LLM-ARK: Knowledge Graph Reasoning Using Large Language Models via Deep Reinforcement Learning

Yuxuan Huang
JiLin University
huangyx21@mails.jlu.edu.cn

December 18, 2023

Abstract

With the evolution of pre-training methods, large language models (LLMs) have exhibited exemplary reasoning capabilities via prompt engineering. However, the absence of Knowledge Graph (KG) environment awareness and the challenge of engineering viable optimization mechanisms for intermediary reasoning processes, constrict the performance of LLMs on KG reasoning tasks compared to smaller models. We introduce LLM-ARK, a LLM grounded KG reasoning agent designed to deliver precise and adaptable predictions on KG paths. LLM-ARK utilizes Full Textual Environment (FTE) prompts to assimilate state information for each step-sized intelligence. Leveraging LLMs to richly encode and represent various types of inputs and integrate the knowledge graph further with path environment data, before making the final decision. Reframing the Knowledge Graph (KG) multi-hop inference problem as a sequential decision-making issue, we optimize our model using the Proximal Policy Optimization (PPO) online policy gradient reinforcement learning algorithm which allows the model to learn from a vast array of reward signals across diverse tasks and environments. We evaluate state-of-the-art LLM(GPT-4) and our method which using open-source models of varying sizes on OpenDialKG dataset. Our experiment shows that LLaMA7B-ARK provides excellent results with a performance rate of 48.75% for the target@1 evaluation metric, far exceeding the current state-of-the-art model by 17.64 percentage points. Meanwhile, GPT-4 accomplished a score of only 14.91%, further highlighting the efficacy and complexity of our methodology. Our code is available on GitHub[1] for further access.

Keywords KG · LLM · DRL

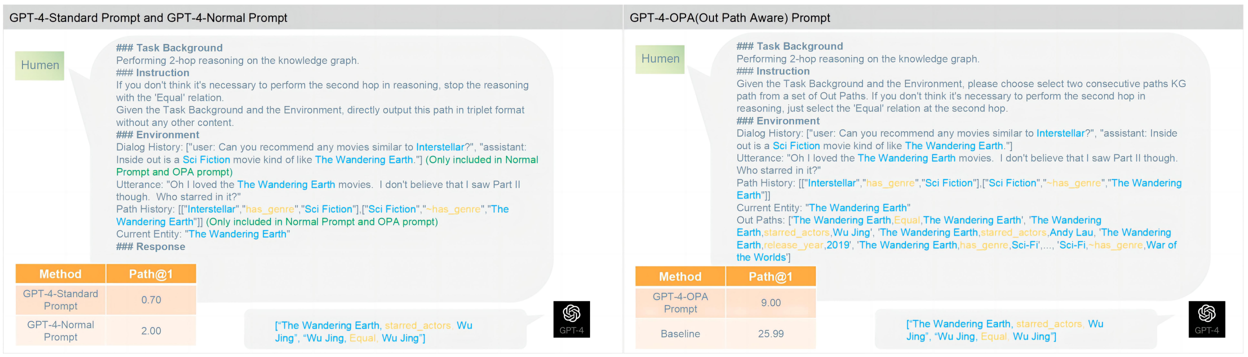
1 Introduction

With significant progress in large-scale language models (LLMs), researchers have recognized their superior capabilities in the field of natural language processing (NLP)[Liu et al., 2023, Shakarian et al., 2023, Lai et al., 2023]. Reasoning ability stands as the most demonstrative of AI intelligence. Recently, to boost the performance of LLMs in reasoning tasks, we noted various optimization strategies adopted by researchers such as Chain of Thought (COT)[Wei et al., 2023] and decomposing subtasks[Kazemi et al., 2023]. At the present time, current LLMs reasoning methodologies have seen limited studies in the KG reasoning task. This research endeavor is aimed at addressing this shortfall in the area and contributing to the large-model KG reasoning task.

Knowledge Graphs composed of vertices or entities connected by edges or relations, gaining popularity in knowledge-based systems for its structured disposition. Previous work[Moon et al., 2019, Zhang et al., 2020, Ni et al., 2022, Tuan et al., 2022] mainly relied on supervised learning methods early on. To assess the capabilities of current state-of-the-art (SOTA) LLM: GPT4[OpenAI, 2023], we initially examine the proficiency of LLMs on KG reasoning, as illustrated in Figure 1, determining their potential application in the KG domain. Empirical studies reveal that, despite demonstrating reasonable performance on KG tasks, indicative of their proficiency in managing complex problems, understanding

[1]https://github.com/Aipura/LLM-ARK

contextual relationships, and utilizing pre-training knowledge, LLMs still present issues and fall short when compared with state-of-the-art models. There are two main challenges applying LLM-based agents. On the one hand, LLMs


GPT-4-Standard Prompt and GPT-4-Normal Prompt

Humen

```
### Task Background
Performing 2-hop reasoning on the knowledge graph.
### Instruction
If you don't think it's necessary to perform the second hop in reasoning, stop the reasoning
with the 'Equal' relation.
Given the Task Background and the Environment, directly output this path in triplet format
without any other content.
### Environment
Dialog History: ["user: Can you recommend any movies similar to Interstellar?", "assistant:
Inside out is a Sci Fiction movie kind of like The Wandering Earth."] (Only included in Normal
Prompt and OPA prompt)
Utterance: "Oh I loved the The Wandering Earth movies.  I don't believe that I saw Part II
though.  Who starred in it?"
Path History: [["Interstellar","has_genre","Sci Fiction"],["Sci Fiction","~has_genre","The
Wandering Earth"]] (Only included in Normal Prompt and OPA prompt)
Current Entity: "The Wandering Earth"
### Response
```

Method	Path@1
GPT-4-Standard Prompt	0.70
GPT-4-Normal Prompt	2.00

GPT-4

["The Wandering Earth, starred_actors, Wu Jing", "Wu Jing, Equal, Wu Jing"]

