TRACE: An Experiential Framework for Coherent Multi-hop Knowledge Graph Question Answering

Yingxu Wang[1] **Jiaxin Huang**[1] **Mengzhu Wang**[2] **Nan Yin**[3] *
[1]Mohamed bin Zayed University of Artificial Intelligence [2]Hebei University of Technology
[3]City University of Hong Kong
yingxv.wang@gmail.com, Jiaxin.Huang@mbzuai.ac.ae, dreamkily@gmail.com
yinnan8911@gmail.com

Abstract

Multi-hop Knowledge Graph Question Answering (KGQA) requires coherent reasoning across relational paths, yet existing methods often treat each reasoning step independently and fail to effectively leverage experience from prior explorations, leading to fragmented reasoning and redundant exploration. To address these challenges, we propose **T**rajectory-aware **R**easoning with **A**daptive **C**ontext and **E**xploration priors (TRACE), an experiential framework that unifies LLM-driven contextual reasoning with exploration prior integration to enhance the coherence and robustness of multi-hop KGQA. Specifically, TRACE dynamically translates evolving reasoning paths into natural language narratives to maintain semantic continuity, while abstracting prior exploration trajectories into reusable experiential priors that capture recurring exploration patterns. A dual-feedback re-ranking mechanism further integrates contextual narratives with exploration priors to guide relation selection during reasoning. Extensive experiments on multiple KGQA benchmarks demonstrate that TRACE consistently outperforms state-of-the-art baselines.

1 Introduction

Knowledge Graph Question Answering (KGQA) aims to enable large language models (LLMs) to answer natural language questions by reasoning over structured knowledge graphs (KGs) (Choi et al., 2023; Saxena et al., 2020). This task is crucial for bridging human language and symbolic knowledge, supporting a wide range of intelligent applications such as search engines (Zhao et al., 2020; Kejriwal and Szekely, 2017), recommender systems (Wang et al., 2019; Guo et al., 2020), and personal assistants (Balog and Kenter, 2019; Liu et al., 2024). Despite its importance, answering complex questions remains challenging since it often requires multi-hop reasoning over relational paths in KGs, demanding both accurate symbolic traversal and robust semantic understanding.

In recent years, growing attention has been devoted to leveraging the reasoning capabilities of LLMs for KGQA (Yu et al., 2022; Liu et al., 2025). Existing studies vary in their assumptions about the roles LLMs should play within the reasoning pipeline, which can be broadly categorized into three paradigms. (1) LLMs as direct reasoners: the model is prompted with a natural language question and a serialized KG subgraph to infer answers in a zero-/few-shot manner(Sun et al., 2023; Bi et al., 2024). (2) LLMs as stepwise explorers: inspired by ReAct (Yao et al., 2023) and CoT (Wei et al., 2022), the model iteratively selects relations, traverses entities, and conditions future steps on prior results (Wang et al., 2026; Shen et al., 2025). (3) LLMs as re-rankers: traditional graph search produces candidate paths that the LLM scores for semantic and logical coherence (Liu et al., 2025; Yao et al., 2025). These paradigms demonstrate LLMs' versatility in bridging symbolic and semantic reasoning (Saxena et al., 2020; Zhang et al., 2022). However, current methods still fail to integrate contextual information from previous reasoning steps or leverage experience from prior explorations, often resulting in incoherent reasoning paths and redundant explorations (Shen et al., 2025).

This paper investigates the design of an experiential KGQA framework that enhances coherent multi-hop reasoning and learns from prior explorations. However, realizing such a framework presents three core challenges. (1) Contextual guidance: existing KGQA models often treat each reasoning step independently, neglecting how previously traversed relations influence subsequent decisions (Wang et al., 2026; Shen et al., 2025). This disrupts reasoning coherence and causes the reasoning path to deviate from the question's intent. The challenge lies in dynamically translating structured relation paths into coherent natural

* Corresponding author.

Findings of the Association for Computational Linguistics: ACL 2026, pages 1830–1851

language contexts that guide each reasoning step. (2) Exploration generalization: most approaches lack explicit mechanisms to incorporate prior exploration trajectories into subsequent reasoning steps, which limits their ability to avoid redundant explorations (Ma et al., 2025b; Yao et al., 2025). The key is to autonomously identify and abstract common patterns from historical explorations into generalizable priors. (3) Unified decision-making: experiential reasoning requires integrating the evolving reasoning context with accumulated exploration priors. Yet, existing methods lack mechanisms to synthesize these complementary sources of information (Liu et al., 2025; Ma et al., 2025a). The challenge is to build a unified framework that fuses contextual cues and experiential priors at each step, ensuring coherent and robust multi-hop reasoning.

To address these challenges, we propose **T**rajectory-aware **R**easoning with **A**daptive **C**ontext and **E**xploration priors (TRACE), an experiential framework that unifies LLM-driven contextual reasoning with exploration-aware knowledge integration. TRACE dynamically translates evolving reasoning trajectories into natural language narratives, ensuring that each decision remains contextually grounded in both the input question and the accumulated reasoning history. Moreover, it abstracts prior exploration trajectories into reusable experiential priors, enabling the model to reduce redundant exploration and guide subsequent reasoning. In addition, a dual-feedback re-ranking mechanism integrates contextual narratives with exploration priors to refine relation selection, resulting in more coherent and robust multi-hop reasoning. Our contributions are summarized as follows:

- We investigate experiential path reasoning in KGQA, where the key challenges are to preserve the coherence of multi-hop reasoning paths and to leverage experience from prior explorations, motivating the development of context-aware and exploration-guided reasoning strategies.
- We propose TRACE, a novel framework that generates context-rich narratives during reasoning and abstracts experience from prior explorations into reusable priors, enabling robust and coherent multi-hop reasoning over knowledge graphs.
- We conduct comprehensive experiments on multiple KGQA benchmarks, showing that the proposed TRACE consistently outperforms state-of-the-art baselines.

2 Related Work

LLMs for KGQA. The integration of LLMs into KGQA has advanced rapidly in recent years. One line of research employs a retrieve-then-read paradigm, wherein a relevant subgraph of the knowledge graph is first retrieved, serialized, and then provided to an LLM to generate the answer (Yao et al., 2025; Liu et al., 2025). While effective in certain scenarios, these methods inherently separate the retrieval and reasoning processes, thereby constraining the LLM's capacity to adaptively refine path exploration based on intermediate reasoning outcomes (Ma et al., 2025b; Fang et al., 2024). In contrast, another line of work enables LLMs to interact with the knowledge graph in a sequential, stepwise manner, allowing the model to iteratively select relations to traverse and explicitly construct multi-hop reasoning paths (Sun et al., 2023; Ma et al., 2025c, 2024). However, these methods still suffer from insufficient incorporation of reasoning path context, as relation selection at each step is typically performed without consideration of the evolving semantic narrative of the reasoning process (Wang et al., 2026; Shen et al., 2025; Lee et al., 2024). This often leads to logical inconsistencies and deviations from the original question intent. To mitigate this limitation, we introduce TRACE, a novel framework that incorporates natural language narratives at every step of the reasoning process, enabling the model to capture evolving semantics and produce more coherent and accurate multi-hop KGQA.

Learning from Experience in Reasoning Models. The ability to leverage experience from prior reasoning processes is crucial for developing robust reasoning systems. In pathfinding tasks, Reinforcement Learning (RL) has traditionally been employed, where sparse rewards are provided only upon successfully reaching a target (Xiong et al., 2017; Das et al., 2018). However, in complex multi-hop KGQA scenarios, successful reasoning paths are relatively rare, resulting in weak supervision signals (Wang et al., 2026; Zhang and Zhao, 2025). As a consequence, acquiring effective exploration strategies under such limited feedback remains challenging. Recent studies have explored leveraging intermediate reasoning trajectories and historical exploration processes to improve reasoning efficiency and stability, but they lack principled mechanisms for abstracting recurring exploration patterns into reusable priors to accumulate strate-

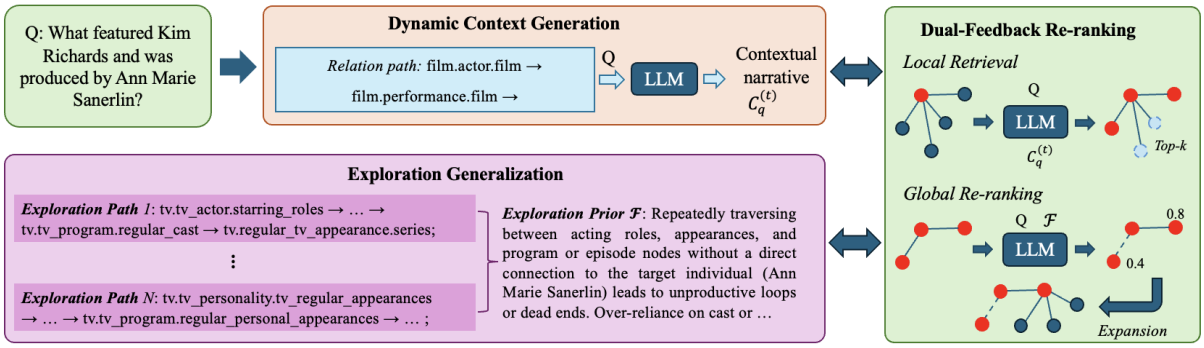


Figure 1: Overview of the proposed TRACE. Dynamic Context Generation translates evolving reasoning paths into natural language narratives, preserving semantic coherence across reasoning steps. Exploration Generalization abstracts prior exploration trajectories into reusable experiential priors that capture recurring exploration patterns. Finally, Dual-Feedback Re-ranking integrates contextual narratives with exploration priors to refine relation selection and enhance the robustness of multi-hop reasoning.

gic experience (Wu et al., 2024; Pan et al., 2023; Huang et al., 2023). To address this limitation, we propose a framework that systematically distills generalizable exploration priors from historical reasoning trajectories, enabling the model to integrate accumulated experience and enhance robustness of multi-hop KGQA.

