algorithm (Williams, 1992) to optimize the above policy networks. The algorithm utilizes the current policy to generate numerous trajectories for the purpose of estimating a stochastic gradient, subsequently updating the policy via stochastic gradient ascent.

For the high-level and low-level policies optimization, we maximize the expected cumulative rewards over all the question-answer pairs $(q, a)$. The object functions for the high-level policy and low-level policy are computed as follows:

$$\begin{aligned}\mathcal{J}(\theta_H) =& E_{(q,a)\in D}[E_{o_1,o_2,...,o_T\sim\pi_h}[\sum_{t=1}^{T}\eta^{T-t} \\ & R^h(o_t, S_t^h, S_T^h)]]\end{aligned} \tag{12}$$

$$\mathcal{J}(\theta_L) = E_{(q,a)\in D}[E_{a_1,a_2,...,a_T\sim\pi_l}[\sum_{t=1}^{T} R^l(S_t^l)]] \tag{13}$$

where $\eta$ is a discount factor. With the likelihood ratio trick, the gradients for the high-level policy and low-level policy are denoted as:

$$\begin{aligned}\nabla_{\theta_H}\mathcal{J}(\theta_H) = E_{(q,a)\in D}[E_{o_1,o_2,...,o_T\sim\pi_h}[\sum_{t=1}^{T}\eta^{T-t} \\ R^h(o_t, S_t^h, S_T^h)\nabla_{\theta_H}\log\pi_h]]\end{aligned} \tag{14}$$

$$\begin{aligned}\nabla_{\theta_L}\mathcal{J}(\theta_L) = E_{(q,a)\in D}[E_{a_1,a_2,...,a_T\sim\pi_l}[\sum_{t=1}^{T} \\ R^l(S_t^l)\nabla_{\theta_L}\log\pi_l]]\end{aligned} \tag{15}$$

**Inference.** In the inference stage, our method imitates human cognitive processes. During the long-term reasoning process to address a complex question, LLMs utilize their preexisting knowledge to provide intuitive assessments for each action, while the trained policy-based agent can offer deliberate and reliable logical reasoning as validation, aiding LLMs in discarding hallucinations.

## 4 Experiment

### 4.1 Dataset

In order to evaluate the effectiveness of the proposed CRF method, we conduct experiments using four public datasets, including WebQuestionSP (WebQSP) (Yih et al., 2016), ComplexWebQuestions (CWQ) (Talmor and Berant, 2018), PathQuestion (PQ) (Zhou et al., 2018) and MetaQA (Zhang et al., 2018). We give a detailed description of each dataset in Appendix A.

### 4.2 Baselines

To comprehensively evaluate our approach, we select a series of following baseline models for comparison, which can be divided into three categories: (1) **IR-based methods**, including EmbedKGQA (Saxena et al., 2020), NSM (He et al., 2021), TransferNet (Shi et al., 2021); (2) **RL-based methods**, including SRN (Qiu et al., 2020), ARL (Zhang et al., 2022), ARN (Cui et al., 2023); (3) **LLMs-based methods**, including Llama-2-70B (Touvron et al., 2023a), ChatGPT, KD-CoT (Wang et al., 2023), ToG (Sun et al., 2023), StructGPT (Jiang et al., 2023). The detailed description is introduced in Appendix B

Table 1: Experimental results ($\%$Hits@1) on four datasets. The best score is in **bold**, second best score is underlined. "-" indicating that no results are reported in the original papers. Results with ♯ are reprinted from (Sun et al., 2023).