GPT-4-OPA(Out Path Aware) Prompt

Humen

```
### Task Background
Performing 2-hop reasoning on the knowledge graph.
### Instruction
Given the Task Background and the Environment, please choose select two consecutive paths KG
path from a set of Out Paths. If you don't think it's necessary to perform the second hop in
reasoning, just select the 'Equal' relation at the second hop.
### Environment
Dialog History: ["user: Can you recommend any movies similar to Interstellar?", "assistant: Inside
out is a Sci Fiction movie kind of like The Wandering Earth."]
Utterance: "Oh I loved the The Wandering Earth movies.  I don't believe that I saw Part II though.
Who starred in it?"
Path History: [["Interstellar","has_genre","Sci Fiction"],["Sci Fiction","~has_genre","The Wandering
Earth"]]
Current Entity: "The Wandering Earth"
Out Paths: ['The Wandering Earth,Equal,The Wandering Earth', 'The Wandering
Earth,starred_actors,Wu Jing', 'The Wandering Earth,starred_actors,Andy Lau, 'The Wandering
Earth,release_year,2019', 'The Wandering Earth,has_genre,Sci-Fi',..., 'Sci-Fi,~has_genre,War of
the Worlds']
```

Method	Path@1
GPT-4-OPA Prompt	9.00
Baseline	25.99

GPT-4

["The Wandering Earth, starred_actors, Wu Jing", "Wu Jing, Equal, Wu Jing"]

Figure 1: Manual prompts on the OpenDialKG dataset. Compare to GPT-4-Standard, GPT-4-Normal has more awareness of dialog context and path history, while GPT-4-OPA has more awareness of 2-hop exit path subgraphs compared to the former two. The experimental results show that the more environmental information GPT-4 perceives, the higher the knowledge graph reasoning path@1 evaluation metric score.

suffer from a limited perception of variable reasoning environments. The alignment between LLMs' knowledge and the environment can be wrong and limit functional competence due to lack of grounding[Carta et al., 2023]. If properly grounded, the model's structure would be both simplified and effective. For KG reasoning tasks, as shown in Figure 1, the agent achieved better scores when provided with as much information about the environment as possible, such as dialog history, inference path history, and all exit paths. Although LLMs are not designed to handle tool use or take actions, Peng et al. [2023], Carta et al. [2023], Du et al. [2023] found that it can be achieved good results in downstream decision making by simply feeding full textual representations as inputs to LLMs.

On the other hand, Yao et al. [2023] indicate that there is a lack of systematic methodologies for consistent model refinement. In other word, LLMs lack the necessary mechanisms for optimizing intermediate reasoning processes in multi-hop reasoning tasks. Manual prompt tuning is still widely used in a lot of the application scenarios. LLM itself is stateless. It has been observed that LLM based agents are easy to run into infinite loops if the states are not handled nicely. Inevitably this methodology runs into the prompt length issues. As the trajectory grows longer, the prompt runs out of spaces. In addition, some restrictions are explicitly specified in the prompt, it is less realistic in knowledge graph environments where a single entity may have more than 100 exit edges. As for supervised fine-tuning, most of the well-performing LLMs are too large to be fit in just one or two GPUs. Most of the issues are caused by the fact that LLMs are not pre-trained or designed with an action-agent application in mind.

Finally, we introduce **LLM-ARK**, a simple yet effective framework that leverage **L**arge **L**anguage **M**odel as **A**gent to perform **R**easoning on **K**nowledge Graphs, as depicted in Figure 3.1. We employ LLMs as agent and express the Large model KG inference task as a reinforcement learning sequential decision-making problem, and using a Full-Textual-Environment prompt to aggregate multiscale inputs, we employ LLMs to obtain a richly informative representation of state embeddings. Moreover, our agent architecture does not necessitate access to LLM parameters or gradient propagation through it. Instead, we adopt a policy gradient approach where the Actor LLM functions as part of the environment. This setup allows the model to learn from a vast array of reward signals across diverse tasks and environments. In summary, our contributions are as follows:

- We assess the capabilities of state-of-the-art LLM: GPT4, on large-scale KG inference datasets and analyze the experimental results in detail to understand the causes of their inferior performance.
- To enhance the performance of the LLM agents, we introduce LLM-ARK. Our method expresses the KG dialog inference problem as a reinforcement learning sequential decision-making issue, using a Full-Textual-Environment prompt to aggregate multiscale inputs, dual-environment sensing on the state and decision side and leverage LLMs to explore on KGs.
- Furthermore, we update only the parameters of the actors that are part of the LLM agent using the policy gradient method, without accessing the LLM parameters or propagating gradients through them. This approach enables learning from diverse reward signals during interactions with the environment.

2 Related Work

2.1 KG Reasoning on Dialog Systems && LLMs

KG, due to its structured nature, is an external information source gaining increased popularity in knowledge-based systems. Jung et al. [2020] put forth a model for navigating the conditional graph of a conversation within a KG dialogue system. Leveraging a bidirectional attention flow, the model maximizes the use of structured information from the KG and flexibly determines the extension range of nodes and edges. Zhang et al. [2020] employed a similar approach, using graph attention to navigate the conceptual space of a commonsense KG. Other models, such as Xu et al. [2020], utilize the KG as an external source for controlling coarse-level discourse generation. These innovative uses of KGs facilitate meaningful dialogue generation and nurture topic-focused conversations supported by common sense knowledge. Moon et al. [2019] developed a retrieval system designed to generate responses based on a graph reasoning task. They employed a graph walker to navigate the graph, propelled by the symbol transformation conditions of the dialog context. Expanding on this, Ni et al. [2022] introduced a hierarchical reinforcement learning KG inference model that aggregates multiple inputs utilizing an attention mechanism. This approach instructs the model to reason in one step and then fine-tunes it using a goal-directed approach. Tuan et al. [2022] employed a similar approach, but with a single transformer model that walks directly over large-scale KGs, reasoning over fine-tunable KGs to generate responses. Similarly, Luo et al. [2023] initially create relational paths derived from KGs as high-confidence plans, which are later utilized to extract valid reasoning paths from KGs for confident reasoning.