3 Methodology

3.1 Problem Formulation

We formulate multi-hop Knowledge Graph Question Answering (KGQA) as a sequential decision-making problem over a knowledge graph $\mathcal{K} = (e_s, r, e_o) \subseteq \mathcal{E} \times \mathcal{R} \times \mathcal{E}$, where $\mathcal{E}$ and $\mathcal{R}$ denote the sets of entities and relations, respectively. Given a natural language question q with an associated topic entity set $\mathcal{E}_q \subset \mathcal{E}$, the objective is to identify the correct set of answer entities $\mathcal{A}_q \subset \mathcal{E}$. This is achieved by discovering an optimal reasoning path $\mathcal{P}_q \subseteq \{(e_1, r_1, e_2), (e_2, r_2, e_3), \ldots\}$ such that traversing the path from a topic entity in $\mathcal{E}_q$ leads to the correct answer entities in $\mathcal{A}_q$.

3.2 Overview of Framework

In this paper, we propose **Trajectory-aware Reasoning with Adaptive Context and Exploration Priors** (TRACE), an experiential framework designed to enhance the coherence and robustness of multi-hop KGQA. TRACE consists of three core components: (1) **Dynamic Context Generation.** At each reasoning step, the evolving reasoning path is dynamically translated into a natural language narrative, ensuring that relation selection is informed by the accumulated reasoning context rather than treated as an isolated decision. (2) **Exploration Generalization.** When a reasoning trajectory terminates, TRACE summarizes the explored path to capture the semantic and structural decisions made during exploration. These summaries are periodically consolidated into generalizable exploration priors, which are then leveraged to inform subsequent reasoning steps. (3) **Dual-Feedback Re-ranking.** At each step, an LLM-based retriever generates a top-k set of candidate relations conditioned on the contextual narrative, and their ranking is subsequently refined by incorporating the relation sequence of the current path together with the exploration priors.

3.3 Dynamic Context Generation

A central challenge in multi-hop KGQA is that relation selection at each step is often made in isolation, overlooking the evolving reasoning context. To overcome this limitation, the **Dynamic Context Generation** module translates preceding constructed reasoning paths into natural language narratives, which serve as contextual guidance for subsequent steps.

Formally, given a question q, we denote the preceding reasoning path up to step t as

$$\mathcal{P}_q^{(t)} = \{(e_1, r_1, e_2), (e_2, r_2, e_3), \ldots, (e_t, r_t, e_{t+1})\}, \quad (1)$$

where $(e_i, r_i, e_{i+1}) \in \mathcal{K}$. The corresponding relation sequence is

$$R_q^{(t)} = (r_1, r_2, \ldots, r_t). \quad (2)$$

At each step t, the relation sequence $R_q^{(t)}$ is transformed into a contextual narrative $C_q^{(t)}$ by an LLM-based generator $f_{\text{ctx}}(\cdot)$:

$$C_q^{(t)} = f_{\text{ctx}}(q, R_q^{(t)}), \quad (3)$$

where $C_q^{(t)}$ is a natural language description of $R_q^{(t)}$ (e.g., "find the director's birthplace") that conveys the semantics of the traversed relations in the context of the question.

By conditioning each decision on both the input question q and the contextual narrative $C_q^{(t)}$, this module maintains semantic coherence across reasoning steps and reduces the risk of logical drift during multi-hop reasoning in KGQA.

3.4 Exploration Generalization

A key limitation of existing KGQA approaches is that prior exploration trajectories are often not explicitly leveraged to inform subsequent reasoning, resulting in redundant exploration behaviors. To address this limitation, the **Exploration Generalization** module systematically summarizes prior exploration trajectories and abstracts them into reusable exploration priors.

For a given question q, a reasoning trajectory $\mathcal{P}_q^{(t)}$ is considered a terminated exploration $\mathcal{T}$ if its associated relation sequence $R_q^{(t)}$ reaches the maximum step limit L or cannot be further expanded:

$$\mathcal{T} = \mathcal{T}_{\text{depth}} \cup \mathcal{T}_{\text{expand}}, \tag{4}$$

where

$$\mathcal{T}_{\text{depth}} = \left\{ \mathcal{P}_q^{(t)} \;\middle|\; |R_q^{(t)}| \geq L \right\}, \tag{5}$$

and

$$\mathcal{T}_{\text{expand}} = \left\{ \mathcal{P}_q^{(t)} \mid \text{no expansion relations at step } t \right\}, \tag{6}$$

where $|R_q^{(t)}|$ denotes the number of traversed relations. For each terminated trajectory $\mathcal{P}_q^{(t)}$, an LLM-based summarization function $f_{\text{sum}}(\cdot)$ generates a trajectory summary D_q that characterizes its relation sequence $R_q^{(t)}$:

$$D_q = f_{\text{sum}}(q, R_q^{(t)}). \tag{7}$$

Although each trajectory summary D_q captures useful information about a specific exploration process, it is inherently path-specific and lacks generality. To address this limitation, TRACE further aggregates and abstracts these trajectory summaries into a set of reusable exploration priors $\mathcal{F}$:

$$\mathcal{F} = f_{\text{gen}}\left(\{ D_q \;\middle|\; \mathcal{P}_q^{(t)} \in \mathcal{T} \} \right), \tag{8}$$

where $f_{\text{gen}}(\cdot)$ is an abstraction function that identifies recurring exploration patterns. The resulting exploration priors $\mathcal{F}$ are stored as experiential knowledge and integrated into subsequent reasoning process in Sec. 3.5.

3.5 Dual-Feedback Re-ranking

While Dynamic Context Generation provides local contextual narratives $C_q^{(t)}$ and Exploration Generalization yields global exploration priors $\mathcal{F}$, an effective mechanism is required to integrate these complementary signals during relation selection. To this end, TRACE introduces a **Dual-Feedback Re-ranking** module that combines both local and global guidance in a two-stage process.

At each step t, an LLM-based retriever $f_{\text{ret}}(\cdot)$ first generates a top-k set of candidate relations from the neighborhood of the current entity e_t, conditioned on the contextual narrative $C_q^{(t)}$:

$$\mathcal{C}_t = f_{\text{ret}}(q, C_q^{(t)}, \mathcal{N}(e_t)), \tag{9}$$

where $\mathcal{N}(e_t)$ denotes the set of outgoing relations associated with $e_t \in \mathcal{E}$. Subsequently, each candidate relation $r \in \mathcal{C}_t$ is re-ranked by an LLM-based scoring function $f_{\text{rank}}(\cdot)$, which is conditioned on the relation sequence $R_q^{(t)}$ and the exploration priors $\mathcal{F}$:

$$s(r) = f_{\text{rank}}(q, R_q^{(t)}, r, \mathcal{F}), \tag{10}$$

where $s(r)$ denotes the relevance score assigned to relation r, reflecting the combined influence of local contextual information and exploration priors.

Rather than restricting expansion to the single top-ranked relation, TRACE preserves all candidates whose scores exceed a confidence threshold ζ, which is introduced as a hyperparameter. This design choice enables the framework to balance exploratory breadth with selection robustness. Each retained relation induces a new branch by extending the current path, resulting in a set of expanded reasoning trajectories:

$$R_q^{(t+1)} = \left(R_q^{(t)} \oplus r \;\middle|\; r \in \mathcal{C}_t,\ s(r) \geq \zeta \right). \tag{11}$$

where $\oplus$ denotes the operation of appending relation r to the current relation sequence $R_q^{(t)}$.

By employing $C_q^{(t)}$ for candidate retrieval and $(R_q^{(t)}, \mathcal{F})$ for re-ranking, this module effectively integrates contextual semantics with experiential knowledge, thereby ensuring coherent and robust reasoning.

