iQUEST: An Iterative Question-Guided Framework for Knowledge Base Question Answering

Shuai Wang
Chalmers University of Technology
Gothenburg, Sweden
shuaiwa@chalmers.se

Yinan Yu
Chalmers University of Technology
Gothenburg, Sweden
yinan@chalmers.se

Abstract

While Large Language Models (LLMs) excel at many natural language processing tasks, they often suffer from factual inaccuracies in knowledge-intensive scenarios. Integrating external knowledge resources, particularly knowledge graphs (KGs), provides a transparent and updatable foundation for more reliable reasoning. Knowledge Base Question Answering (KBQA), which queries and reasons over KGs, is central to this effort, especially for complex, multi-hop queries. However, multi-hop reasoning poses two key challenges: (1) maintaining coherent reasoning paths, and (2) avoiding prematurely discarding critical multi-hop connections. To address these issues, we introduce iQUEST, a question-guided KBQA framework that iteratively decomposes complex queries into simpler sub-questions, ensuring a structured and focused reasoning trajectory. Additionally, we integrate a Graph Neural Network (GNN) to look ahead and incorporate 2-hop neighbor information at each reasoning step. This dual approach strengthens the reasoning process, enabling the model to explore viable paths more effectively. Detailed experiments demonstrate the consistent improvement delivered by iQUEST across four benchmark datasets and four LLMs.

1 Introduction

Large language models (LLMs) have achieved remarkable success across diverse Natural Language Processing (NLP) tasks, yet they often exhibit hallucinations and factual errors in specialized, knowledge-intensive domains (Huang et al., 2024; Martino et al., 2023; Minaee et al., 2024). Fine-tuning LLMs on curated datasets can embed domain-specific knowledge into model parameters, but it is computationally expensive, difficult to update, and opaque in terms of interpretability (Hu et al., 2023). As an alternative, Retrieval-Augmented Generation (RAG) techniques query
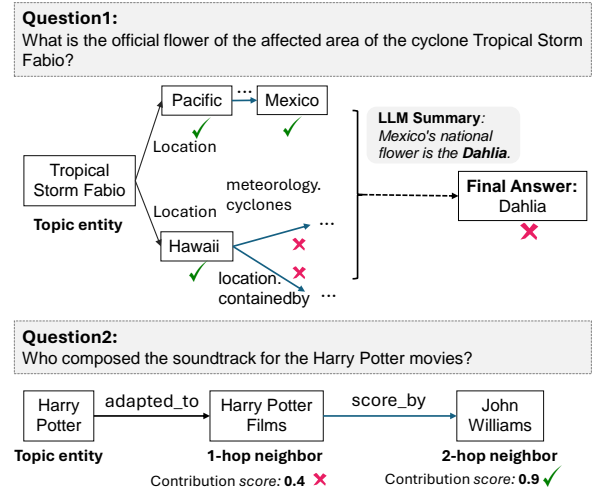

Figure 1: Examples of multi-step reasoning in a knowledge graph, each starting from a topic entity. *Question 1* illustrates how maintaining coherent reasoning paths can be challenging, whereas *Question 2* shows how critical multi-hop connections may be prematurely discarded. Entities marked with ✓ are selected in the reasoning path, and those marked with ✗ are unselected.

external resources during inference, thereby reducing reliance on repeated retraining (Gao et al., 2023). Among these resources, knowledge graphs (KGs) offer structured and trustworthy information that is directly verifiable and maintainable, an essential feature for high-stakes scenarios such as healthcare and autonomous driving (Wen et al., 2024).

In this context, Knowledge Base Question Answering (KBQA) has emerged as a crucial research direction, aiming to query and reason over KGs to answer natural language questions. Complex KBQA tasks often require multi-hop reasoning or multiple constraints (Lan et al., 2022), as illustrated by questions like "In the North Pacific region, what is the official flower of the affected area of the cyclone Tropical Storm Fabio?" Answering such queries necessitates iterative exploration of entities, relations, and constraints within large, heterogeneous KGs.