| Model | WebQSP | CWQ | PathQuestion | | | MetaQA | | |
|---|---|---|---|---|---|---|---|---|
| | Mix | Mix | 2H | 3H | Mix | 1H | 2H | 3H |
| EmbedKGQA (Saxena et al., 2020) | 66.6 | 37.5 | - | - | - | 97.5 | 98.8 | 94.8 |
| NSM (He et al., 2021) | 68.7 | 47.6 | - | - | - | 94.8 | 97.0 | 91.0 |
| TransferNet (Shi et al., 2021) | 71.4 | 48.6 | - | - | - | 96.5 | 97.5 | 90.1 |
| SRN (Qiu et al., 2020) | - | - | 96.3 | 89.2 | 89.3 | 97.0 | 95.1 | 75.2 |
| ARL (Zhang et al., 2022) | 72.9 | 48.9 | - | - | - | 97.5 | 99.9 | 98.9 |
| ARN (Cui et al., 2023) | 68.0 | - | 98.9 | 90.5 | 93.6 | 96.7 | 93.5 | 97.0 |
| Llama-2-70B(Touvron et al., 2023a) ♯ | 57.4 | 39.1 | - | - | - | - | - | - |
| ChatGPT ♯ | 62.2 | 38.8 | - | - | - | 61.9 | 31.0 | 43.2 |
| KD-CoT (Wang et al., 2023) | 73.7 | 50.5 | - | - | - | - | - | - |
| ToG w/ChatGPT (Sun et al., 2023) | 76.2 | 58.9 | - | - | - | - | - | - |
| StructGPT (Jiang et al., 2023) | 72.6 | - | - | - | - | 94.2 | 93.9 | 80.2 |
| **CRF(ours)** | **79.5** | **68.2** | **99.4** | **95.7** | **97.1** | **99.6** | **99.9** | **99.2** |

### 4.3 Experimental Setting

Following (Cui et al., 2023), we use the PageRank-Nibble algorithm (PRN) to find KB entities near the labeled topic entities in the question, which helps extract a relatively small question-relevant subgraph containing answer entities. Since the inference process involves inverse relations, we also add the inverse of a fact triple. For example, given a triple $(e_1, r, e_2)$, we add the inverse triple $(e_2, r^{-1}, e_1)$, where $r^{-1}$ is the inverse of relation $r$. Throughout our experiments, we apply 300 dimensional pre-trained GloVe word embeddings and set the dimension of KG embeddings (i.e. entity embeddings and relation embeddings) to 200. The KG embeddings are assigned with pre-trained ones, which are learned under the constraint following TransE (Bordes et al., 2013). Moreover, we use a two-layer unidirectional LSTM with a hidden state dimension of 200 as the decision history encoder. For the question encoder, we use a Transformer with 2 layers and 4 heads. For REINFORCE algorithm, the discount factor $\eta$ is set 0.95. In addition, we use GPT-3.5-turbo API as the LLM for our model. Here, we utilize the Hits@1 score to assess model performance, indicating the accuracy of the top one among predicted answer entities.

### 4.4 Main Results

Table 1 shows the experimental results in four datasets. From the results, our method achieves promising performance on all datasets. Specifically, when faced with challenging datastes such as CWQ and WebQSP, our model still demonstrates impressive performance compared to other methods, attributed to the proposed collaborative reasoning framework powered by RL and LLMs. Furthermore, we observe that many methods on MetaQA exhibit good performance since the dataset is relatively simple and only focus on the movie domain. Compared with RL-based models ($e.g.$, SRN, ARL and ARN), our approach performs well overall. In order to enhance the interpretability of the model, our method is based on a RL framework. However, RL-based methods usually face challenges of aimless exploration and low-quality rewards. We use LLMs to assist in enhancing the RL agent, which significantly improves the performance of our model. Moreover, our methods exhibit superior performance, particularly on CWQ. This advantage stems from our approach which formalizes the KGQA task as a hierarchical decision-making process, effectively addressing complex questions with constraints.

As for LLM-based methods, we notice that directly using LLMs ($e.g.$, ChatGPT and Llama-2-70B) performs not well on the complex datasets, such as CWQ, MetaQA-2H and MetaQA-3H. It indicates that relying solely on LLMs is challenging for effectively solving the complex KGQA task. Therefore, some methods incorporate external knowledge graphs to enhance LLMs in addressing complex questions ($e.g.$, KD-CoT, ToG and StructGPT). Although these methods demonstrate positive outcomes, our approach makes even greater advancements, achieving a $3.3\%$ improvement on WebQSP and $9.3\%$ improvement on more complex CWQ compared to the best one. This is because that we propose a collaborative reasoning framework to mimic human cognitive processes. In the inference stage, LLMs use prior knowledge for intuitive assessments, while the trained policy-based agents provide logical reasoning as validation, helping LLMs relieve hallucinations.