3.6 Reasoning Process

The overall reasoning pipeline of TRACE is illustrated in Appendix A.1. Reasoning trajectories

Datasets	#Train	#Valid	#Test
WebQSP	2,848	250	1,639
CWQ	27,639	3,519	3,531

Table 1: Statistics of KGQA benchmarks.

are initialized from the topic entity set $\mathcal{E}_q$ and iteratively expanded through the joint use of Dynamic Context Generation and Dual-Feedback Reranking. At each step, the partial relation sequence $R_q^{(t)}$ is transformed into a contextual narrative $C_q^{(t)}$, which conditions both the retrieval and re-ranking of candidate relations. A trajectory $\mathcal{P}_q^{(t)}$ terminates when the maximum hop limit is reached or no feasible expansion is available. For each terminated trajectory, the Exploration Generalization module summarizes the explored path and abstracts it into exploration priors $\mathcal{F}$. Throughout the process, candidate relations with scores exceeding the confidence threshold ζ are retained to expand new reasoning trajectories. After finishing the above process, the answer set $\mathcal{A}$ is generated from the reasoning path with the highest relevance score.

4 Experiments

4.1 Experimental Settings

Datasets. To evaluate the effectiveness of TRACE, we conduct experiments on two widely used KGQA benchmarks: WebQSP (Talmor and Berant, 2018) and CWQ (Yih et al., 2016). The statistics of these benchmarks are reported in Table 1. More details about datasets are provided in Appendix A.2.

Baselines. We compare TRACE with a comprehensive set of baselines. These baselines include: the semantic parsing methods, e.g., KV-Mem (Miller et al., 2016), EmbedKGQA (Saxena et al., 2020), QGG (Lan and Jiang, 2020), NSM (He et al., 2021), TransferNet (Shi et al., 2021), KGT5 (Saxena et al., 2022), and DECAF (Yu et al., 2022); the retrieval-based methods, e.g., GraftNet (Sun et al., 2018), PullNet (Sun et al., 2019), SR+NSM (Zhang et al., 2022), and SR+NSM+E2E (Zhang et al., 2022); the general LLMs, including Flan-T5-xl (Chung et al., 2024), Alpaca-7B (Taori et al., 2023), Llama3-8B (Dubey et al., 2024), Qwen2.5-7B (Team, 2024), ChatGPT (Schulman et al., 2022), and ChatGPT+CoT (Wei et al., 2022); and recent LLMs with KG methods, including UniKGQA (Jiang et al., 2022), KD-CoT (Wang et al., 2023), Nutrea (Choi et al., 2023), ToG (Sun et al., 2023), RoG (Luo et al., 2023), KAPING (Baek et al., 2023), ReasoningLM (Jiang et al., 2023), FiDeLis (Sui et al., 2024), GNN-RAG (Mavromatis and Karypis, 2024), DoG (Ma et al., 2025b), DualR (Liu et al., 2025) , DP (Ma et al., 2025c), and RwT (Shen et al., 2025). The details are provided in Appendix A.3.

Type	Methods	WebQSP		CWQ	
		Hits@1	F1	Hits@1	F1
Semantic Parsing	KV-Mem	46.7	34.5	18.4	15.7
	EmbedKGQA	66.6	-	45.9	-
	QGG	73.0	73.8	36.9	37.4
	NSM	68.7	62.8	47.6	42.4
	TransferNet	71.4	-	48.6	-
	KGT5	56.1	-	36.5	-
	DECAF	82.1	78.8	70.4	-
Retrieval	GraftNet	66.4	60.4	36.8	32.7
	PullNet	68.1	-	45.9	-
	SR+NSM	68.9	64.1	50.2	47.1
	SR+NSM+E2E	69.5	64.1	49.3	46.3
LLMs	Flan-T5-xl	31.0	-	14.7	-
	Alpaca-7B	51.8	-	27.4	-
	Llama3-8B	30.3	25.7	30.5	27.8
	Qwen2.5-7B	28.4	23.7	25.9	24.1
	ChatGPT	66.8	-	39.9	-
	ChatGPT+CoT	75.6	-	48.9	-
LLMs with KGs	UniKGQA	77.2	72.2	51.2	49.0
	KD-CoT	68.6	52.5	55.7	-
	Nutrea	77.4	72.7	53.6	49.5
	ToG	81.9	76.0	68.5	60.2
	RoG	80.8	70.8	57.8	56.2
	KAPING	72.4	65.1	53.4	50.3
	ReasoningLM	78.5	71.0	69.0	64.0
	FiDeLis	84.3	78.3	71.5	64.3
	GNN-RAG	82.8	73.5	62.8	60.4
	DoG	65.4	55.6	41.0	46.4
	DualR	81.5	71.6	65.3	62.1
	DP	87.5	81.4	75.8	69.4
	RwT	87.0	79.7	72.4	66.7
	TRACE	**91.6**	**81.7**	**76.9**	**72.9**

Table 2: Performance comparison (%) on WebQSP and CWQ datasets. **Bold** results indicate the best performance.

Implementation Details. In TRACE, GPT-4.1 serves as the backbone for context generation, candidate retrieval, relation re-ranking, and exploration summarization (Liu et al., 2023). Following prior work (Sun et al., 2023; Ma et al., 2024), the reasoning process follows a beam search paradigm with a maximum number of iterations $I = 30$ and a maximum length of $L = 4$ hops. At each step, the retriever proposes $k = 3$ candidate relations for the WebQSP dataset and $k = 4$ for the CWQ dataset, which are then filtered and re-ranked by the Dual-Feedback Re-ranking module under a threshold of $\zeta = 0.5$. Following prior work (Luo et al., 2023;

Method	WebQSP		CWQ	
	Hits@1	F1	Hits@1	F1
TRACE (Llama2-13B)	88.1	78.3	74.3	70.7
TRACE (Qwen3-14B)	88.4	79.1	73.9	70.1
TRACE (GPT 4.1-mini)	90.3	80.6	75.7	71.3
TRACE (GPT 4.1)	**91.6**	**81.7**	**76.9**	**72.9**

Table 3: Performance of TRACE using different LLM-based planners as backbones on the WebQSP and CWQ datasets. **Bold** values denote the best results.

Method	WebQSP		CWQ	
	Hits@1	F1	Hits@1	F1
TRACE w/o CT	89.0	78.0	74.1	70.6
TRACE w/o ER	88.6	77.4	74.8	71.1
TRACE w/o ALL	87.5	76.7	73.6	69.9
TRACE	**91.6**	**81.7**	**76.9**	**72.9**

Table 4: The results of ablation studies on the WebQSP and CWQ datasets. **Bold** results indicate the best performance.

Wang et al., 2026; Ma et al., 2025c), performance is evaluated using Hits@1 and F1 score, capturing both exact-match accuracy and robustness in the presence of multiple valid answers.

4.2 Performance Comparison

Table 2 reports the performance of TRACE against state-of-the-art baselines on KGQA datasets. The results yield the following observations: (1) Semantic parsing methods map questions into logical forms, while retrieval-based methods extract candidate subgraphs for reasoning. Although these approaches provide interpretability and capture structural semantics, semantic parsing struggles with diverse query patterns, and retrieval methods separate retrieval from reasoning, limiting adaptability in multi-hop inference. In contrast, LLMs with KGs integrate question understanding with structured graph exploration, enabling more flexible path construction and stronger generalization. (2) General-purpose LLMs, such as Flan-T5-xl, Alpaca-7B, and Llama3-8B, rely on strong language understanding and broad contextual reasoning. They can handle diverse queries and generate coherent answers, but without grounding in structured graph semantics they often produce hallucinations or logically inconsistent results, particularly in multi-hop scenarios. In contrast, LLMs with KGs explicitly leverage graph structure to constrain reasoning and ensure factual consistency. As a result, general LLMs consistently underperform compared with LLM+KG methods. (3) LLMs with KGs methods have emerged as the leading paradigm in current KGQA research, as they couple the semantic fluency of LLMs with the structural rigor of knowledge graphs. By explicitly grounding reasoning in KGs, these methods reduce hallucination, ensure factual consistency, and achieve stronger generalization to compositional and multi-hop queries. Recent advances, such as DP and RwT, further illustrate the benefits of incorporating question-specific priors and structured exploration, confirming the effectiveness of this hybrid paradigm. Compared with prior methods, TRACE delivers improved performance, attributed to two main innovations: (i) the use of contextual narratives, which preserve semantic coherence across reasoning steps and prevent logical drift, and (ii) the generalization of prior exploration trajectories into reusable experiential priors, which guide subsequent reasoning and reduce redundant exploration. Together, these components enable TRACE to outperform existing LLM+KG approaches with more coherent and robust multi-hop reasoning.

4.3 Impact of Different LLMs

To evaluate the impact of different LLMs as backbone within the TRACE framework, we evaluate several backbones including Llama2-13B (Roque, 2025), Qwen3-14B (Team, 2024), GPT-4.1-mini, and GPT-4.1, as reported in Table 3. The results reveal that: (1) The effectiveness of TRACE depends on the reasoning capacity of the underlying LLMs. The consistent performance gap between medium-scale open-source models and larger proprietary models illustrates the sensitivity of KGQA to planner quality, indicating that more capable LLMs enable more accurate relation selection and more stable multi-hop reasoning. (2) TRACE remains effective even with relatively weaker planners. When instantiated with Llama2-13B, the framework still outperforms strong baselines such as RwT and FiDeLis on both datasets. This indicates that although high-capacity LLMs bring additional improvements, the design of TRACE ensures robustness and competitive performance across a wide range of backbones.