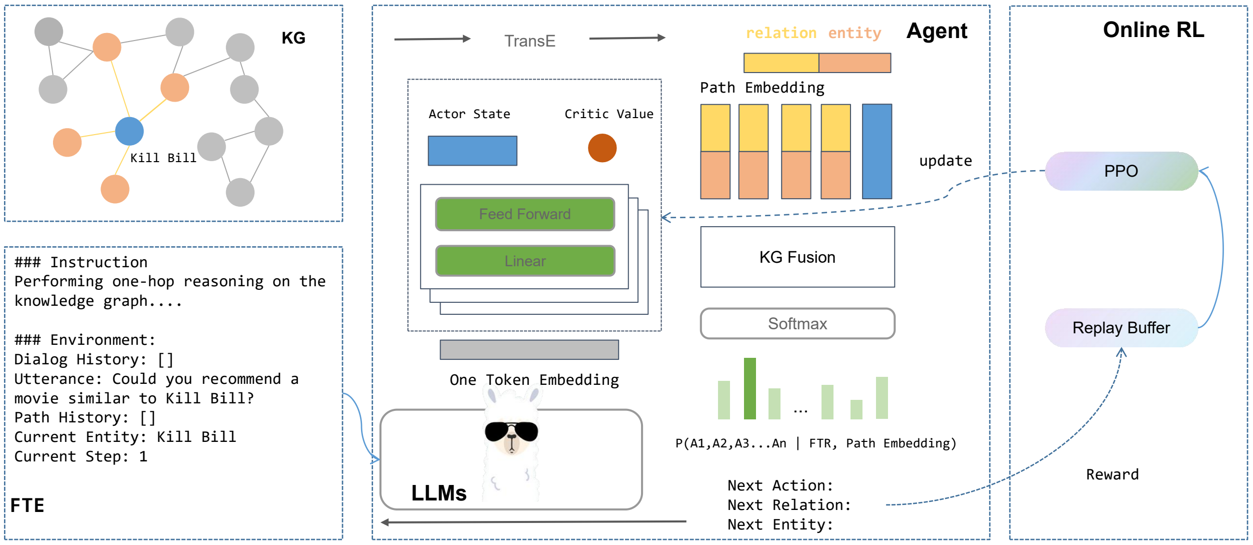
2.2 LLMs with Deep Reinforcement Learning

The framework carries out exploration and reasoning by pinpointing entities pertinent to a given problem and extracting relevant triples from external knowledge bases. Expanding the scope of the research, Carta et al. [2023] examined large models' interaction with physical environments, using text to describe these environments, and enhanced them using online reinforcement learning. Yao et al. [2023] employed a special RLHF technique to tailor the model to human preferences, generating beneficial, non-toxic, and safe data for training while also training reward models to evaluate LLMs. Retroformer, a significant improvement over Chain of Thought (COT), is primarily applied to reasoning tasks and uses a unique RLHF methodology. Huang et al. [2022] consistently integrate feedback from diverse sources into the planning language cues of the LLM, thereby enabling it to reason and replan to solve complex problems in both simulated and real-world environments. Zhu et al. [2023] developed robots for open-world environments with generalized capabilities through a large language model grounded in textual knowledge and memory. The LLM utilizes knowledge and memory-based strategies to execute a multilevel disassembly by first breaking down the goal into subgoals via the LLM Decomposer. Then, it breaks down the subgoals into structured actions using the LLM Planner, and finally decomposes the structured actions into a series of actual actions using the LLM Interface. Singh et al. [2022] incorporate Large Language Models to propose a procedural LLM hint structure that facilitates plan generation functionality in contextual environments, robot capabilities, and tasks. In contrast, Shinn et al. [2023] introduce a novel framework, Reflexion, that strengthens linguistic agents through linguistic feedback, rather than updating weights. The Reflexion agent verbally reflects on task feedback signals, and then stores its reflection text in an episodic memory buffer to make better decisions in subsequent trials.

3 Methods

3.1 Overview

As shown in figure 3.1, our model has the following main components, FTE(Full-Textual-Environment), LLM(Large Language Model) and RL(Reinforcement Learning). FTE can be seen as state manager, using a Full-Textual-Environment prompt to aggregate multi-scale inputs, updating and maintaining state transfers between itself and the environment, employs a text dictionary format. At first LLMs obtain a richly informative representation of state embeddings. To capture the path embedding information of the KG, we pre-train the KG on TransE[Fan et al., 2014]. Instead of feeding in the probability distribution of the action space, our Actor introduces the probability distribution that syncs with the path embedding dimension. Rather than directly introducing the probability distribution of the action space, our Actor feeds the probability distribution along with the path embedding, subsequently eliminating invalid paths (we utilise 'Pad' for this adaptation process) before outputting a precise and legitimate action. We formulate the large model KG inference task within an online reinforcement learning framework and continuously optimize the decision network based on the collected experience. Finally, we refine the adapter using the Proximal Policy Optimization (PPO) reinforcement learning method, which is based on the obtained reward information.



```
### Instruction
Performing one-hop reasoning on the
knowledge graph....

### Environment:
Dialog History: []
Utterance: Could you recommend a
movie similar to Kill Bill?
Path History: []
Current Entity: Kill Bill
Current Step: 1
```

```
Next Action:
Next Relation:
Next Entity:
```

Figure 2: The overall architecture of LLM-ARK.

3.2 Prompt Engineering

As illustrated in Figure 1, we detail the prompting schemes, encompassing the normal prompt and out path aware prompt. To guide LLMs in performing specific dialogue tasks, we can formulate the normal prompt scheme as:

$$p(r|D, C) \tag{1}$$

Given the task background D and the conversation history C, instruct the LLM to generate the response r. More complex path aware prompt aims to provide alternative options for LLMs to decide what kinds of actions should be taken in the response, instead of simply responding to the instruction. It can be formulated as:

$$p(a, r|D, C, A) \tag{2}$$

Given the task background D, conversation history C, and a set of potential dialogue acts A, the LLM is guided to select the most appropriate dialogue act $a \in A$, which then generates the response r.

3.3 LLM-ARK

Knowledge Graphs (KGs) are structured knowledge networks composed of vertices, interpreted as entities, associated via edges or relationships. Let $\mathcal{E}$ stand for a collection of entities and $\mathcal{R}$ for a collection of relations. We represent the external KG as $\mathcal{G}$ = {V,E,$\mathcal{R}$}, where V and E denote the vertices and edges of the graph, respectively. Note that V = $\mathcal{E}$ and E $\subseteq V \times \mathcal{R} \times$ V. Let v denote a node and e denote an edge in $\mathcal{G}$. Given dialog context X = and $\mathcal{G}$, we can identify an entity in the KG(e.g., an entity name The Wondering Earth) and and represent it as $v_s, v_s \in V$. The goal is to select a proper edge e_t at the t-th timestamp for one-hop reasoning.