15616

A straightforward way to handle complex queries is the Chain-of-Thought (CoT) (Wei et al., 2022) framework, which makes each inference step explicit. Building on this idea, some methods treat LLMs as agents that iteratively explore the KG from a topic node (e.g., ToG (Sun et al., 2024), Interactive-KBQA (Xiong et al., 2024)), while others combine a smaller model with an LLM to gather evidence before synthesizing the final answer (e.g., KG-CoT (Zhao et al., 2024)). Another direction focuses on sub-question decomposition, either by inserting delimiters into the original query (Huang et al., 2023) or by using an LLM to break down the query for a fine-tuned smaller model (Yixing et al., 2024). Although these approaches have shown promise, multi-step reasoning over KGs still face two key challenges.

Difficulty in discovering effective reasoning paths. Multi-hop inference demands sustained logical focus. However, as reasoning involves multiple interwoven subproblems, models often struggle to maintain direction, losing track after just a few steps. Additionally, ambiguous or tangential entities (e.g., "Mexico" in Question 1 of Figure 1) can introduce noise, further disrupting the reasoning path and hindering the identification of the correct answer.

Prematurely discarding critical multi-hop connections. Existing methods frequently rely on local (1-hop) relations between the question and a topic entity's neighbors, risking the early elimination of valuable multi-hop paths. For instance, in Question 2 of Figure 1, "Harry Potter Films" might be incorrectly discarded due to a lower immediate relevance score, even though its 2-hop neighbor "John Williams" strongly matches the query. Such oversights can derail the entire reasoning process and lead to incorrect answers.

In complex problem-solving scenarios, people often lose focus if their attention is not guided continuously. However, research indicates that by posing and solving smaller questions, humans can maintain higher levels of attention during the reasoning process, thereby improving decision-making and problem-solving (Salmon and Barrera, 2021; Tofade et al., 2013). Inspired by this insight, we propose to guide LLMs in multi-step KBQA by iteratively posing simpler sub-questions, thereby maintaining a clear reasoning trajectory over multiple hops.

To discover more robust reasoning paths on KGs, we devise an iterative question-guided framework

iQUEST which, at each iteration, generates a sub-question fully answerable from the current context. The LLM then targets that sub-question, retrieves evidence entities, and returns an intermediate answer. Concurrently, we mitigate the brittleness of multi-hop exploration by integrating a Graph Neural Network (GNN) to incorporate semantic information from second-hop neighbors, thereby helping the model look one step ahead and avoid discarding potentially crucial connections.

The key contributions of this paper are summarized as follows:

- We propose a question-guided reasoning framework that differs from prior work focused solely on question decomposition. Instead of decomposing the original question once, our method enables the LLM to iteratively generate a new sub-question at each reasoning step based on the current state, effectively guiding the reasoning process over the knowledge graph.

- We design a GNN-based method to aggregate semantic information from second-hop neighbors, allowing the model to look ahead in the KG at each step and enhancing the robustness and accuracy of multi-hop reasoning.

- We conduct extensive experiments on four benchmark datasets, demonstrating the effectiveness and generalizability of our proposed methods.

2 Related Work

2.1 KBQA

Knowledge Base Question Answering (KBQA) aims to leverage a KG to generate answers. Traditional methods are generally IR-based or SP-based: the former retrieves candidate entities and relations from a KG and then ranks them (Saxena et al., 2020; Dai et al., 2023; Chen et al., 2022), whereas the latter converts natural language questions into executable structured queries (e.g., SPARQL) (Das et al., 2021; Ye et al., 2022).

Recently, LLMs have introduced new possibilities. Several approaches integrate LLMs in different ways: Interactive-KBQA (Xiong et al., 2024) iteratively refines SPARQL queries; ToG (Sun et al., 2024) uses an LLM agent to explore a KG
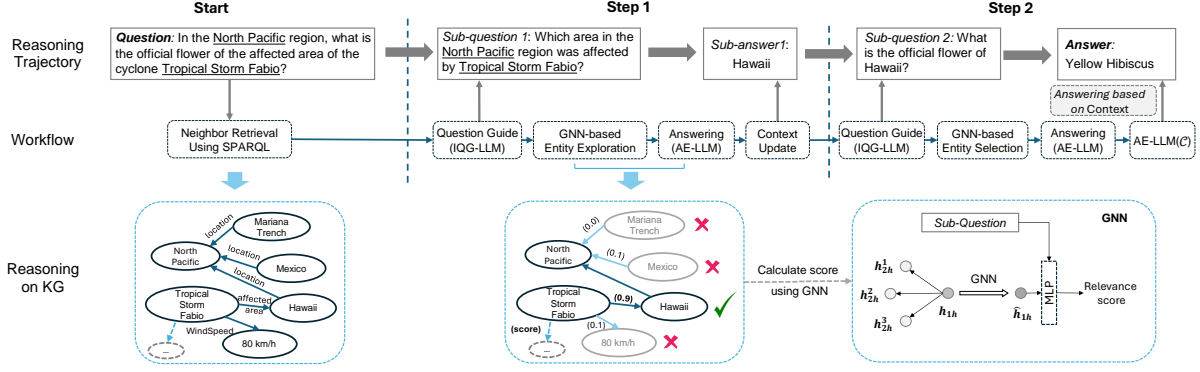
Figure 2: Overview of our framework, illustrating the reasoning trajectory, workflow, and multi-step reasoning process over KGs.