## 4.5 Further Analysis

### 4.5.1 Ablation Study

We conduct various ablation studies to verify the effectiveness of different factors in CRF. The ablation studies are carried out on two datasets, PQ and CWQ. The ablation results are shown in Table 2.

**w/o LLMs as reward function.** In the high-level process, we remove the intermediate rewards generated by LLMs. The ablated model exhibits poorer performance than the original one, indicating that employing LLMs as the reward function helps mitigate low-quality reward challenges.

**w/o LLMs to guide low-level policy.** Eliminating the use of LLMs to guide the low-level policy indicates that our approach only relies on a RL framework to solve the KGQA task. The lack of prior knowledge provided by LLMs poses a challenge of aimless exploration for the agents when tackling complex questions that require long-term reasoning. The significant performance decline of the ablated model on the more complex CWQ highlights the critical role of LLMs.

**w/o reinforcement learning.** When we remove reinforcement learning, we can find that the performance gap between the ablated model and the original model increases as the dataset becomes more complex. This is because there is no RL-based agent to provide reliable logical reasoning as verification to eliminate the illusions of LLMs. As questions increase in complexity, there is a higher likelihood of LLMs generating hallucinations, which results in a decline in model performance.

**w/o hierarchical policy structure.** Without hierarchical policy structure, the RL-based agent can only iteratively select actions to execute path reasoning and are unable to account for constraint conditions, leading to reduce model performance in handling complex questions with constraints.

Table 2: Ablation study results (%Hits@1) on PQ and CWQ. Best results are marked **bold**.

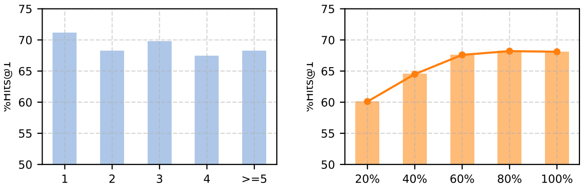
| Model | PathQuestion | CWQ |
|---|---|---|
| | Mix | Mix |
| CRF(full model) | **97.1** | **68.2** |
| w/o LLMs as reward function | 96.8 | 66.7 |
| w/o LLMs to guide low-level policy | 94.9 | 52.3 |
| w/o reinforcement learning | 96.4 | 58.6 |
| w/o hierarchical policy structure | 96.5 | 65.5 |

### 4.5.2 Stability Study

We define the complexity of the question as the number of options chosen in the reasoning process. The stability study is conducted on CWQ with various complex questions. As shown in Figure 2a, it is evident that our model maintains consistent performance even with the increasing complexity of questions. The stability demonstrates the effectiveness and robustness of CRF at reasoning over KGs for complex questions with constraints.

### 4.5.3 Few-shot Study

Figure 2b shows the performance of CRF on different proportions of CWQ training data. We observe that our method still performs well even with only 20% of the training data used. And our method can achieve essentially the same effect using only 60% of the data as using all the training data. The results show that our method can achieve good performance in few-shot situations. This is because that the common sense priors and planning capabilities of LLMs can be injected into the proposed model to enable it to effectively answer complex questions.



(a) Stability Study

(b) Few-shot Study

Figure 2: (a) Performance on CWQ with different complexity. (b) Performance on different proportions of CWQ training data.

In addition, we present a case study to show our CRF model in Appendix D.

## 5 Conclusion

In this paper, we introduce a collaborative reasoning framework powered by hierarchical RL and LLMs to mimic human cognitive processes. The proposed model leverages the common sense priors contained in LLMs while utilizing RL to provide learning from the environment, resulting in a hierarchical agent that uses LLMs to solve the complex KGQA task. The high-level agent accurately identifies constraints encountered during reasoning, while the low-level agent conducts efficient path reasoning by selecting the most promising relations in KG. Extensive experiments conducted on four benchmark datasets clearly demonstrate the effectiveness of the proposed model, which surpasses state-of-the-art approaches.

## 6 Limitations

In our work, we primarily use the frozen LLM (ChatGPT), whose capabilities may be limited by its pretraining. In the future, it would be worthwhile to explore how fine-tuning LLMs can more effectively guide the reasoning of RL-based agents and improve the accuracy of intermediate rewards provided by LLMs. Additionally, we assume that the given questions contain topic entities, enabling reasoning within the knowledge graph to obtain answers. Consequently, the questions lacking entities cannot be answered through reasoning. For this case, we rely on the LLM to directly generate the answers.