Graph attention-based models require significant annotation effort since all potential paths must be evaluated, which can be computationally expensive for large Knowledge Bases (KBs) with millions of entities. To overcome this challenge, our study employs a policy gradient model that efficiently traverses the KG to select relationships and ultimately achieves the target, demonstrating proficiency in multi-hop reasoning.

KG reasoning naturally reduces to a finite horizon, deterministic partially observed Markov decision process that lie on a KG $\mathcal{G}$. We formulate KG reasoning as a Markov Decision Process (MDP) described by a five-tuple (S, O, A, T, R, γ):

- State. $S_t \in \mathcal{S}$ is an infinite set of environment states, which encode information stored in Working Memory, including task background tb, user query q, dialog history h, current entity v_c, path history ph, current step k, The normal state is represented using a six-tuple: S = (tb, q, h, v_c, ph, k).
- Observation. The complete state of the environment can be observed. Formally, the observation function $\mathcal{O} = S$.

- Action. The set of possible actions $\mathcal{A}_{Sm}$ from a state S_m consists of all outgoing edges of the vertex v_p in G. Formally $\mathcal{A}_S = \{e \in E\colon S\} \cup \{(s, \varnothing, s)\}$. This means that each state's agent can choose one of all output edges e which having the knowledge of the label of the edge $r_t, r_t \in \mathcal{R}$ and destination vertex v_t.
- Transition. Depending on the edge selected by the agent at time step t, the environment is changed deterministically by updating the state to the new vertex. For single turn dialogue, we update current entity, path history and step. Formally, the transition function : $\delta\colon \mathcal{S} \times \mathcal{A} \to \mathcal{S}$ is defined by $\delta(S, A) = (tb, q, h, v\_c^{'}, ph^{'}, k^{'})$, For multi-turn dialogue wo also need to update user query and dialogue history. $\delta(S, A) = (tb, q^{'}, h^{'}, v\_c^{'}, ph^{'}, k^{'})$, where $S = (tb, q, h, c, ph, k)$ and $A = (v_c, e_t))$.
- Reward. We have a final reward of +1 if the current entity is the target entity v_g and -1 otherwise. if $S_T = (tb, q^{'}, h^{'}, v\_c^{'}, ph^{'}, k^{'})$ is the final state, then we have a final reward of +1 if $v\_c^{'} = v_g$, else -1, i.e., $R(S_T) = I_f\{v_p = v_g\}$.
- γ denote reward discounts factor are used to compute the reward information of each intermediate process when agent reaches the goal, or the end of the maximum step T.

3.3.1 Full Textual Environment

This module (FTE) tracks the agent's state that captures all essential information in the conversation so far. The same as Prompt Engineering normal prompt.

3.3.2 Agent

Reference to previous work Carta et al. [2023], we use standard RL practices by adding action heads - a Multi-Layer Perceptron (MLP) with |A| outputs - on top of the LLM. Thus, we can use only pretrained operations from the LLM and leverage language modeling heads' prior, this method is robust to any action space and can thus be used on any textual environment with no change. We design a randomized nonstationary policy $\pi = (d_0, d_1, ..., d_{t-1})$, where $d_t = P(\mathcal{A}_{S_t})$ is a policy at time step t. Agent has two components, one of which is LLM and the other is PA-MLP.

LLM We initially utilize a LLM to encode the state $S_t = (S_{t-1}, A_{t-1}, V_t)$ into a continuous vector $s_t \in \mathbb{R}^{2d}$. As a rule of thumb, for BERT models the cls token is used to represent the semantics of the whole sequence, while standard transformers and GPT-like LLMs use the embedding of the last token. We used the model on huggingface hub as well as the code to get the sequence vector representation[2]. It is defined by:

$$s = LLM(FTE) \tag{3}$$

PA-MLP

Instead of just adding an MLP with a single output for the value on top of the last layer of the first decoder block as in the conventional multicategorization task, to enhance the ability of the large model to perceive the environment, we further fused the hidden state after the MLP with the Knowledge Graph exit path information, called PA-MLP(**P**ath **A**ware MLP). Recall that each possible action represents an outgoing edge e with information about the edge relation label r_l and the target vertex/entity v_d. So the embedding for each $A \in A_{St}$ is $[r_l; v_d]$, and stacking the embeddings for all outgoing edges we get the matrix A_t. The network taking this as input is parameterized as a three-layer feed-forward network (MLP) with $\tanh$ nonlinearity, which takes the FTE representation FTE_t and the embedding for the outgoing paths embedding e_g and outputs a probability distribution over the possible actions from which a discrete action is sampled. Note that the dimension of the MLP output hidden state must be equal to the dimension of the path embedding. Finally, formulated as:

$$\begin{aligned} \mathbf{h_t} &= \mathbf{A_t}\left(\mathbf{W_3}\tanh\left(\mathbf{W_2}\tanh\left(\mathbf{W_1}\left(s_t\right)\right)\right)\right) \\ a_t &\sim \text{ Categorical }\left(\operatorname{softmax}\left(\mathbf{h_t}\right)\right) \end{aligned} \tag{4}$$

3.3.3 Training

1. **Optimizer** We propose leveraging the experiences accumulated by LLM-ARK to execute KG reasoning. More formally, for the above policy network (π_θ), we want to find the parameter θ that maximizes the reward.

$$\begin{aligned} J(\theta) = &\mathbb{E}_{(e_s, \mathcal{P}, e_g) \sim D} \mathbb{E}_{A_1, \ldots, A_{T-1} \sim \pi_\theta} \\ &\left[R\left(S_T\right) \mid S_1 = (FTE_1)\right], \end{aligned} \tag{5}$$

[2]https://huggingface.co

where we assume that there is a true underlying distribution $(e1, r, e2) \sim \mathcal{P}$. To address this optimization challenge, we adopt an online reinforcement learning policy gradient algorithm, Proximal Policy Optimization (PPO). PPO is a family of policy optimization methods that use multiple epochs of stochastic gradient ascent to perform each policy update. These methods have the stability and reliability of trust-region methods[Schulman et al., 2017]. For value approximation, we include a three-layer feed-forward network with a single output for the value, given by:

$$\mathbf{V} = \mathbf{W_3}\left(\tanh\left(\mathbf{W_2}\tanh\left(\mathbf{W_1}\left(s_t\right)\right)\right)\right) \tag{6}$$

Significantly, the LLM remains frozen for both the actor and critic modules, with only the linear forward layer being trained.