for evidence; FlexKBQA (Li et al., 2024) synthesizes training data for smaller models; KG-CoT (Zhao et al., 2024) fuses small-model reasoning with LLM-based answer finalization. Other work uses tool-like interfaces for external KG reasoning (Jiang et al., 2023) or multi-agent strategies (Zong et al., 2024) to enhance KBQA performance.

2.2 Question Decomposition

A central challenge in complex KBQA is decomposing the query into tractable sub-questions. Early methods used neural architectures (e.g., Seq2Seq or BERT+LSTM) to split questions, but often lacked robustness or altered entity semantics (Zhang et al., 2019; Gu and Su, 2022). Later work introduced explicit delimiters to preserve meaning and sequentially answer sub-questions (Huang et al., 2023). Increasingly, LLMs have been adopted for decomposition and subsequent reasoning (Yixing et al., 2024). However, most focus on segmentation rather than adaptive inference. In contrast, our approach uses targeted intermediate prompts to guide and validate each reasoning step, harnessing the LLM's capabilities more comprehensively.

2.3 Entity Exploration

Many KBQA systems rely on 1-hop neighbors for candidate retrieval, employing text similarity (e.g., BM25), direct semantic relevance, or MLP-based scoring (Sun et al., 2024; Xiong et al., 2024; Zhao et al., 2024; Yixing et al., 2024). These methods often overlook 2-hop neighbors, potentially missing critical inference paths. Our work addresses this gap by explicitly incorporating 2-hop neighbors into the search process, enabling more thorough exploration and improved answer derivation.

3 Problem defination

We consider a KG $\mathcal{G} = \{\langle e, r, e' \rangle \mid e, e' \in \mathcal{E}, r \in \mathcal{R}\}$, where $\mathcal{E}$ and $\mathcal{R}$ represent the sets of entities and relations, respectively. Each triple $\langle e, r, e' \rangle$ encodes a relationship r between entity e and entity e'.

Given a natural language question Q that contains a topic entity e_{topic}, the goal is to identify an answer entity $e_{\text{ans}} \in \mathcal{E}$ within $\mathcal{G}$ that correctly answers Q. Unlike simple KBQA, where e_{ans} is directly connected to e_{topic}, complex KBQA requires multi-hop reasoning, as e_{ans} may be multiple hops away from e_{topic}.

4 Method

Our method, shown in Figure 2 and detailed in Algorithm 1, consists of three modules: (1) *Iterative Question Guidance*, (2) *Two-hop Entity Exploration*, and (3) *Answer Extraction*. Given a complex query, *Iterative Question Guidance* breaks it down into simpler sub-questions, each guiding the next exploration step. *Two-hop Entity Exploration* expands relevant entities and relations by leveraging both 1-hop and 2-hop connections. Finally, *Answer Extraction* integrates the retrieved knowledge to generate the final answer.

4.1 Iterative Question Guidance

Given a multi-step reasoning question Q, our goal is to dynamically decompose it into simpler sub-questions, guiding the knowledge graph exploration step by step. For instance, given the original question: *"In the North Pacific region, what is the official flower of the affected area of the cyclone Tropical Storm Fabio?"*, our method first extracts the most relevant sub-question: *"Which area in the*

Algorithm 1: Question-Guided KBQA

Input: Natural language question Q; Topic entity e_{topic}; Knowledge graph $\mathcal{G}$.

Output: Answer A to the question Q.