4.4 Ablation Study

We conduct ablation studies to evaluate the contributions of key components in TRACE: (1) TRACE w/o CT, removing the dynamic context generation module; (2) TRACE w/o ER, removing the explo-

ration generalization mechanism; and (3) TRACE w/o ALL, removing both modules simultaneously.

Experimental results are summarized in Table 4. From the results, we find that: (1) Removing the dynamic context generation module (TRACE w/o CT) leads to a clear performance drop, indicating that maintaining semantic coherence across reasoning steps is crucial for accurate path reasoning. Without this component, the model cannot ensure that intermediate decisions remain aligned with the overall query goal. (2) Removing the exploration generalization mechanism (TRACE w/o ER) leads to degraded performance, indicating that abstracting exploration priors into reusable knowledge is crucial for improving model performance. Without this component, the model is more likely to repeat similar exploration patterns, reducing the overall reasoning performance of TRACE. (3) When both modules are removed simultaneously (TRACE w/o ALL), the proposed TRACE suffers the largest drop in performance. This result shows that dynamic context generation and exploration generalization are complementary, and their combined effect is critical to realizing the effectiveness of TRACE.

4.5 Sensitivity Analysis

We perform a sensitivity analysis to examine the effect of three key hyperparameters in TRACE: the number of selected relations k, the maximum reasoning path length L, and the confidence threshold ζ. The parameter k specifies how many candidate relations are proposed by the LLM-based retriever at each step, L determines the maximum number of hops allowed during path construction, and ζ is the threshold for relation selection in path expansion.

Figure 2 presents the performance of TRACE on the WebQSP and CWQ datasets as we vary k, L within the range of $\{2, 3, 4, 5\}$ and ζ within the range of $\{0.4, 0.5, 0.6, 0.7\}$. From the results, we find that: (1) As shown in Figure 2(a), increasing k initially improves performance, but the gains plateau and eventually decline as the search space expands. Larger values of k promote broader relational exploration but also introduce irrelevant candidates and additional cost, while smaller values restrict the diversity of relations. To balance these trade-offs, we set $k = 3$ for WebQSP and $k = 4$ for CWQ as the default configurations. (2) As shown in Figure 2(b), extending the reasoning path length L yields diminishing returns beyond a certain depth. On both WebQSP and CWQ, performance improves steadily from $L = 2$ to $L = 4$, but declines at $L = 5$, suggesting that moderately deep paths are most effective while overly long horizons introduce noise. To balance accuracy and efficiency, we adopt $L = 4$ as the default path length in all experiments. (3) As shown in Figure 2(c), the confidence threshold ζ has a substantial impact on relation selection. Lower thresholds allow noisy candidates that reduce precision, whereas higher thresholds overly restrict expansion and harm recall. A moderate value provides the best trade-off, and we therefore set $\zeta = 0.5$ as the default value.

Method	WebQSP		CWQ	
	#Tokens	#Calls	#Tokens	#Calls
DoG	22,538	30.9	37,741	58.1
ToG	16,372	23.2	26,183	41.9
RwT	10,680	15.1	17,885	28.6
TRACE	**8,782**	**14.2**	**16,414**	**27.8**

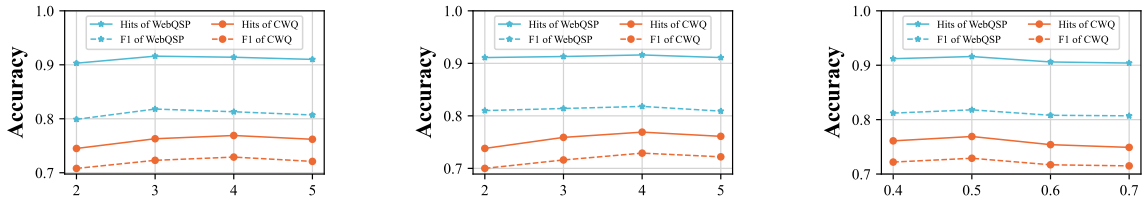
Table 5: Statistics of average number of LLM calls and token consumption per question on WebQSP and CWQ datasets.

4.6 Computation Consumption Analysis

Table 5 compares the computation consumption of different KGQA methods in terms of LLM calls and token usage, where TRACE achieves lower token consumption and fewer LLM calls. On the WebQSP dataset, TRACE lowers token usage from 10,680 to 8,782 and LLM calls from 15.1 to 14.2, outperforming RwT. On the CWQ dataset, it further reduces tokens from 17,885 to 16,414 and calls from 28.6 to 27.8, again demonstrating clear efficiency gains over RwT. These efficiency gains arise from TRACE, in which contextual narratives preserve semantic coherence across reasoning steps while exploration generalization reduces redundant exploration. As a result, TRACE lowers both LLM call frequency and token consumption, yielding a more efficient and scalable reasoning framework than existing baselines. More results on worst or failure cases can be found in Appendix. A.4.

4.7 Case study

Table 6 presents a case study comparing the outputs of TRACE with four representative LLMs: Llama-2-13B, Qwen-3-14B, GPT 4.1, and GPT 4.1-mini. For the query *"What featured Kim Richards and was produced by Ann Marie Sanerlin?"*, all baseline models produce incorrect answers, often hallucinating films or conflating unrelated television productions. In contrast, TRACE correctly identifies



(a) Number of selected relations k. (b) Reasoning path length L. (c) Threshold ζ.

Figure 2: Sensitivity analysis of hyperparameters on the WebQSP and CWQ datasets.

Question	What featured Kim Richards and was producted by Ann Marie Sanerlin?
Answer	Race to Witch Mountain
Llama-2-13B	The TV movie "The Christmas Card" (2006) featured Kim Richards and was produced by Anne Marie Sanerlin, however I couldn't verify if it matches exactly "Ann Marie Sanerlin".
Qwen-3-14B	The 2019 film "The Last Days of American Crime" featured Kim Richards and was produced by Ann Marie Sanerlin. However, this information is not widely documented, and Sanerlin's role in the production is not prominently noted in mainstream sources.
GPT 4.1	The project that featured Kim Richards and was produced by Ann Marie Sanerlin is the 1977 television film "The Car." Kim Richards starred in the film, and Ann Marie Sanerlin was credited as the producer.
GPT 4.1-mini	The project featuring Kim Richards and produced by Ann Marie Sanerlin is the reality TV series "The Real Housewives of Beverly Hills." Kim Richards is a cast member, and Ann Marie Sanerlin has served as a producer on the show.
TRACE	**`Reasoning Path`**: Entity (id: 1384221) → film.actor.film → film.performance.film → Race to Witch Mountain. **`Contextual Narrative`**: Identify the films featuring Kim Richards by tracing her acting roles and performances, and then determine which of these films were produced by Ann Marie Sanerlin. **`Exploration Paths`**: (i) tv.tv_actor.starring_roles → tv.regular_tv_appearance.series → tv.tv_program.regular_cast → tv.regular_tv_appearance.series; (ii) tv.tv_actor.guest_roles → tv.tv_guest_role.episodes_appeared_in; (iii) tv.tv_personality.tv_regular_appearances → tv.tv_regular_personal_appearance.program → tv.tv_program.regular_personal_appearances → tv.tv_regular_personal_appearance.program. **`Exploration Patterns`**: Repeatedly traversing between acting roles, appearances, and program or episode nodes without a direct connection to the target individual (Ann Marie Sanerlin) leads to unproductive loops or dead ends. Over-reliance on cast or appearance relationships, especially when the target is a producer rather than an actor, tends to exceed path limits without yielding relevant results. Future reasoning should prioritize direct production or crew-related links over cast or appearance-based paths when seeking information about behind-the-scenes personnel.

Table 6: Case study of TRACE. We highlight the correct answers in Red and the wrong answers in Blue.

Race to Witch Mountain by following a structured `Reasoning Path` that connects Kim Richards to her film performances and integrates production information to refine the candidate set. The reasoning process of TRACE begins by locating entities associated with Kim Richards and expanding along film-related relations that represent her acting roles. At each hop, the `Contextual Narrative` guides the model to remain semantically aligned with the query, first retrieving films featuring Kim Richards and then filtering them according to the production information linked to Ann Marie Sanerlin. This narrative grounding ensures that relation selection remains coherent and context-aware throughout the multi-hop reasoning process. In parallel, `Exploration Patterns` abstracted from prior exploration trajectories capture recurring unproductive behaviors, such as cast–series loops that fail to satisfy producer-related constraints. Through the interplay of narrative guidance and exploration prior, TRACE effectively narrows the search space and converges on the correct answer. Additional case study is provided in Appendix A.5.

5 Conclusion

In this paper, we propose **T**rajectory-aware **R**easoning with **A**daptive **C**ontext and **E**xploration priors (TRACE), an experiential framework that enhances the coherence and robustness of multi-hop knowledge graph question answering (KGQA). TRACE comprises three synergistic components: dynamic context generation, which preserves se-

mantic continuity across reasoning steps; exploration generalization, which abstracts prior exploration trajectories into reusable experiential priors; and dual-feedback re-ranking, which integrates contextual and experiential signals to guide relation selection. Extensive experiments on the WebQSP and CWQ datasets demonstrate that TRACE consistently outperforms state-of-the-art baselines.