2. **Replay Buffer** The replay buffer stores the triplets $rb = (c^e, s, l, a, s^{'}, done)$ of the reflection prompt, indicating the current entity, the current state, logits, the selected action, the next state, and whether the episode has ended. The reason for recording the current entity is that we need to get all exit paths of the current entity for further fusion 4.

4 Experiments and Results

4.1 Datasets

OpenDialKG is a publicly available parallel corpus, consisting of 91,000 dialogues, each of which is supplemented with paths that link knowledge graph (KG) entities and their relations. The aim of the corpus is to render human dialogues' implicit reasoning processes as explicit computer operations within the KG. The dataset is partitioned into a 70% training set, 15% validation set, and 15% test set, following the methodology described in Moon et al. [2019].

4.2 Baselines

We compared our results with these baseline models: Tri-LSTM, Seq2Seq, Seq2Path, DialKG Walker[Young et al., 2018, Moon et al., 2019], DiffKG, AttnFlow, AttnIO[Jung et al., 2020] and HiTKG[Ni et al., 2022]. HiTKG is a hierarchical transformer-based tool that uses diverse inputs to forecast knowledge graph paths. Our team is highly influenced by this research and has chosen HiTKG as a strong baseline. To evaluate the performance of the state-of-the-art LLMs on the knowledge graph inference task, we designed three prompts methods: GPT4-Standard, GPT4-Normal and GPT4-OPA. The difference between GPT4-Standard, GPT4-Normal is that GPT4-Normal has more awareness of dialog context and path history, while GPT4-OPA has more awareness of 2-hop exit path subgraphs compared to the former two. See appendix A.3 for full prompts.

4.3 Implement Details

The training was conducted on A40. Informed by prior research from Jung et al. [2020], Ni et al. [2022], we pre-trained the knowledge graph using TransE[Fan et al., 2014] based on this GitHub repository[3]. The objective was to unearth and explore entity relationships, expand the knowledge network for connection prediction, and enable diverse reward function design. To reduce GPU memory usage and increase the pace of training, all experiments - excluding LLaMA7B-ARK-FP32 were carried out with bfloat16 computational data type. We selected LLaMA-7B and LLaMA-13B, and since all true paths in OpenDialKG are at most 2 hops, we set the maximum path length to $T = 2$. We included "Equal" to ensure that the model stops automatically after the second hop. To ensure fairness, we randomly shuffled the exit paths of the knowledge graph. We set max patience to 20, meaning that training is terminated if there is no boost for 20 rewards on the validation set. Further information on the hyperparameters is available in the Appendix A.

4.4 Evaluation Metrics

In line with the baselines, we utilize recall@k as the evaluation metric for both path-level (path@k) and target entity-level (target@k) correctness.

4.5 Comparative Experiments

For the KG reasoning task, we assessed path recall at different K values (1, 3, 5, 10, 25) and target entity recall at position K (1, 3, 5, 10, 25). Our findings, which are presented in Table 4.5, demonstrate that our proposed LLM-ARK performs better than all benchmarked baselines in target@k metrics, surpassing the GPT-4. The performance gain

[3]https://github.com/thunlp/OpenKE

Model	Path@k					Target@k				
	path@1	path@3	path@5	path@10	path@25	target@1	target@3	target@5	target@10	target@25
Tri-LSTM	3.2	14.2	22.6	36.3	56.2	-	-	-	-	-
Seq2Seq	3.1	18.3	29.7	44.1	60.2	-	-	-	-	-
DialKG Walker	13.2	26.1	35.3	47.9	62.2	-	-	-	-	-
Seq2Path	14.92	24.95	31.1	38.68	48.15	15.65	27.04	33.86	42.52	53.28
AttnFlow	17.37	24.84	30.68	39.48	51.4	18.97	36.23	45.48	58.84	71.35
AttnIO	23.72	37.53	43.57	52.17	62.86	24.98	43.78	53.49	65.48	78.79
HiTKG	**25.99**	**38.67**	**49.18**	**59.32**	**71.27**	31.11	46.29	55.59	71.61	86.09
T5-DiffKG	-	-	-	-	-	26.80	54.33	61.75	-	-
GPT-4-Standard	0.007	-	-	-	-	14.91	-	-	-	-
GPT-4-Normal	0.02	-	-	-	-	13.30	-	-	-	-
GPT-4-OPA	0.09	-	-	-	-	12.19	-	-	-	-
LLaMA7B-ARK	22.37	32.79	38.92	49.45	60.18	**48.75**	**63.73**	**72.15**	**84.12**	**90.50**

Table 1: Path-level (path@k) and target-level (target@k) performance of KG path reasoning. LLM-ARK is benchmarked against several state-of-the-art baselines models on the OpenDialKG dataset.

Model	Path@k					Target@k				
	path@1	path@3	path@5	path@10	path@25	target@1	target@3	target@5	target@10	target@25
LLaMA7B-ARK(proposed)	22.37	**32.79**	**38.92**	**49.45**	**60.18**	**48.75**	**63.73**	72.15	84.12	90.50
LLaMA13B-ARK	**22.53**	31.40	37.29	46.67	55.45	48.44	61.43	69.52	81.00	88.71
LLaMA7B-ARK-UI	17.27	26.06	32.04	41.53	50.27	46.64	62.22	71.28	81.88	89.49
LLaMA7B-ARK-FP32	21.36	28.34	33.30	41.30	50.62	42.38	52.91	61.64	74.89	87.25
LLaMA7B-ARK-WT	16.13	21.42	27.40	36.79	47.41	32.88	46.24	55.69	71.12	84.32
LLaMA7B-ARK-WP	1.05	2.31	3.26	4.62	5.66	41.54	63.21	**75.07**	**87.22**	**96.47**

Table 2: Path-level (path@k) and target-level (target@k) performance of supervised KG path reasoning (metric: recall@k). LLM-ARK is benchmarked against several ablation models on the OpenDialKG dataset.

is signifcant, especially in recalls with taget@k: there is a 17.64% relative improvement in target@1, 17.44% in target@5 and 16.56% in target@10. Unfortunately, our model's path@k evaluation matrix socores do not outperform the state-of-the-art (SOTA) model because we trained using only the target arrival reward function, but we are very extensible and there is potential for improvement.