## References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

Kurt D. Bollacker, Colin Evans, Praveen K. Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: a collaboratively created graph database for structuring human knowledge. In *Proceedings of the ACM SIGMOD International Conference on Management of Data, SIGMOD 2008, Vancouver, BC, Canada, June 10-12, 2008*, pages 1247–1250. ACM.

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

Zi-Yuan Chen, Chih-Hung Chang, Yi-Pei Chen, Jijnasa Nayak, and Lun-Wei Ku. 2019. Uhop: An unrestricted-hop relation extraction framework for knowledge-based question answering. In *Proceedings of NAACL-HLT*, pages 345–356.

Hai Cui, Tao Peng, Feng Xiao, Jiayu Han, Ridong Han, and Lu Liu. 2023. Incorporating anticipation embedding into reinforcement learning framework for multi-hop knowledge graph question answering. *Inf. Sci.*, 619:745–761.

Jiale Han, Bo Cheng, and Xizhou Wang. 2020. Two-phase hypergraph based reasoning with dynamic relations for multi-hop kbqa. In *IJCAI*, pages 3615–3621.

Gaole He, Yunshi Lan, Jing Jiang, Wayne Xin Zhao, and Ji-Rong Wen. 2021. Improving multi-hop knowledge base question answering by learning intermediate supervision signals. In *Proceedings of the 14th ACM international conference on web search and data mining*, pages 553–561.

Jinhao Jiang, Kun Zhou, Zican Dong, Keming Ye, Wayne Xin Zhao, and Ji-Rong Wen. 2023. Structgpt: A general framework for large language model to reason over structured data. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 9237–9251.

Daniel Kahneman. 2011. *Thinking, fast and slow*. macmillan.

Tianle Li, Xueguang Ma, Alex Zhuang, Yu Gu, Yu Su, and Wenhu Chen. 2023. Few-shot in-context learning on knowledge base question answering. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6966–6980.

Haoran Luo, Zichen Tang, Shiyao Peng, Yikai Guo, Wentai Zhang, Chenghao Ma, Guanting Dong, Meina Song, Wei Lin, et al. 2023. Chatkbqa: A generate-then-retrieve framework for knowledge base question answering with fine-tuned large language models. *arXiv preprint arXiv:2310.08975*.

Xin Lv, Xu Han, Lei Hou, Juanzi Li, Zhiyuan Liu, Wei Zhang, Yichi Zhang, Hao Kong, and Suhui Wu. 2020. Dynamic anticipation and completion for multi-hop reasoning over sparse knowledge graph. In *EMNLP*, pages 5694–5703.

Yunqi Qiu, Yuanzhuo Wang, Xiaolong Jin, and Kun Zhang. 2020. Stepwise reasoning for multi-relation question answering over knowledge graph with weak supervision. In *WSDM*, pages 474–482. ACM.

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

Jiaxin Shi, Shulin Cao, Lei Hou, Juanzi Li, and Hanwang Zhang. 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*, pages 4149–4158.

Jiashuo Sun, Chengjin Xu, Lumingyuan Tang, Saizhuo Wang, Chen Lin, Yeyun Gong, Lionel M. Ni, Heung-Yeung Shum, and Jian Guo. 2023. Think-on-graph: Deep and responsible reasoning of large language model on knowledge graph. *Preprint*, arXiv:2307.07697.

Alon Talmor and Jonathan Berant. 2018. The web as a knowledge-base for answering complex questions. In *NAACL-HLT*, pages 641–651.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton-Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurélien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. 2023a. Llama 2: Open foundation and fine-tuned chat models. *CoRR*, abs/2307.09288.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023b. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Keheng Wang, Feiyu Duan, Sirui Wang, Peiguang Li, Yunsen Xian, Chuantao Yin, Wenge Rong, and Zhang Xiong. 2023. Knowledge-driven cot: Exploring faithful reasoning in llms for knowledge-intensive question answering. *Preprint*, arXiv:2308.13259.

Ronald J Williams. 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8:229–256.

Hongbin Ye, Tong Liu, Aijia Zhang, Wei Hua, and Weiqiang Jia. 2023. Cognitive mirage: A review of hallucinations in large language models. *arXiv preprint arXiv:2309.06794*.