Limitations

Although TRACE demonstrates strong empirical performance, it still has several limitations. The framework currently relies on manually designed prompts and exploration priors, which may limit its adaptability to heterogeneous or noisy knowledge graphs. Moreover, despite improved efficiency compared to prior approaches, the integration of multiple reasoning components introduces non-trivial inference costs. Future work will explore automatic prompt optimization, more lightweight retrieval mechanisms, and reinforcement learning–based controllers to further improve scalability and generalization. Extending TRACE beyond KGQA to domains such as scientific discovery and multi-agent collaboration also represents a promising direction.

References

Jinheon Baek, Alham Fikri Aji, and Amir Saffari. 2023. Knowledge-augmented language model prompting for zero-shot knowledge graph question answering. *arXiv preprint arXiv:2306.04136*.

Krisztian Balog and Tom Kenter. 2019. Personal knowledge graphs: A research agenda. In *Proceedings of the 2019 ACM SIGIR International Conference on Theory of Information Retrieval*, pages 217–220.

Zhenni Bi, Kai Han, Chuanjian Liu, Yehui Tang, and Yunhe Wang. 2024. Forest-of-thought: Scaling test-time compute for enhancing llm reasoning. *arXiv preprint arXiv:2412.09078*.

Hyeong Kyu Choi, Seunghun Lee, Jaewon Chu, and Hyunwoo J Kim. 2023. Nutrea: Neural tree search for context-guided multi-hop kgqa. *Proceedings of the Conference on Neural Information Processing Systems*, 36:35954–35965.

Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, and 1 others. 2024. Scaling instruction-finetuned language models. *Journal of Machine Learning Research*, 25(70):1–53.

Rajarshi Das, Shehzaad Dhuliawala, Manzil Zaheer, Luke Vilnis, Ishan Durugkar, Akshay Krishnamurthy, Alex Smola, and Andrew McCallum. 2018. Go for a walk and arrive at the answer: Reasoning over paths in knowledge bases using reinforcement learning. In *Proceedings of the International Conference on Learning Representations*.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, and 1 others. 2024. The llama 3 herd of models. *arXiv e-prints*, pages arXiv–2407.

Siyuan Fang, Kaijing Ma, Tianyu Zheng, Xinrun Du, Ningxuan Lu, Ge Zhang, and Qingkun Tang. 2024. Karpa: A training-free method of adapting knowledge graph as references for large language model's reasoning path aggregation. *arXiv preprint arXiv:2412.20995*.

Qingyu Guo, Fuzhen Zhuang, Chuan Qin, Hengshu Zhu, Xing Xie, Hui Xiong, and Qing He. 2020. A survey on knowledge graph-based recommender systems. *IEEE Transactions on Knowledge and Data Engineering*, 34:3549–3568.

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 International ACM Conference on Web Search & Data Mining*, pages 553–561.

Jie Huang, Xinyun Chen, Swaroop Mishra, Huaixiu Steven Zheng, Adams Wei Yu, Xinying Song, and Denny Zhou. 2023. Large language models cannot self-correct reasoning yet. *arXiv preprint arXiv:2310.01798*.

Jinhao Jiang, Kun Zhou, Wayne Xin Zhao, Yaliang Li, and Ji-Rong Wen. 2023. Reasoninglm: Enabling structural subgraph reasoning in pre-trained language models for question answering over knowledge graph. *arXiv preprint arXiv:2401.00158*.

Jinhao Jiang, Kun Zhou, Wayne Xin Zhao, and Ji-Rong Wen. 2022. Unikgqa: Unified retrieval and reasoning for solving multi-hop question answering over knowledge graph. *arXiv preprint arXiv:2212.00959*.

Mayank Kejriwal and Pedro A. Szekely. 2017. Knowledge graphs for social good: An entity-centric search engine for the human trafficking domain. *IEEE Transactions on Big Data*, 8:592–606.

Yunshi Lan and Jing Jiang. 2020. Query graph generation for answering multi-hop complex questions from knowledge bases. Proceedings of the Annual Meeting of the Association for Computational Linguistics.

Seongmin Lee, Jaewook Shin, Youngjin Ahn, Seokin Seo, Ohjoon Kwon, and Kee-Eung Kim. 2024. Zero-shot multi-hop question answering via monte-carlo tree search with large language models. *arXiv preprint arXiv:2409.19382*.

Guangyi Liu, Yongqi Zhang, Yong Li, and Quanming Yao. 2025. Dual reasoning: A gnn-llm collaborative framework for knowledge graph question answering. In *The Second Conference on Parsimony and Learning (Proceedings Track)*.

Hanmeng Liu, Ruoxi Ning, Zhiyang Teng, Jian Liu, Qiji Zhou, and Yue Zhang. 2023. Evaluating the logical reasoning ability of chatgpt and gpt-4. *arXiv preprint arXiv:2304.03439*.

Lingyuan Liu, Huifang Du, Xiaolian Zhang, Mengying Guo, Haofen Wang, and Meng Wang. 2024. A question-answering assistant over personal knowledge graph. In *Proceedings of the 47th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 2708–2712.

Linhao Luo, Yuan-Fang Li, Gholamreza Haffari, and Shirui Pan. 2023. Reasoning on graphs: Faithful and interpretable large language model reasoning. *arXiv preprint arXiv:2310.01061*.

Chuangtao Ma, Yongrui Chen, Tianxing Wu, Arijit Khan, and Haofen Wang. 2025a. Large language models meet knowledge graphs for question answering: Synthesis and opportunities. *arXiv preprint arXiv:2505.20099*.

Jie Ma, Zhitao Gao, Qi Chai, Wangchun Sun, Pinghui Wang, Hongbin Pei, Jing Tao, Lingyun Song, Jun Liu, Chen Zhang, and 1 others. 2025b. Debate on graph: a flexible and reliable reasoning framework for large language models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 23, pages 24768–24776.

Jie Ma, Ning Qu, Zhitao Gao, Rui Xing, Jun Liu, Hongbin Pei, Jiang Xie, Linyun Song, Pinghui Wang, Jing Tao, and 1 others. 2025c. Deliberation on priors: Trustworthy reasoning of large language models on knowledge graphs. *arXiv preprint arXiv:2505.15210*.

Shengjie Ma, Chengjin Xu, Xuhui Jiang, Muzhi Li, Huaren Qu, Cehao Yang, Jiaxin Mao, and Jian Guo. 2024. Think-on-graph 2.0: Deep and faithful large language model reasoning with knowledge-guided retrieval augmented generation. *arXiv preprint arXiv:2407.10805*.

Costas Mavromatis and George Karypis. 2024. Gnn-rag: Graph neural retrieval for large language model reasoning. *arXiv preprint arXiv:2405.20139*.

Alexander Miller, Adam Fisch, Jesse Dodge, Amir-Hossein Karimi, Antoine Bordes, and Jason Weston. 2016. Key-value memory networks for directly reading documents. *arXiv preprint arXiv:1606.03126*.

Liangming Pan, Michael Saxon, Wenda Xu, Deepak Nathani, Xinyi Wang, and William Yang Wang. 2023. Automatically correcting large language models: Surveying the landscape of diverse self-correction strategies. *arXiv preprint arXiv:2308.03188*.

Luís Roque. 2025. The evolution of llama: From llama 1 to llama 3.1.

Apoorv Saxena, Adrian Kochsiek, and Rainer Gemulla. 2022. Sequence-to-sequence knowledge graph completion and question answering. *arXiv preprint arXiv:2203.10321*.

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.

John Schulman, Barret Zoph, Christina Kim, Jacob Hilton, Jacob Menick, Jiayi Weng, Juan Felipe Ceron Uribe, Liam Fedus, Luke Metz, Michael Pokorny, and 1 others. 2022. Chatgpt: Optimizing language models for dialogue. *OpenAI blog*, 2(4).

Tiesunlong Shen, Jin Wang, Xuejie Zhang, and Erik Cambria. 2025. Reasoning with trees: Faithful question answering over knowledge graph. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics*, pages 3138–3157.

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. *arXiv preprint arXiv:2104.07302*.

Yuan Sui, Yufei He, Nian Liu, Xiaoxin He, Kun Wang, and Bryan Hooi. 2024. Fidelis: Faithful reasoning in large language model for knowledge graph question answering. *arXiv preprint arXiv:2405.13873*.

Haitian Sun, Tania Bedrax-Weiss, and William W Cohen. 2019. Pullnet: Open domain question answering with iterative retrieval on knowledge bases and text. *arXiv preprint arXiv:1904.09537*.

Haitian Sun, Bhuwan Dhingra, Manzil Zaheer, Kathryn Mazaitis, Ruslan Salakhutdinov, and William W Cohen. 2018. Open domain question answering using early fusion of knowledge bases and text. *arXiv preprint arXiv:1809.00782*.