The GPT-4-Standard and GPT-4-Normal methods are deficient in path awareness. Nonetheless, our models exhibit improved outcomes with the addition of path awareness. Furthermore, GPT-4 rely solely on Zero Shot decoding for output path decoding. In contrast, our model implements PA-MLP to improve the environment-aware capabilities of LLMs. Additionally, due to the substantial comprehension of instructions and comprehensive information encoding features of the large model, we can modify cue words, eliminating the need for the complex multi-input aggregation technique utilized in HiTKG. Viewing large models as intelligent agents that explore a knowledge graph to acquire experience can reap benefits from the positive-negative feedback optimization mechanism of the Reinforcement Learning Policy Supervisor Algorithm. This methodology strengthens the training of our model to perform flexible reasoning on KGs in multi-step scenarios, outperforming not only GPT-4 but also smaller models. The efficacy of our approach is corroborated by our model's superior performance.

4.6 Analysis Experiment

As shown in Table 4.5, LLM-ARK was benchmarked against multiple ablation models on the OpenDialKG dataset. To evaluate the impact of instructions on model performance, we trained the LLaMA7B-ARK-UI model without instruction. The results of this model are the closest to those of LLaMA7B-ARK, indicating that the presence or absence of commands has an effect on the model's results, but not a serious one. To conserve computational resources, we use the bfloat16 computational type for all experiments except for LLaMA7B-ARK-FP32. This is due to the fact that LLMs require more GPU memory space, necessitating a reduction in the batchsize and a slower training speed. To our surprise, the Float32 computational type did not achieve higher experimental results. To determine the impact of model parameter size on training outcomes, experiments were conducted utilizing the LLaMA7B and LLaMA13B models. The model LLaMA7B-ARK-FP32 with a greater number of parameters only achieve higher scores for path@1. The KG reasoning task does not demand the model to execute intricate commands, thus a 7B parameter-sized model can effectively fit the KG reasoning dataset. LLaMA7B-ARK-WT that was trained by randomly initializing relation and entity embedding for the LLaMA7B-ARK-WT model is less effective than LLaMA7B-ARK. We conducted ablation experiments LLaMA7B-ARK-WP and found that the performance of our export environment-aware sub-module

### Task Background: Performing 2-hop reasoning on the knowledge graph. ### Instruction If you don’t think it’s necessary to perform the second hop in reasoning, stop the reasoning with the ’Equal’ relation. Given the Task Background and the Environment, directly output this path in triplet format without any other content.		
Success	FTE	### Environment: Dialog History: [] Utterance: Could you recommend popular books by Gail Carson Levine? Path History: [] Current Entity: Gail Carson Levine
	Ground Truth Path	["Gail Carson Levine","∼written_by","The Two Princesses of Bamarre"]
	LLM-ARK Reasoning Path	[["Gail Carson Levine","∼written_by","The Two Princesses of Bamarre"], ["The Two Princesses of Bamarre","Equal","The Two Princesses of Bamarre"]]
Failed	FTE	### Environment: Dialog History: ["user: Can you recommend a movie like the Shooter?", "assistant: A movie similar to Shooter is Nothing to Lose."] Utterance: "Ok who is in that one?" Path History: [["Shooter","has_genre","Thriller"],["Thriller","∼has_genre","Nothing to Lose"],["Nothing to Lose","starred_actors","Michael McKean"]] Current Entity: "Michael McKean"
	Ground Truth Path	["Michael McKean","∼starred_actors","Nothing to Lose"]
	LLM-ARK Reasoning Path	[["Michael McKean","∼starred_actors","Used Cars"],["Used Cars", "Equal", "Used Cars"]]

Table 3: Successes and failures of our model when performing inference tasks on the OpenDialKG dataset.

PA-MLP decreases substantially if we do not consider the exit paths. The score for path@1 is only 1.05%, which is 21.32% lower than LLaMA7B-ARK. These ablation experiments and results demonstrate the contribution and necessity of our subcomponents to the model.

4.7 Case Study

We resort to a case study, for a clear presentation of LLM-ARK’s path reasoning process as shown in Table A. Note that there are hundreds of neighbor nodes connected to each entity in the external KG. Intuitively, there could be diverse knowledge paths as response to the user’s question. As the success story shows, our model makes good use of FTE information and exit path information to make decisions, rather than making decisions based on relationships alone, because Gail Carson Levine’s work is not limited to The Two Princesses of Bamarre. As shown in the error case, our model still reasons about wrong paths, partly due to the dataset itself, because OpenDialKG is an open-domain conversational knowledge graph inference dataset, and similar contexts and the same starting entities in the training set choose different Groud Truth exit paths, and so it can interfere with the training of our model. It is worth mentioning that OpenDialKG is not a unique path inference; there are many potential paths to reach the target entity. To summarize, our model would have the potential for better performance on non-open-domain conversational knowledge graph inference datasets.

5 Conclusion

Our study explores the potential of LLMs in KG reasoning tasks through an examination of GPT-4 (OpenAI, 2023) and an understanding of the basic capabilities of LLMs. Our model, by conducting flexible and precise learning with rich environmental information, enabled flexible and accurate multilevel inference, attaining superior results compared to both GPT-4, and smaller models. Ultimately, our experimental results affirmed that misalignment between model knowledge and environmental data can significantly undermine model performance. Experiments also affirmed the critical role of considering all aspects in the model optimization process. We hope our research provides a valuable contribution to the field and inspires future work in this area.

Acknowledgments

We thank Lida Shi, Anqi Liu and members of the HAI Deep Learning group for valuable discussions and comments.

References

Aiwei Liu, Xuming Hu, Lijie Wen, and Philip S. Yu. A comprehensive evaluation of chatgpt’s zero-shot text-to-sql capability, 2023.