Initialization:

$\mathcal{C} \leftarrow \emptyset$ // context

$\mathcal{E} \leftarrow \{ e_{\text{topic}} \}$ // candidate entities

for $i \leftarrow 1$ **to** *MaxIter* **do**

   (1) Subquestion Generation:

   $Q_{\text{sub}} \leftarrow$ IGQ-LLM$(Q, \mathcal{C})$;

   (2) Neighbor Retrieval:

   Nbr $\leftarrow$ SPARQL$(\mathcal{G}, \mathcal{E})$;

   (3) GNN-based Entity Selection:

     (a) Collect 2-hop neighbors:

     $\mathcal{E}_{2h} \leftarrow \bigcup_{e \in \text{Nbr}}$ SPARQL$(\mathcal{G}, e)$;

     (b) Relevance scoring:

     Scores $\leftarrow$ GNN$(\mathcal{E}_{2h}, Q_{\text{sub}})$;

     (c) Top-k update:

     $\mathcal{E} \leftarrow$ Top-k(Scores);

   (4) Subquestion Answering:

   $A_{\text{sub}} \leftarrow$ AE-LLM$(Q_{\text{sub}}, \mathcal{E})$;

   (5) Context Update:

   $\mathcal{C} \leftarrow \mathcal{C} \cup \{ (Q_{\text{sub}}, A_{\text{sub}}) \}$;

   (6) Sufficiency Check:

   if AE-LLM$(\mathcal{C}) ==$ *"sufficient"* **then**

       break   // exit if enough evidence

return AE-LLM$(Q, \mathcal{C})$ // final answer

North Pacific region was affected by Tropical Storm Fabio?".

To achieve this dynamic decomposition, we extract the next sub-question by considering both the reasoning context $\mathcal{C}$ and the original question Q. Here, the context $\mathcal{C}$ consists of previously generated sub-questions $Q_{\text{sub}}^{(i)}$ and their corresponding answers $A_{\text{sub}}^{(i)}$, providing a structured history of reasoning steps. This process is formally described in Step (1) of the Algorithm 1. The extraction process can be formulated as:

$$Q_{\text{sub}}^{(n)} = \text{IQG-LLM}\Big(Q, \mathcal{C} \Big) \qquad (1)$$

where $\mathcal{C} = \big[Q_{\text{sub}}^{(1)}, A_{\text{sub}}^{(1)}, Q_{\text{sub}}^{(2)}, \ldots, A_{\text{sub}}^{(n-1)} \big]$.

Following the Chain-of-Thought paradigm, we iteratively update prompts to reflect new sub-questions and answers. This structured, step-by-step prompting ensures coherent reasoning and preserves contextual consistency. At each step, the LLM also decides whether further decomposition is required. If not, the current sub-question is used

to guide exploration of the knowledge graph, ensuring efficiency.

4.2 Two-Hop Based Entity Exploration

Neighbor Retrieval. To retrieve relevant knowledge for each sub-question, we explore the neighbors of the target entity via SPARQL queries, as shown in Step (2) of Algorithm 1. We employ a generic SPARQL template to extract 1-hop neighbors. Since SPARQL query syntax of retrieving neighbors is fixed, we can reuse a template to systematically obtain them. For instance, if we wish to find all movies directed by Christopher Nolan, we can issue the following SPARQL query:

```
SELECT ?tailEntity
WHERE {
  ns:m.0bxtg ns:film.director.film ?tailEntity .
}
```

where `ns:m.0bxtg` is the Freebase ID for Christopher Nolan, and `ns:film.director.film` is the relevant predicate.

We repeat this process to retrieve all outgoing edges and neighboring nodes for both head and tail entities in the current state of reasoning, ensuring a comprehensive collection of candidate neighbors.

Once we obtain the 1-hop neighbors of the current entity, we score each neighbor according to its relevance to the query. We then select the top k neighbors as the next step in the reasoning process.