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. *arXiv preprint arXiv:2307.07697*.

Alon Talmor and Jonathan Berant. 2018. The web as a knowledge-base for answering complex questions. *arXiv preprint arXiv:1803.06643*.

Rohan Taori, Ishaan Gulrajani, Tianyi Zhang, Yann Dubois, Xuechen Li, Carlos Guestrin, Percy Liang, and Tatsunori B Hashimoto. 2023. Stanford alpaca: An instruction-following llama model.

Qwen Team. 2024. Qwen2 technical report. *arXiv preprint arXiv:2407.10671*.

Hongwei Wang, Miao Zhao, Xing Xie, Wenjie Li, and Minyi Guo. 2019. Knowledge graph convolutional networks for recommender systems. In *The world wide web conference*, pages 3307–3313.

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. *arXiv preprint arXiv:2308.13259*.

Yingxu Wang, Shiqi Fan, Mengzhu Wang, Siyang Gao, Chao Wang, and Nan Yin. 2026. DAMR: Efficient and adaptive context-aware knowledge graph question answering with LLM-guided MCTS. In *Proceedings of the International Conference on Learning Representations*.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, and 1 others. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Proceedings of the Conference on Neural Information Processing Systems*, 35:24824–24837.

Tsung-Han Wu, Long Lian, Joseph E Gonzalez, Boyi Li, and Trevor Darrell. 2024. Self-correcting llm-controlled diffusion models. In *The IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6327–6336.

Wenhan Xiong, Thien Hoang, and William Yang Wang. 2017. Deeppath: A reinforcement learning method for knowledge graph reasoning. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 564–573.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. 2023. React: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations (ICLR)*.

Tianjun Yao, Haoxuan Li, Zhiqiang Shen, Pan Li, Tongliang Liu, and Kun Zhang. 2025. Learning efficient and generalizable graph retriever for knowledge-graph question answering. *arXiv preprint arXiv:2506.09645*.

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.

Donghan Yu, Sheng Zhang, Patrick Ng, Henghui Zhu, Alexander Hanbo Li, Jun Wang, Yiqun Hu, William Wang, Zhiguo Wang, and Bing Xiang. 2022. Decaf: Joint decoding of answers and logical forms for question answering over knowledge bases. *arXiv preprint arXiv:2210.00063*.

Jing Zhang, Xiaokang Zhang, Jifan Yu, Jian Tang, Jie Tang, Cuiping Li, and Hong Chen. 2022. Subgraph retrieval enhanced model for multi-hop knowledge base question answering. *arXiv preprint arXiv:2202.13296*.

Zhiqiang Zhang and Wen Zhao. 2025. A collaborative reasoning framework powered by reinforcement learning and large language models for complex questions answering over knowledge graph. In *Proceedings of the 31st International Conference on Computational Linguistics*, pages 10672–10684.

Xuejiao Zhao, Huanhuan Chen, Zhenchang Xing, and Chunyan Miao. 2020. Brain-inspired search engine assistant based on knowledge graph. *IEEE Transactions on Neural Networks and Learning Systems*, 34:4386–4400.

A Appendix

A.1 Algorithm

Algorithm 1: Trajectory-aware Reasoning with Adaptive Context and Exploration Priors (TRACE)

Input: Question q, knowledge graph $\mathcal{K}$, maximum reasoning steps L, candidate pool size k, confidence threshold ζ, maximum iterations I

Output: Predicted answer entity set $\mathcal{A}$

Initialize exploration priors $\mathcal{F} \leftarrow \emptyset$
Initialize current reasoning path $\mathcal{P}_q^{(0)}$ from topic entities $\mathcal{E}_q$
for $i = 1$ to I **do**
  for $t = 1$ to L **do**
    Dynamic Context Generation: Generate contextual narrative $C_q^{(t)} = f_{\text{ctx}}(q, R_q^{(t-1)})$
    Candidate Retrieval: Retrieve top-k candidate relations from the local neighborhood $\mathcal{C}_t = f_{\text{ret}}(q, C_q^{(t)}, \mathcal{N}(e_t))$
    Dual-Feedback Re-ranking: For each $r \in \mathcal{C}_t$, compute relevance score $s(r) = f_{\text{rank}}(q, R_q^{(t)}, r, \mathcal{F})$, and expand the reasoning path with relations satisfying $s(r) \geq \zeta$
  end for
  if $\mathcal{P}_q^{(t)} \in \mathcal{T}_{\text{depth}} \cup \mathcal{T}_{\text{expand}}$ **then**
    Trajectory Summarization: Generate exploration summary $D_q = f_{\text{sum}}(q, R_q^{(t)})$
    Exploration Generalization: Update exploration priors $\mathcal{F} \leftarrow f_{\text{gen}}(\mathcal{F} \cup \{D_q\})$
  end if
end for
return $\mathcal{A}$ from the reasoning trajectory with the highest relevance score

A.2 Datasets

A.2.1 Dataset Description

To evaluate the effectiveness of the proposed TRACE, we conduct extensive experiments on two widely adopted multi-hop Knowledge Graph Question Answering (KGQA) benchmarks: WebQSP (Talmor and Berant, 2018) and CWQ (Yih et al., 2016). The detailed description of each dataset is provided below:

- WebQSP is a widely adopted benchmark for multi-hop KGQA, derived from the WebQuestions dataset (Yih et al., 2016). Each query is paired with an annotated SPARQL program over Freebase, enabling precise evaluation of reasoning accuracy. The benchmark emphasizes complex reasoning phenomena such as multi-hop inference, relation composition, and entity disambiguation, making it a rigorous testbed for KGQA models. It comprises 2,848 training, 250 validation, and 1,639 test instances.

- ComplexWebQuestions (CWQ) is a large-scale benchmark for evaluating compositional reasoning in KGQA. It is derived by decomposing seed queries from WebQuestionsSP into multiple sub-questions and recombining them into more complex forms (Talmor and Berant, 2018). Each question is annotated with SPARQL programs over Freebase, enabling systematic assessment of multi-hop reasoning. Compared with WebQSP, CWQ introduces longer reasoning chains and richer compositional structures, making it a more challenging benchmark. The dataset contains 27,639 training, 3,519 validation, and 3,531 test instances.

A.2.2 Data processing

Following prior work (Shen et al., 2025; Wang et al., 2026), we preprocess these two benchmarks by constructing localized subgraphs centered on the topic entity of each question to reduce the search space. For every query in WebQSP (Yih et al., 2016) and CWQ (Talmor and Berant, 2018), we extract a subgraph from Freebase that includes all triples within a predefined number of hops from the question entity. This

strategy retains the essential contextual information needed for multi-hop reasoning while substantially improving computational efficiency. Following prior work (Sun et al., 2023; Wang et al., 2026), we randomly sample 1,000 questions from the test set of each dataset for evaluation.

A.3 Baselines

In this part, we introduce the details of the compared baselines as follows:

- **Semantic Parsing Methods.** We compare our TRACE with seven semantic parsing methods:
  - **KV-Mem**: KV-Mem (Miller et al., 2016) facilitate direct document reading by encoding information into key–value memory slots, enabling efficient retrieval and integration of relevant content for question answering without relying solely on sequential context.
  - **EmbedKGQA**: EmbedKGQA (Saxena et al., 2020) enhances multi-hop question answering over knowledge graphs by leveraging pretrained knowledge base embeddings, enabling effective reasoning over entities and relations without the need for explicit path enumeration.
  - **QGG**: QGG (Lan and Jiang, 2020) tackles multi-hop complex question answering over knowledge bases by constructing query graphs that capture the compositional structure of questions, enabling systematic reasoning across entities and relations.
  - **NSM**: NSM (He et al., 2021) advances multi-hop knowledge base question answering through a teacher–student framework, where the teacher network employs bidirectional reasoning to produce intermediate supervision signals, enabling the student network to learn more reliable reasoning paths and reduce spurious inference.
  - **TransferNet**: TransferNet (Shi et al., 2021) provides an effective and transparent framework for multi-hop question answering over relation graphs by transferring relational evidence across hops, thereby enhancing interpretability and reasoning accuracy.
  - **KGT5**: KGT5 (Saxena et al., 2022) addresses knowledge graph completion and question answering within a unified sequence-to-sequence framework, enabling the model to jointly generate missing triples and answer queries through end-to-end training.
  - **DECAF**: DECAF (Yu et al., 2022) jointly decodes both answers and logical forms for question answering over knowledge bases, enabling the model to generate interpretable reasoning paths alongside answers within a unified sequence generation framework.
- **Retrieval-Based Methods.** We compare our TRACE with four retrieval-based methods:
  - **GraftNet**: GraftNet (Sun et al., 2018) addresses open-domain question answering by integrating knowledge bases and textual evidence in an early fusion framework, allowing the model to jointly reason over structured and unstructured information sources.
  - **PullNet**: PullNet (Sun et al., 2019) enables open-domain question answering by iteratively retrieving relevant facts from both knowledge bases and text, dynamically constructing a subgraph to support multi-hop reasoning across heterogeneous information sources.
  - **SR+NSM**: SR+NSM (Zhang et al., 2022) enhances multi-hop knowledge base question answering by integrating a subgraph retrieval mechanism with a neural state machine, enabling the model to first retrieve a relevant subgraph and then perform stepwise reasoning within this focused context for improved answer accuracy.
  - **SR+NSM+E2E**: SR+NSM+E2E (Zhang et al., 2022) further advances multi-hop knowledge base question answering by jointly training subgraph retrieval and neural state machine reasoning in an end-to-end manner, enabling more effective integration between subgraph selection and multi-hop inference for improved answer prediction.
- **General Large Language Models (LLMs).** We compare our TRACE with six general LLMs:
  - **Flan-T5-xl**: Flan-T5-xl (Chung et al., 2024) is a large-scale instruction-finetuned language model that builds on the T5 architecture, achieving superior generalization across diverse tasks by leveraging fine-tuning on a broad set of instructional prompts.