Paulo Shakarian, Abhinav Koyyalamudi, Noel Ngu, and Lakshmivihari Mareedu. An independent evaluation of chatgpt on mathematical word problems (mwp), 2023.

Viet Dac Lai, Nghia Trung Ngo, Amir Pouran Ben Veyseh, Hieu Man, Franck Dernoncourt, Trung Bui, and Thien Huu Nguyen. Chatgpt beyond english: Towards a comprehensive evaluation of large language models in multilingual learning, 2023.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. Chain-of-thought prompting elicits reasoning in large language models, 2023.

Mehran Kazemi, Najoung Kim, Deepti Bhatia, Xin Xu, and Deepak Ramachandran. Lambada: Backward chaining for automated reasoning in natural language, 2023.

Seungwhan Moon, Pararth Shah, Anuj Kumar, and Rajen Subba. OpenDialKG: Explainable conversational reasoning with attention-based walks over knowledge graphs. In Anna Korhonen, David Traum, and Lluís Màrquez, editors, *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 845–854, Florence, Italy, July 2019. Association for Computational Linguistics. doi:10.18653/v1/P19-1081. URL `https://aclanthology.org/P19-1081`.

Houyu Zhang, Zhenghao Liu, Chenyan Xiong, and Zhiyuan Liu. Grounded conversation generation as guided traverses in commonsense knowledge graphs. In Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel Tetreault, editors, *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2031–2043, Online, July 2020. Association for Computational Linguistics. doi:10.18653/v1/2020.acl-main.184. URL `https://aclanthology.org/2020.acl-main.184`.

Jinjie Ni, Vlad Pandelea, Tom Young, Haicang Zhou, and Erik Cambria. Hitkg: Towards goal-oriented conversations via multi-hierarchy learning. *Proceedings of the AAAI Conference on Artificial Intelligence*, 36(10):11112–11120, Jun. 2022. doi:10.1609/aaai.v36i10.21360. URL `https://ojs.aaai.org/index.php/AAAI/article/view/21360`.

Yi-Lin Tuan, Sajjad Beygi, Maryam Fazel-Zarandi, Qiaozi Gao, Alessandra Cervone, and William Yang Wang. Towards large-scale interpretable knowledge graph reasoning for dialogue systems. In Smaranda Muresan, Preslav Nakov, and Aline Villavicencio, editors, *Findings of the Association for Computational Linguistics: ACL 2022*, pages 383–395, Dublin, Ireland, May 2022. Association for Computational Linguistics. doi:10.18653/v1/2022.findings-acl.33. URL `https://aclanthology.org/2022.findings-acl.33`.

OpenAI. Gpt-4 technical report, 2023.

Thomas Carta, Clément Romac, Thomas Wolf, Sylvain Lamprier, Olivier Sigaud, and Pierre-Yves Oudeyer. Grounding large language models in interactive environments with online reinforcement learning, 2023.

Baolin Peng, Michel Galley, Pengcheng He, Hao Cheng, Yujia Xie, Yu Hu, Qiuyuan Huang, Lars Liden, Zhou Yu, Weizhu Chen, and Jianfeng Gao. Check your facts and try again: Improving large language models with external knowledge and automated feedback, 2023.

Yuqing Du, Olivia Watkins, Zihan Wang, Cédric Colas, Trevor Darrell, Pieter Abbeel, Abhishek Gupta, and Jacob Andreas. Guiding pretraining in reinforcement learning with large language models. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett, editors, *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 8657–8677. PMLR, 23–29 Jul 2023. URL `https://proceedings.mlr.press/v202/du23f.html`.

Weiran Yao, Shelby Heinecke, Juan Carlos Niebles, Zhiwei Liu, Yihao Feng, Le Xue, Rithesh Murthy, Zeyuan Chen, Jianguo Zhang, Devansh Arpit, Ran Xu, Phil Mui, Huan Wang, Caiming Xiong, and Silvio Savarese. Retroformer: Retrospective large language agents with policy gradient optimization, 2023.

Jaehun Jung, Bokyung Son, and Sungwon Lyu. AttnIO: Knowledge Graph Exploration with In-and-Out Attention Flow for Knowledge-Grounded Dialogue. In Bonnie Webber, Trevor Cohn, Yulan He, and Yang Liu, editors, *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3484–3497, Online, November 2020. Association for Computational Linguistics. doi:10.18653/v1/2020.emnlp-main.280. URL `https://aclanthology.org/2020.emnlp-main.280`.

Jun Xu, Haifeng Wang, Zheng-Yu Niu, Hua Wu, Wanxiang Che, and Ting Liu. Conversational graph grounded policy learning for open-domain conversation generation. In Dan Jurafsky, Joyce Chai, Natalie Schluter, and Joel Tetreault, editors, *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 1835–1845, Online, July 2020. Association for Computational Linguistics. doi:10.18653/v1/2020.acl-main.166. URL `https://aclanthology.org/2020.acl-main.166`.

Linhao Luo, Yuan-Fang Li, Gholamreza Haffari, and Shirui Pan. Reasoning on graphs: Faithful and interpretable large language model reasoning, 2023.

Wenlong Huang, Fei Xia, Ted Xiao, Harris Chan, Jacky Liang, Pete Florence, Andy Zeng, Jonathan Tompson, Igor Mordatch, Yevgen Chebotar, Pierre Sermanet, Noah Brown, Tomas Jackson, Linda Luu, Sergey Levine, Karol Hausman, and Brian Ichter. Inner monologue: Embodied reasoning through planning with language models, 2022.

Xizhou Zhu, Yuntao Chen, Hao Tian, Chenxin Tao, Weijie Su, Chenyu Yang, Gao Huang, Bin Li, Lewei Lu, Xiaogang Wang, Yu Qiao, Zhaoxiang Zhang, and Jifeng Dai. Ghost in the minecraft: Generally capable agents for open-world environments via large language models with text-based knowledge and memory, 2023.

Ishika Singh, Valts Blukis, Arsalan Mousavian, Ankit Goyal, Danfei Xu, Jonathan Tremblay, Dieter Fox, Jesse Thomason, and Animesh Garg. Progprompt: Generating situated robot task plans using large language models, 2022.