Two-Hop Neighbor Aggregation with GNN. While 1-hop neighbors can be useful, they may not provide sufficient semantic context. To incorporate information from 2-hop neighbors, we perform another round of SPARQL queries to retrieve these nodes. We then employ a GNN to aggregate these 2-hop neighborhood signals, as shown in Step (3) in Algotithm 1. Specifically, we first convert textual features of each node (e.g., entity descriptions or relevant text) into dense vector representations using BERT. Let $\mathbf{h}_{1h}$ denote the representation for a 1-hop neighbor e_{1h}, while $\mathbf{h}_{2h}$ stands for a 2-hop neighbor e_{2h}. We adopt a GraphSAGE-like aggregation (Hamilton et al., 2017) to update the representation of each 1-hop neighbor with its 2-hop neighbors:

$$\hat{\mathbf{h}}_{1h} = \sigma\Big(\mathbf{W} \cdot [\mathbf{h}_{1h} || \text{AGG}\{\mathbf{h}_{2h} \mid e_{2h} \in \mathcal{N}(e_{1h})\}] \Big), \qquad (2)$$

where $\mathcal{N}(e_{1h})$ denotes the set of neighbor nodes of e_{1h}, $\mathbf{W}$ is a trainable weight matrix, and $\text{AGG}(\cdot)$ is an aggregation function such as mean-pooling, which can aggregate neighboring nodes without

any numerical limitations. By concatenating the central node's representation $\mathbf{h}_{1h}$ with the aggregated neighbor representation $\mathbf{h}_{2h}$, we preserve the original information of the central node and mitigate the risk of diluting its identity.

Relevance Classification. After updating the representation of each 1-hop neighbor to $\hat{\mathbf{h}}_{1h}$, we concatenate it with the sub-question representation Q_{sub}, forming a combined vector $\mathbf{h}$. We then feed $\mathbf{h}$ through a two-layer MLP to perform a binary classification (relevant vs. irrelevant). Formally,

$$\hat{\mathbf{y}} = \text{Softmax}\big(\mathbf{W}_2\,\sigma(\mathbf{W}_1\,\mathbf{h} + \mathbf{b}_1) + \mathbf{b}_2\big), \quad (3)$$

where $\mathbf{W}_1, \mathbf{W}_2$ are weight matrices, $\mathbf{b}_1, \mathbf{b}_2$ are biases, and $\sigma(\cdot)$ is a nonlinear activation function such as ReLU. The Softmax function ensures that the output $\hat{\mathbf{y}}$ represents the probability of the neighbor being relevant or irrelevant. To obtain a normalized relevance score, we take the probability corresponding to the "relevant" class:

$$score = \hat{\mathbf{y}}[1] \in (0, 1). \quad (4)$$

Finally, we select the top-k entities with the highest scores as the supporting evidence for answering the question.

We use cross-entropy loss with one-hot encoding for training:

$$L = -\sum_{i=1}^{2} y_i \log(\hat{y}_i). \quad (5)$$

Here, y_i is the ground-truth label for class i, and $\hat{y}_i$ is the predicted probability of class i.

4.3 Answer Extraction LLM (AE-LLM)

After selecting the top-k most relevant entities, we use an LLM to answer the current sub-question, following steps (4)–(6) in Algorithm 1. Specifically, we construct a prompt instructing the LLM to generate an answer based on both the retrieved knowledge and its internal knowledge. The generated answer is then incorporated into the context $\mathcal{C}$ along with the sub-question. Next, we assess whether $\mathcal{C}$ contains sufficient information to directly answer the original question. If so, we prompt the LLM to synthesize all sub-questions and answers to produce the final response. Otherwise, we iteratively generate additional sub-questions, retrieve relevant entities, and obtain answers until enough information is accumulated.

5 Experiments

Datasets We evaluate our approach on four standard KBQA datasets, all of which require multi-hop reasoning: ComplexWebQuestions (CWQ) (Talmor and Berant, 2018), WebQuestionsSP (WebQSP) (Yih et al., 2016), WebQuestions (Berant et al., 2013), and GrailQA (Gu et al., 2021). Among them, GrailQA is also used to assess the generalization ability of the retrieval model. All these datasets are based on the Freebase knowledge graph (Bollacker et al., 2008). Following previous studies, we adopt Hit@1 score as the primary evaluation metric.

Implementation Details. For the LLMs, we utilize GPT-4o (`gpt-4o-2024-05-13`), one of the most advanced general-purpose LLMs, alongside DeepSeek-R1 (70B), a recently released, powerful open-source reasoning model (Guo et al., 2025). Additionally, we incorporate LLaMA 3.1-70B (hereafter referred to as LLaMA 70B) and a smaller-scale model, LLaMA 3.2-3B (referred to as LLaMA 3B), for comparative analysis.