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

Qixuan Zhang, Xinyi Weng, Guangyou Zhou, Yi Zhang, and Jimmy Xiangji Huang. 2022. ARL: an adaptive reinforcement learning framework for complex question answering over knowledge base. *Inf. Process. Manag.*, 59(3):102933.

Yuyu Zhang, Hanjun Dai, Zornitsa Kozareva, Alexander Smola, and Le Song. 2018. Variational reasoning for question answering with knowledge graph. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32.

Mantong Zhou, Minlie Huang, and Xiaoyan Zhu. 2018. An interpretable reasoning network for multi-relation question answering. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 2010–2022.

Anjie Zhu, Deqiang Ouyang, Shuang Liang, and Jie Shao. 2022. Step by step: A hierarchical framework for multi-hop knowledge graph reasoning with reinforcement learning. *Knowledge-Based Systems*, 248:108843.

## A Datasets

In order to evaluate the effectiveness of the proposed CRF method, we conduct experiments using four public datasets, including WebQuestionSP(WebQSP) (Yih et al., 2016), ComplexWebQuestions(CWQ) (Talmor and Berant, 2018), PathQuestion(PQ)(Zhou et al., 2018) and MetaQA (Zhang et al., 2018). Table 3 shows the statistics of the four datasets.

Table 3: Statistics of the experiment datasets.

| Datasets | | KG | Train | Valid | Test |
|---|---|---|---|---|---|
| WebQSP | Mix | Freebase | 2848 | 250 | 1639 |
| CWQ | Mix | Freebase | 27639 | 3519 | 3531 |
| PQ | 2H | Freebase | 1528 | 189 | 191 |
| | 3H | Freebase | 4163 | 515 | 520 |
| | Mix | Freebase | 5691 | 704 | 711 |
| MetaQA | 1H | OMDb | 96106 | 9992 | 9947 |
| | 2H | OMDb | 118980 | 14872 | 14872 |
| | 3H | OMDb | 114196 | 14274 | 14274 |

**WebQuestionSP(WebQSP)** (Yih et al., 2016) contains 4373 questions, where the answer entities are within a maximum of 2 hops from the topic entity on the Freebase (Bollacker et al., 2008).

**ComplexWebQuestions(CWQ)** (Talmor and Berant, 2018) is constructed based on WebQSP, which is more complex. It extends the question entities or adds constraints to answers to construct complex questions consisting of four types and requiring up to 4-hops of reasoning based on Freebase.

**PathQuestion(PQ)** (Zhou et al., 2018) is from the general domain, and based on the subsets of Freebase. It extracts paths between two entities in the KG, and generated questions more real by some rules. PQ-2H and PQ-3H deNote 2-hop and 3-hop questions, respectively. PQ-Mix represents the mix of all questions.

**MetaQA** (Zhang et al., 2018) focuses on the movie domain, comprising over 400,000 questions. Based on the number of hops, the dataset includes three sets of question-answer pairs: 1-hop, 2-hop, and 3-hop.

## B Baselines

To comprehensively evaluate our approach, we select a series of following baseline models for comparison, which can be divided into three categories: (1) **IR-based methods**, including EmbedKGQA (Saxena et al., 2020), NSM (He et al., 2021), TransferNet (Shi et al., 2021); (2) **RL-based methods**, including SRN (Qiu et al., 2020), ARL (Zhang et al., 2022), ARN (Cui et al., 2023); (3) **LLMs-based methods**, including Llama-2-70B (Touvron et al., 2023a), ChatGPT, KD-CoT (Wang et al., 2023), ToG (Sun et al., 2023), StructGPT (Jiang et al., 2023). The elaborate descriptions of baselines are as follows.

**EmbedKGQA** (Saxena et al., 2020) embeds the question and KG triples into a vector space. Subsequently, a scoring function is employed to evaluate the candidate answer entities, with the highest-scoring entity as the predicted answer.

**NSM** (He et al., 2021) proposes a novel teacher-student framework where the student network focuses on answering queries, while the teacher network provides intermediate supervision to enhance the student's reasoning ability.

**TransferNet** (Shi et al., 2021) jointly manages labeled and textual relations, navigating between entities in several steps. At each step, it focuses on parts of the question, calculates scores for relations, and moves the scores of entities along those active relations smoothly.