Noah Shinn, Federico Cassano, Edward Berman, Ashwin Gopinath, Karthik Narasimhan, and Shunyu Yao. Reflexion: Language agents with verbal reinforcement learning, 2023.

Miao Fan, Qiang Zhou, Emily Chang, and Thomas Fang Zheng. Transition-based knowledge graph embedding with relational mapping properties. In *Proceedings of the 28th Pacific Asia Conference on Language, Information and Computing*, pages 328–337, Phuket,Thailand, December 2014. Department of Linguistics, Chulalongkorn University. URL `https://aclanthology.org/Y14-1039`.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *CoRR*, abs/1707.06347, 2017. URL `http://arxiv.org/abs/1707.06347`.

Tom Young, Erik Cambria, Iti Chaturvedi, Hao Zhou, Subham Biswas, and Minlie Huang. Augmenting end-to-end dialogue systems with commonsense knowledge. *Proceedings of the AAAI Conference on Artificial Intelligence*, 32(1), Apr. 2018. doi:10.1609/aaai.v32i1.11923. URL `https://ojs.aaai.org/index.php/AAAI/article/view/11923`.

George Tucker, Surya Bhupatiraju, Shixiang Gu, Richard E. Turner, Zoubin Ghahramani, and Sergey Levine. The mirage of action-dependent baselines in reinforcement learning, 2018.

Chao Yu, Akash Velu, Eugene Vinitsky, Jiaxuan Gao, Yu Wang, Alexandre Bayen, and Yi Wu. The surprising effectiveness of PPO in cooperative multi-agent games. In *Thirty-sixth Conference on Neural Information Processing Systems Datasets and Benchmarks Track*, 2022.

A Appendix

A.1 Data Format

We preprocessed the OpenDialKG raw data to fit our KG inference task. There are individual errors in the raw data, and the information of the dataset after our screening is shown in Table 4.7.

A.2 Tricks

In the original paper of PPO, no implementation details and techniques are mentioned other than the use of GAE to compute the dominance function. Referring to this repository[4], we employ several optimization tricks.In the actual code implementation, to encourage the diversity of paths sampled by the strategy during training, we added an entropy regularization term to our loss function, scaled by the constant (β). We used the operation of normalization of advantage proposed in the paper [Tucker et al., 2018]. Learning rate decay can enhance the smoothness in the late training stage to some extent and improve the training effect. Here we use the linear decay of learning rate, with the number of training steps learning rate from the initial value of a linear decline to 0. Gradient clipping is a trick introduced to prevent the gradient from exploding during the training process, which also serves to stabilize the training process. Orthogonal Initialization is a neural network initialization method proposed to prevent problems such as gradient vanishing and gradient explosion at the beginning of training. Referring to the MAPPO [Yu et al., 2022], the Adam optimizer individually sets eps=1e-5, and this particular setting can improve the training performance of the algorithm to some extent.

A.3 GPT4 Prompts

For GPT4-OPA prompts, since max length is 2, we need to recursively get all exit paths at the next level of all exit paths of the current entity, most of which are omitted due to the large number of KG subgraph triples of exit paths.

A.4 HyperParameters

[4]https://github.com/Lizhi-sjtu/DRL-code-pytorch

Knowledge Graph			Dataset	
Entity	Relation	Triplets	Train Data	Test Data
100,719	1,383	1,172,554	31,428	5,547

Table 4: Detailed information about the number of knowledge graph entity-relationship triples and the number of dataset segmentation samples after processing the OpenDialKG dataset.

Standard Prompt
Task Background Performing 2-hop reasoning on the knowledge graph. ### Instruction If you don’t think it’s necessary to perform the second hop in reasoning, stop the reasoning with the ’Equal’ relation. Given the Task Background and the Environment, directly output this path in triplet format without any other content. ### Environment Utterance: What do you think about the Washinton Redskins? Are you a fan? Current Entity: Washington Redskins ### Examples . . . ### Response

Table 5: GPT4-Standard prompt only perceived user’s query and Current Entity.

Normal Prompt
Task Background Performing 2-hop reasoning on the knowledge graph. ### Instruction If you don’t think it’s necessary to perform the second hop in reasoning, stop the reasoning with the ’Equal’ relation. Given the Task Background and the Environment, directly output this path in triplet format without any other content. ### Environment Dialog History: [] Utterance: What do you think about the Washinton Redskins? Are you a fan? Path History: [] Current Entity: Washington Redskins ### Examples . . . ### Response

Table 6: GPT4-Normal prompt has more awareness of dialog context and path history.

OPA(Out Paths Aware) Prompt

```
### Task Background
Performing 2-hop reasoning on the knowledge graph.

### Instruction
Given the Task Background and the Environment, please choose select two consecutive paths KG path from a set of Out Paths.
If you don't think it's necessary to perform the second hop in reasoning, just select the 'Equal' relation at the second hop.
Directly output these path in triplet format without any other content.

### Environment
Dialog History: []
Utterance: What do you think about the Washinton Redskins? Are you a fan?
Path History: [] Current Entity: Washington Redskins
Out Path: ['Washington Redskins,Equal,
Washington Redskins',
'Washington Redskins,~Game,Mike Sellers',
'Washington Redskins,~Runner-up,Super Bowl VII',
.
.
.
'Ladell Betts,Ethnicity,African American']

### Examples
.
.
.

### Response
```

Table 7: GPT4-OPA prompt has more awareness of 2-hop exit KG path subgraphs.

Computing Infrastructure	Tesla A40 GPU
Search Strategy	Beam Search
Training Efficiency	6 seconds per step

Hyperparameter	Best Setting
use transe	True
out path aware	True
bf16	True
relation embedding size	200
entity embedding size	200
max out	50
train times	8
batchsize	4096
mini batch size	1024
train batch size	4
test batch size	48
positive reward	1
negative reward	-1
actor learning rate	5e-5
critic learning rate	5e-5
gamma	0.95
lamda	0.95
epsilon	0.2
K epochs	10
use advantage normalization	True
entropy coef	0.01
use learning rate decay	True
use gradient clip	True
use orthogonal init	True
set adam eps 1e-5	True
use tanh	True

Table 8: Additional implementation detail of LLM-ARK.