For the GNN-based reasoning module, we use `bert-base-uncased` as the encoder, with a hidden dimension of 768. The GNN itself has a hidden dimension of 128. During inference, we retrieve the top-3 most relevant entities as supporting evidence to answer the question. Since the GNN evaluates the relevance between entities and the question independently at each inference step, we train its parameters using single-hop reasoning samples from the training datasets. Negative samples are generated through random negative sampling.[1]

5.1 Main results

Table 1 presents our experimental results across four standard KBQA benchmarks. Our iQUEST achieves state-of-the-art performance on WebQSP and WebQuestions and ranks second on CWQ and GrailQA. Compared to baselines that do not fine-tune LLMs, such as ToG and Interactive-KBQA, our approach demonstrates superior reasoning over large knowledge graphs. Interactive-KBQA employs Mistral-13B, whose relatively small size limits its performance, while ToG benefits from GPT-4 through an API. Despite these differences, our method significantly outperforms both, underscoring its effectiveness without requiring LLM fine-tuning.

---

[1]https://github.com/Wangshuaiia/iQUEST/tree/main

Method	Multi-hop reasoning			Generalization
	CWQ	WebQSP	WebQuestion	GrailQA
Question-Decomposion (Huang et al., 2023)	72.8	-	-	-
Chain-of-Question (Yixing et al., 2024)	**78.8**	78.10	-	-
KG-CoT (Zhao et al., 2024)	62.3	<u>84.90</u>	<u>68.00</u>	-
FlexKBQA (Li et al., 2024)	-	46.20	-	68.90
Interactive-KBQA (Xiong et al., 2024)	49.07	71.2	-	-
ToG (Sun et al., 2024)	69.5	82.6	57.90	**81.4**
iQUEST (GPT-4o based)	<u>73.85</u>	**88.93**	**81.23**	<u>73.52</u>

Table 1: Performance comparison of different methods. **Best results** in bold and <u>second-best</u> in underline. All the comparison results are taken from their corresponding papers.

For the GrailQA dataset (64k samples), ToG randomly selected 1,000 instances to reduce computation, and we followed the same strategy for fairness, though results may vary slightly due to different samples. On WebQuestions, where ambiguity is common in real user queries, our approach uses the original question as guidance, achieving a 23% performance gain over ToG.

Among fine-tuned approaches, FlexKBQA, KG-CoT, Chain-of-Question, and Question-Decomposition deliver competitive results. Chain-of-Question leverages GPT-3.5-turbo for reasoning and a fine-tuned T5 model for question decomposition, achieving strong performance. Nevertheless, our method maintains an advantage across most datasets. Overall, iQUEST excels in multi-hop reasoning and generalization. Instead of fine-tuning an LLM, it trains a GNN for entity exploration, enhancing adaptability while simplifying training.

5.2 Ablation Study

5.2.1 Assumptions

Ablation studies on LLM systems are challenging due to their black-box nature. To analyze the impact of our design choices, we explicitly outline our assumptions based on prior knowledge and empirical studies presented in the literature:

A1. The most relevant capabilities of an LLM in terms of QA tasks are *reasoning* and *internal knowledge*.

A2. LLMs with significantly smaller parameter sizes generally exhibit lower overall capability. For example, LLaMA 3B performs worse than LLaMA 70B and DeepSeek-R1 (70B), while GPT-4o is considered the most capable LLM.

A3. We assume that model size serves as a sufficient proxy for internal knowledge. That is, LLMs of similar size tend to have comparable levels of internal knowledge.

A4. Based on prior empirical evidence and available information, we assume that DeepSeek-R1 surpasses LLaMA-70B in reasoning due to its reasoning-focused pretraining, and DeepSeek-R1 has at least comparable reasoning capability to GPT-4o. we summarize our assumptions about each LLM in the following Table.

Model	Reasoning	Internal Knowledge	Overall
LLaMA 3B	Low	Low	Low
LLaMA 70B	Medium	Medium	Medium
DeepSeek-R1	High	Medium	Medium
GPT-4o	High	High	High

5.2.2 Experimental Design

In this paper, we introduce three key components (C1-C3) and conduct corresponding ablations to analyze their impact. The complete ablation results can be found in Table 6 (Appendix A).