**SRN** (Qiu et al., 2020) pioneers the formalization of complex KGQA as a sequential decision-making process grounded in reinforcement learning. Through a potential-based reward shaping strategy, SRN mitigates the challenges posed by delayed and sparse rewards.

**ARL** (Zhang et al., 2022) proposes a new adaptive reinforcement learning framework and introduces three atomic operations to adaptively extend the relation paths.

**ARN** (Cui et al., 2023) incorporates KG embeddings as anticipation information into RL framework to capture the potential target information for multihop reasoning. Moreover, a KEQA framework is designed to assign soft rewards for the RL agent.

**ChatGPT** is large language model developed by OpenAI. We can use their provided APIs to access them and solve KGQA tasks.

**Llama-2-70B** (Touvron et al., 2023a) is the large language model developed by Meta. We can use their provided APIs to solve KGQA tasks.

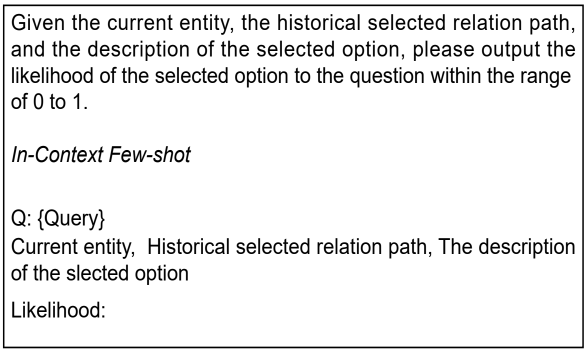
**KD-CoT** (Wang et al., 2023) proposes an interactive framework that utilizes a QA system to access external knowledge and provide high-quality answers to LLMs for solving knowledge-intensive KBQA tasks.

**ToG** (Sun et al., 2023) enables the LLM agent to iteratively execute beam search on KG, discover the most promising reasoning paths, and return the most likely reasoning results.

**StructGPT** (Jiang et al., 2023) gathers relevant evidence from structured data, allowing LLMs to focus on the reasoning task using the acquired information.

## C Prompt for Reward Function

Figure 3 shows the detail of the prompt $\rho$.



Given the current entity, the historical selected relation path, and the description of the selected option, please output the likelihood of the selected option to the question within the range of 0 to 1.