- **Alpaca-7B**: Alpaca-7B (Taori et al., 2023) is an instruction-following language model based on LLaMA, developed by Stanford, and fine-tuned on a curated set of instructional demonstrations to enhance generalization and task-following abilities.
- **Llama3-8B**: Llama3-8B (Dubey et al., 2024) is a next-generation instruction-following language model from Meta, featuring 8 billion parameters and trained with advanced alignment techniques to deliver robust performance across a wide range of tasks.
- **Qwen2.5-7B**: Qwen2.5-7B (Team, 2024) is a 7-billion-parameter instruction-tuned language model from Alibaba, designed for high-quality, general-purpose reasoning and enhanced task following through extensive multi-domain instruction fine-tuning.
- **ChatGPT**: ChatGPT (Schulman et al., 2022) is a conversational large language model developed by OpenAI, optimized for dialogue and task-oriented interactions through extensive instruction tuning and reinforcement learning from human feedback.
- **ChatGPT+CoT**: ChatGPT with Chain-of-Thought (CoT) (Wei et al., 2022) augments the standard ChatGPT model with step-by-step reasoning capabilities, enabling more interpretable and accurate responses on complex tasks through explicit multi-step thought processes.

• **LLMs with KG.** We compare our TRACE with thirteen LLMs with KG methods:

- **UniKGQA**: UniKGQA (Jiang et al., 2022) unifies retrieval and reasoning for multi-hop question answering over knowledge graphs, jointly optimizing entity retrieval and reasoning steps within an end-to-end framework to enhance answer accuracy and reasoning transparency.
- **KD-CoT**: KD-CoT (Wang et al., 2023) explores faithful reasoning in large language models for knowledge-intensive question answering by integrating knowledge-grounded chain-of-thought prompting, thereby improving factual accuracy and interpretability in multi-hop reasoning tasks.
- **Nutrea**: Nutrea (Choi et al., 2023) introduces a neural tree search framework for context-guided multi-hop knowledge graph question answering, enabling efficient exploration and aggregation of reasoning paths guided by both question context and graph structure.
- **ToG**: ToG (Sun et al., 2023) is a framework that enables large language models to perform deep and responsible reasoning over knowledge graphs by combining structured graph information with iterative thinking and verification mechanisms for reliable multi-hop QA.
- **RoG**: RoG (Luo et al., 2023) enables deep and responsible reasoning of large language models over knowledge graphs by explicitly modeling structured multi-hop inference, enhancing both reasoning transparency and factual reliability in complex question answering scenarios.
- **KAPING**: KAPING (Baek et al., 2023) promotes faithful and interpretable large language model reasoning by leveraging explicit graph-based structures, ensuring transparent multi-hop inference and improved answer reliability for knowledge-intensive tasks.
- **ReasoningLM**: ReasoningLM (Jiang et al., 2023) enables structural subgraph reasoning in pre-trained language models for question answering over knowledge graphs, allowing the model to explicitly exploit subgraph structures for more accurate and interpretable multi-hop reasoning.
- **FiDeLis**: FiDeLis (Sui et al., 2024) enables faithful reasoning in large language models for knowledge graph question answering by integrating explicit logical constraints and stepwise supervision, thereby improving factual consistency and interpretability in multi-hop inference.
- **GNN-RAG**: GNN-RAG (Mavromatis and Karypis, 2024) enhances large language model reasoning by employing graph neural retrieval, enabling the model to incorporate structured knowledge from graphs for more accurate and context-aware question answering.
- **DoG**: DoG (Ma et al., 2025b) presents a flexible and reliable reasoning framework for large language models, leveraging debate-style interactions over knowledge graphs to improve reasoning transparency, flexibility, and answer robustness.
- **DualR**: DualR (Liu et al., 2025) introduces a collaborative framework for knowledge graph question answering, where graph neural networks and large language models jointly perform complementary reasoning to enhance multi-hop inference accuracy and interpretability.

Setting	WebQSP		CWQ	
	#Tokens	#Calls	#Tokens	#Calls
Worst	13,787	22.3	29,381	46.3
Failure	10,623	16.3	20,849	34.0
Average	**8,782**	**14.2**	**16,414**	**27.8**

Table 7: Statistics of average number of LLM calls and token consumption per question under worst and failure cases on the WebQSP and CWQ datasets datasets.

- **DP**: DP (Ma et al., 2025c) facilitates trustworthy reasoning of large language models on knowledge graphs by incorporating prior knowledge and iterative deliberation, thereby enhancing reliability, transparency, and robustness in complex question answering tasks.
- **RwT**: RwT (Shen et al., 2025) achieves faithful question answering over knowledge graphs by organizing multi-hop inference as a tree-based reasoning process, ensuring interpretable and reliable answer generation.

A.4 More Computation Consumption Analysis

To provide a more comprehensive scalability analysis, we re-ran the experiments on WebQSP and CWQ and recorded per-question token consumption and LLM call statistics. The corresponding worst-case and failure-case results are reported in Table 7. Notably, in failure mode, both token usage and LLM calls are only moderately higher than the average. As described in Section 3.4, reasoning expansion terminates when the LLM determines that no valid relations can be proposed. Although failure cases incur slightly higher costs than the average due to difficulty, the mechanism ensures they terminate essentially when no valid expansions are found, preventing the exponential cost blow-up typically seen in unconstrained search. In contrast, worst-case instances involve deeper exploration across multiple candidate paths, resulting in higher computational cost. Even in this scenario, token usage increases to approximately 1.5× the average on the WebQSP dataset and 1.7× on the CWQ dataset, demonstrating bounded growth rather than uncontrolled expansion. Furthermore, the exploration prior mechanism suppresses redundant or previously identified unproductive exploration patterns, preventing repeated branching and mitigating the risk of exponential cost escalation.

A.5 More case studies

Table 8 presents a case study comparing the outputs of TRACE with four representative LLMs: Llama-2-13B, Qwen-3-14B, GPT 4.1, and GPT 4.1-mini. For the query *What guitar does Corey Taylor play?*, all baseline models produce incorrect answers, often attributing instruments such as Gibson, Jackson, or Fender guitars. In contrast, TRACE correctly identifies *Bass guitar* by traversing a structured `Reasoning Path` that links Corey Taylor to his instrument associations in the knowledge graph. This process is guided by a `Contextual Narrative` that remains semantically aligned with the question, allowing the model to prioritize instrument-related relations over indirect contribution-based paths. In parallel, `Exploration Patterns` capture recurring unproductive traversal behaviors, guiding the search away from contribution- or recording-centric relations that fail to surface instrument information. As illustrated in this case, TRACE efficiently arrives at the correct answer while preserving interpretability through explicit KG-grounded reasoning.

A.6 Prompts

To elucidate the role of prompting in TRACE, we highlight the prompt designs within its three core components: (i) In Dynamic Context Generation, prompts are designed to transform the evolving relation sequence into coherent contextual narratives, thereby ensuring that each reasoning step remains consistent with both the original query and the accumulated path, as shown in Figure. 3. (ii) In Exploration Generalization, prompts guide the LLM to summarize terminated exploration trajectories and distill recurring exploration patterns into reusable experiential priors, as illustrated in Figure 4 and Figure 5. (iii) In Dual-Feedback Re-ranking, prompts are employed in a two-stage process: the first stage retrieves a

top-k set of candidate relations conditioned on the contextual narrative, and the second stage re-ranks these candidates by jointly considering the evolving relation sequence and the exploration priors, as shown in Figure. 6 and 7.