C1. KG Augmentation We integrate a KG with a GNN-based forward-looking retrieval mechanism to complement the LLM's internal knowledge. We investigate the following aspects:

- **C1.1 KG effectiveness:** Prior studies show that KGs effectively enrich knowledge. While this is not our primary focus, we conduct an ablation where we exclude the KG to replicate this conclusion.

- **C1.2 GNN-based retrieval:** Retrieving relevant neighbors from the KG is challenging due to the abundance of semi-relevant nodes. Excluding too many may omit crucial information, while including too many introduces noise and reduces efficiency. We evaluate the effectiveness of our GNN-based forward-looking two-hop neighbor retrieval model in balancing this trade-off.

Model Combination	Dataset			
	CWQ	WebQSP	WebQuestion	GrailQA
LLaMA 3B (AE) + GPT-4o (IQG)	20.14	40.42	48.65	32.87
LLaMA 3B (AE) + GPT-4o (IQG) + GNN	23.66 (+3.52)	43.73 (+3.31)	50.11 (+1.46)	34.19 (+1.32)
LLaMA 70B (AE) + GPT-4o (IQG)	47.24	83.07	74.61	58.46
LLaMA 70B (AE) + GPT-4o (IQG) + GNN	50.30 (+3.06)	84.36 (+1.29)	76.65 (+2.04)	60.45 (+1.99)
DeepSeek-R1 (AE) + GPT-4o (IQG)	52.44	80.43	74.63	58.23
DeepSeek-R1 (AE) + GPT-4o (IQG) + GNN	55.64 (+3.20)	83.21 (+2.78)	77.87 (+3.24)	60.27 (+2.04)
GPT-4o (AE) + GPT-4o (IQG)	68.42	88.10	80.20	69.30
GPT-4o (AE) + GPT-4o (IQG) + GNN	73.85 (+5.43)	88.93 (+0.83)	81.23 (+1.03)	73.52 (+4.22)

Table 2: GNN-based search across AE-LLMs and datasets.

Model Combination	Dataset			
	CWQ	WebQSP	WebQuestion	GrailQA
GPT-4o (No IQG - Baseline)	63.34	84.25	77.72	67.42
GPT-4o (AE) + LLaMA 3B (IQG)	64.95 (+1.61)	85.70 (+1.45)	78.46 (+0.74)	67.69 (+0.27)
GPT-4o (AE) + LLaMA 70B (IQG)	67.46 (+4.12)	87.01 (+2.76)	79.78 (+2.06)	68.76 (+1.34)
GPT-4o (AE) + DeepSeek-R1 (IQG)	68.16 (+4.82)	87.85 (+3.60)	80.11 (+2.39)	69.45 (+2.03)
GPT-4o (AE) + GPT-4o (IQG)	68.42 (+5.08)	88.10 (+3.85)	80.20 (+2.48)	69.30 (+1.88)
GPT-4o (AE) + GPT-4o (IQG) + GNN	73.85 (+10.51)	88.93 (+4.68)	81.23 (+3.51)	73.52 (+6.10)

Table 3: Performance differences compared to GPT-4o (No IQG as baseline) across datasets.

C2. Iterative Question Guidance (IQG-LLM)
IQG-LLM serves as an interface between the KG and the answer-generation LLM, effectively offloading the reasoning burden. To evaluate its impact, we conduct the following ablations:

- **C2.1 Removing IQG-LLM:** We assess the impact of directly generating answers without this intermediate reasoning module.

- **C2.2 Reasoning importance:** We hypothesize that IQG-LLM primarily enhances reasoning. To test this, we examine whether stronger reasoning LLMs perform better in this role. Specifically, if the IQG-LLM has strong reasoning capabilities while the AE-LLM does not, the IQG-LLM should improve the AE-LLM's QA performance more than an IQG-LLM with weaker reasoning abilities.

C3. Answer Extraction LLM (AE-LLM)
AE-LLM takes the question and retrieved context as input and is responsible for understanding and summarizing relevant information. We explore the following hypotheses:

- **C3.1 Internal knowledge vs reasoning requirement:** We evaluate the extent to which AE-LLM relies on its internal knowledge vs reasoning. Our hypothesis is that if IQG-LLM provides strong reasoning supports, the need for AE-LLM to independently reason is significantly reduced. However, the internal knowledge still plays an important role.