*In-Context Few-shot*

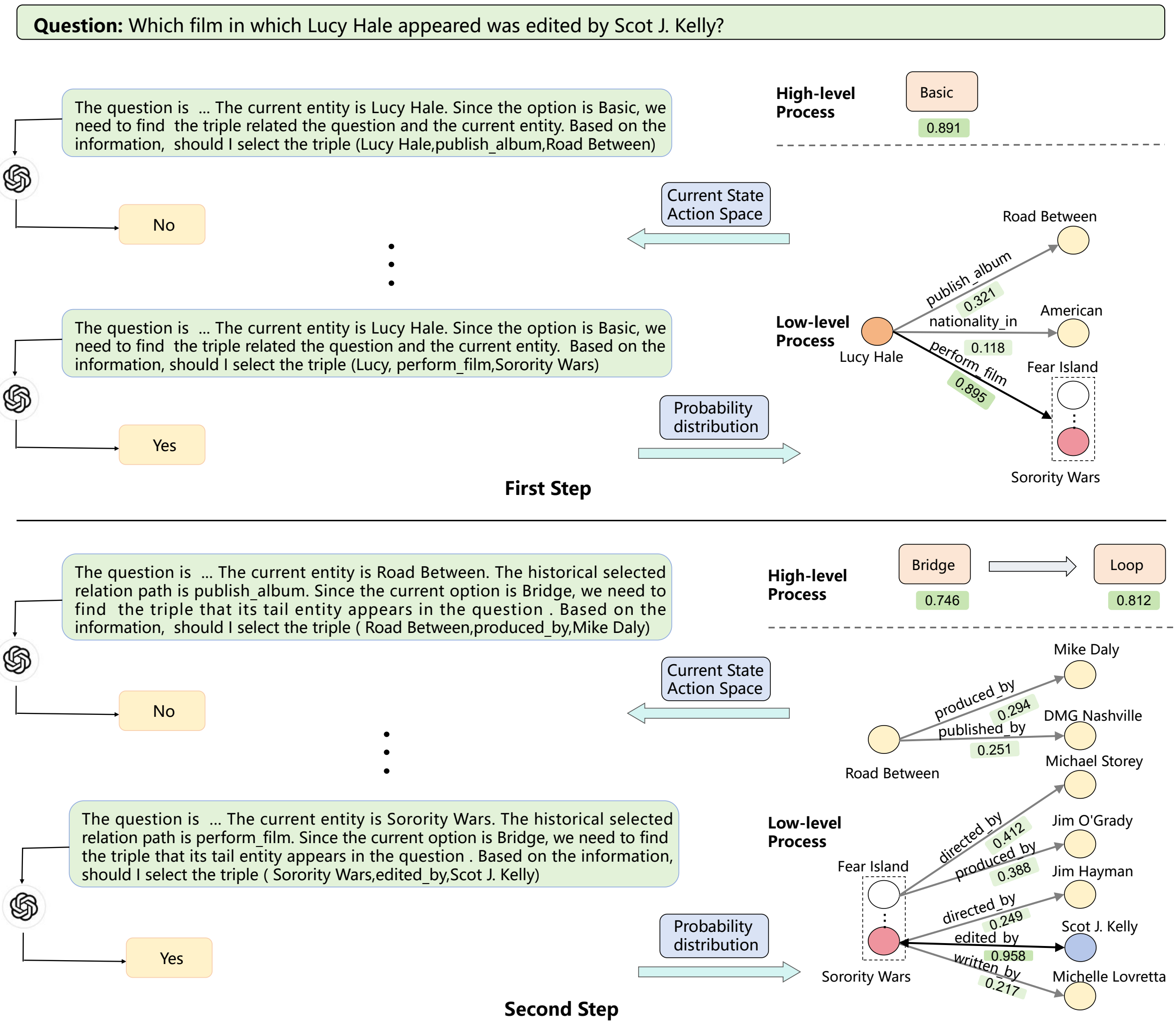
Q: {Query}
Current entity, Historical selected relation path, The description of the slected option

Likelihood:

Figure 3: Prompt for Reward Function

## D Case Study

As shown in Figure 4, we present a case study to make a better understanding how our proposed CRF model works. Given a complex question "*Which film in which Lucy Hale appeared was edited by Scot J. Kelly?*", the topic entity is *Lucy Hale* and the entity constraint is *Scot J. Kelly*. To solve the above question, two-hop reasoning is required with a constraint. In the first step, the high-level process selects the "Basic" option and transfers it to the low-level process. Next, we compose the input prompt for LLMs based on the current state and action space of the low-level and generate the probability distribution of the action space. Finally, we select the top-3 actions, i.e., *perform_film*,*publish_album* and *nationality_in*, by combining the low-level RL policy. In the second step, the option "Bridge" is selected to direct the low-level process in taking actions for executing path reasoning concerning the entity constraint. Concretely, the agent reaches *Sorority Wars*. Since the option "Loop" is chosen in the subsequent step, indicating the termination of the reasoning process, the entity *Sorority Wars* is considered as the predicted answer.



**Question:** Which film in which Lucy Hale appeared was edited by Scot J. Kelly?

The question is ... The current entity is Lucy Hale. Since the option is Basic, we need to find the triple related the question and the current entity. Based on the information, should I select the triple (Lucy Hale,publish_album,Road Between)

No

The question is ... The current entity is Lucy Hale. Since the option is Basic, we need to find the triple related the question and the current entity. Based on the information, should I select the triple (Lucy, perform_film,Sorority Wars)

Yes

The question is ... The current entity is Road Between. The historical selected relation path is publish_album. Since the current option is Bridge, we need to find the triple that its tail entity appears in the question . Based on the information, should I select the triple ( Road Between,produced_by,Mike Daly)

No

The question is ... The current entity is Sorority Wars. The historical selected relation path is perform_film. Since the current option is Bridge, we need to find the triple that its tail entity appears in the question . Based on the information, should I select the triple ( Sorority Wars,edited_by,Scot J. Kelly)

Yes

Figure 4: A case of the hierarchical decision process.