Question	What guitar does Corey Taylor play?
Answer	Bass guitar
Llama-2-13B	Corey Taylor, the lead vocalist of Slipknot and Stone Sour, is often seen playing various guitars, but his primary guitar is a custom Ibanez RG, and he also uses other Ibanez models and Dean Guitars.
Qwen-3-14B	Corey Taylor, known for his work with Slipknot and Stone Sour, primarily plays Jackson JS Series guitars, including the JS69 and JS32 models. He also uses his signature Jackson Corey Taylor JS guitars, designed for heavy rock and metal styles.
GPT 4.1	Corey Taylor, primarily known as a vocalist for Slipknot and Stone Sour, occasionally plays guitar. He has been seen using Gibson Explorer, Gibson Les Paul, and Fender Telecaster guitars during live performances and studio sessions, especially with Stone Sour.
GPT 4.1-mini	Corey Taylor, lead vocalist of Slipknot and Stone Sour, primarily plays Gibson guitars, especially the Gibson Les Paul models. He has also been seen using Fender Stratocasters and other custom guitars during performances. His Gibson Les Paul Custom is one of his signature instruments.
TRACE	**`Reasoning Path`**: Entity (id: 1242859) → music.group_member.instruments_played → Bass guitar. **`Contextual Narrative`**: Identify Corey Taylor's musical contributions, determine the recordings he contributed to, and then find the tracks associated with those contributions to discover what guitar he plays. **`Exploration Paths`**: (i) music.artist.contribution → music.recording_contribution.contributor → music.artist.track_contributions → music.track_contribution.role; (ii) music.artist.contribution → music.recording_contribution.contributor → music.artist.contribution → music.recording_contribution.album; (iii) music.artist.contribution → music.recording_contribution.contributor → music.artist.track_contributions → music.track_contribution.track; **`Exploration Patterns`**: Tracing an artist's instrument details by following their musical contributions, recording credits, or associated tracks/albums often leads to overly long and unproductive paths that do not yield specific information about the instruments they play. These approaches tend to exhaust path depth limits without success because contribution and recording relations rarely encode direct instrument details. Future reasoning should prioritize direct biographical or equipment-related relations over indirect contribution-based paths when seeking information about an artist's instruments.

Table 8: Case study of TRACE. We highlight the correct answers in Red and the wrong answers in Blue.

Prompt Template for Dynamic Context Generation

Role:
You are an expert assistant for Knowledge Graph Question Answering (KGQA). Your capability lies in transforming structured relation sequences into coherent natural language narratives that capture their semantics in the context of the input question.
Task:
Given a natural language question and a sequence of relations representing a reasoning path, generate a concise and contextually faithful narrative that describes the meaning of this path. The narrative will serve as dynamic context to guide subsequent reasoning steps.
Rules and Constraints:

- **Faithful Representation**: The narrative must accurately reflect the semantics of the provided relations without introducing external knowledge.
- **Conciseness**: Express the reasoning path in a clear and compact natural language form.
- **Context Awareness**: Ensure that the generated narrative maintains coherence with the original question and the evolving reasoning trajectory.

Example:

- **Input:**
  - `Question`: "Where was the director of the movie Titanic born?"
  - `Path Relations`: {"movie.directed_by", "person.place_of_birth"}
- **Output:**

  ```
  "Find the director of the movie Titanic, and then find the birthplace
  of that person."
  ```

- **Input:**
  - `Question`: "What kind of currency does Germany use?"
  - `Path Relations`: {"country.currency_used"}
- **Output:**

  ```
  "Find the currency used by the country Germany."
  ```

Your Task

- `Question`: {question}
- `Path Relations`: {relations_list}

Output:

Figure 3: Prompt template used in the Dynamic Context Generation module to transform relation sequences into natural language narratives.

Prompt Template for Trajectory Summary

Role:
You are a **"Trajectory Analyst"** responsible for analyzing terminated reasoning trajectories in Knowledge Graph Question Answering (KGQA).
Task:
Given a natural language question, an explored reasoning trajectory, and the reason why this trajectory terminated, generate a concise natural language summary that explains: (i) what specific reasoning direction the trajectory attempted, and (ii) why it reached a stopping point (e.g., dead end, repetitive loop, or semantic drift). This summary should describe the specific instance and **must not** be generalized into a reusable rule.
Rules and Constraints:

- **Specificity**: Clearly describe the concrete reasoning direction taken by the trajectory.
- **Termination Awareness**: Explicitly explain why the trajectory stopped (e.g., max depth, irrelevant relations, or lack of feasible expansion).
- **Conciseness**: Keep the summary short and focused on the essential reasoning behavior.
- **No Generalization**: Do not abstract the summary into a general rule or pattern.

Example:

- **Input:**
  - `Question`: "Where was the director of Titanic born?"
  - `Explored Trajectory`: {"movie.directed_by", "person.spouse"}
  - `Reason for Termination`: Max depth reached
- **Output:**

```
"The trajectory attempted to follow the director's spouse,
but this direction drifted away from the original question
about the director's birthplace."
```

Your Task

- `Question`: {question}
- `Explored Trajectory`: {explored_path}
- `Reason for Termination`: {reason_for_termination}

Output:

Figure 4: Prompt template used in the Exploration Generalization module to summarize terminated reasoning trajectories into trajectory-level descriptions.

Prompt Template for Exploration Pattern Extraction

Role:
You are a **"Pattern Recognizer"** responsible for identifying recurring patterns in reasoning trajectories for Knowledge Graph Question Answering (KGQA).
Task:
Given a list of **Trajectory Summaries** derived from recent terminated or stalled reasoning attempts, identify common reasoning patterns, shared unproductive behaviors, or structural pitfalls. Synthesize these observations into concise **Exploration Patterns** that can guide future reasoning away from similar mistakes.
Rules and Constraints:

- **Pattern Abstraction**: Identify recurring trends or shared characteristics across multiple trajectory summaries.
- **Conciseness**: Express the extracted exploration patterns in a brief, compact paragraph.
- **Actionability**: Phrase the patterns as guidance that can warn or steer future reasoning steps.
- **No Case-Specific Details**: Avoid mentioning entity names or instance-specific information.

Example

- **Input:**
  - `Trajectory Summaries`:
    - "The trajectory explored the `spouse` relation but found no link to birthplace."
    - "The trajectory followed `children`, leading to a dead end for location-related queries."
    - "The trajectory checked `sibling`, which was irrelevant to the target information."
- **Output:**

```
"Family-related relations such as spouse, children, and sibling
often lead to unproductive paths for location-focused questions.
Future reasoning should prioritize direct location or biography
relations instead."
```

Your Task

- `Trajectory Summaries`: {trajectory_summaries}

Output:

Figure 5: Prompt template used in the Exploration Generalization module to distill trajectory summaries into reusable exploration patterns.

Prompt Template for Candidate Retrieval (Dual-Feedback Stage 1)

Role:
You are a **"Relation Retriever"** for Knowledge Graph Question Answering (KGQA).
Task:
Given a natural language question, the current reasoning context, and a list of candidate relations, select up to k relations that are most likely to extend the reasoning path toward the correct answer.
Rules and Constraints:

- **Fidelity to Candidates**: Selections must come strictly from the provided `Candidate Relations` list.
- **Quantity Limit**: Return no more than k relations. If fewer are relevant, return only those.
- **Output Format**: The response must be a Python-parseable list of strings. If no relation is relevant, return an empty list `[]`.

Example

- **Input:**
  - `Question`: "Who is the CEO of Tesla?"
  - `Current Context`: "This is the start of the path."
  - `Candidate Relations`: {"organization.leadership", "organization.founders", "organization.headquarters", "organization.industry"}
  - `K`: 2
- **Output:**

```
["organization.leadership", "organization.founders"]
```

Your Task

- `Question`: {question}
- `Current Context`: {context_narrative}
- `Candidate Relations`: {candidate_relations}
- `K`: {k}

Output:

Figure 6: Prompt template used in the Candidate Retrieval stage of Dual-Feedback Re-ranking to select top-k relations.

Prompt Template for Dual-Feedback Re-ranking (Stage 2)

Role:
You are a **"Path Evaluator"** that incorporates contextual coherence and prior exploration experience when ranking candidate relations.
Task:
Given a natural language question, a historical reasoning path, a list of top-k candidate relations, and a summary of past exploration experiences, evaluate the plausibility of each candidate relation as the next reasoning step. Assign a numerical score between 0.0 (bad fit) and 1.0 (perfect fit), reflecting both logical relevance to the question and consistency with effective exploration patterns.
Rules and Constraints:

- **Contextual Coherence**: Evaluate how well each candidate relation extends the current reasoning path toward answering the question.
- **Exploration Awareness**: Deprioritize candidates that resemble previously observed unproductive or low-yield exploration patterns.
- **Output Format**: The response must be a Python-parseable dictionary with candidate relation names as keys and scores as values.

Example:

- **Input:**
  - `Question`: "Where was the director of the movie Titanic born?"
  - `Historical Path`: {`movie.directed_by`}
  - `Candidate Relations`: {`person.place_of_birth`, `person.nationality`, `person.spouse`}
  - `Summary of Exploration Experience`: "Exploration paths focusing on nationality are often too coarse when a specific birthplace is required, and spouse relations rarely contribute to location-based queries."
- **Output:**

```
{"person.place_of_birth": 0.9, "person.nationality": 0.2, "person.spouse": 0.1}
```

Your Task:

- `Question`: {question}
- `Historical Path`: {historical_path}
- `Candidate Relations`: {top_k_relations}
- `Summary of Exploration Experience`: {exploration_experience}

Output:

Figure 7: Prompt template used in the Dual-Feedback Re-ranking module to re-rank candidate relations by integrating contextual coherence and prior exploration experience.