5.2.3 Results
In this section, we examine the impact of these components by systematically removing or altering them. Given the assumptions presented in Section 5.2.1, we present our ablation results for each component C1–C3 in detail.

C1.1 KG Effectiveness
To validate the importance of KG augmentation, we remove the KG and observe its impact across different models (rows without KG vs with KG). The results in Table 6 show a substantial drop in performance. For example, GPT-4o (without KG) achieves only 40.14% on CWQ, compared to 63.34% with KG, indicating that even the strongest LLMs significantly benefit from structured external knowledge. This can be observed across all datasets for larger LLMs (i.e., all LLMs except for LLaMA 3B). Interestingly, for LLaMA 3B, the QA capability decreases when the KG is present. This suggests that smaller models may struggle to effectively integrate external knowledge and may even be hindered by conflicting or irrelevant KG information.

C1.2 GNN-based Two-Hop Neighbor Retrieval
Our GNN-based retrieval model addresses the trade-off between retrieving too many semi-relevant nodes (introducing noise) and excluding too many (losing critical information by looking one step ahead and incorporating semantic information from second-hop neighbors. Table 2 demon-

IQG-LLM	Dataset			
	CWQ	WebQSP	WebQuestion	GrailQA
DeepSeek-R1 (baseline)	46.15	82.64	73.41	57.61
LLaMA 70B (weaker reasoning)	47.13 (+0.98)	83.15 (+0.51)	74.36 (+0.95)	58.57 (+0.96)
GPT-4o (equivalent reasoning)	47.24 (-0.11)	83.07 (+0.08)	74.61 (-0.25)	58.46 (+0.11)

Table 4: Performance comparison for LLaMA 70B as AE-LLM across different IQG-LLM models. Numbers in the parentheses indicate relative change from DeepSeek-R1 as AE-LLM (baseline).

AE-LLM	Dataset			
	CWQ	WebQSP	WebQuestion	GrailQA
DeepSeek-R1 (baseline)	55.64	83.21	77.87	60.27
LLaMA 70B (weaker reasoning)	50.3 (-5.34)	84.36 (+1.15)	76.65 (-1.22)	60.45 (+0.18)
GPT-4o (stronger internal knowledge)	73.85 (+18.21)	88.93 (+5.72)	81.23 (+3.36)	73.52 (+13.25)

Table 5: Impact of AE-LLM selection on QA performance using GPT-4o as IQG-LLM with GNN-based retrieval. Numbers in the parentheses indicate relative change from DeepSeek-R1 as AE-LLM (baseline).

strates that this mechanism consistently improves performance across all datasets and model configurations. Even GPT-4o (AE) + GPT-4o (IQG), the strongest model pairing, benefits from GNN-based search, particularly on CWQ (+5.43%) and GrailQA (+4.22%). This highlights the importance of fine-grained retrieval even for highly capable models, particularly for complex multi-hop reasoning and generalization tasks. While stronger AE-LLMs already perform well, they still leverage improved knowledge selection to refine answer accuracy.

C2.1 Removing IQG-LLM

IQG-LLM plays a crucial role in structuring multi-hop reasoning by breaking complex questions into intermediate steps. As shown in Table 6, removing IQG-LLM leads to performance drops across all datasets. Particularly, we illustrate this for the most capable LLM GPT-4o in Table 3. Even using a much smaller LLaMA 3B as IQG-LLM increases the QA performance.

C2.2 Reasoning Importance in IQG-LLM

We expect AE-LLMs with weaker reasoning abilities, such as LLaMA 70B, to gain more from a reasoning-capable IQG-LLM. Given our assumption that DeepSeek-R1 has stronger reasoning than LLaMA 70B and is at least as capable as GPT-4o in reasoning, while GPT-4o has superior internal knowledge and general capabilities, we analyze how different IQG-LLMs with varying reasoning strengths affect LLaMA 70B used as AE-LLM. From Table 4 we see that using LLaMA 70B as AE-LLM, replacing DeepSeek-R1 with LLaMA 70B as IQG-LLM improves performance across all datasets. However, further upgrading to GPT-4o as IQG-LLM results in minimal changes, with

slight improvements on WebQSP (**+0.08%**) but minor drops on CWQ (**-0.11%**) and WebQuestion (**-0.25%**). These results indicate diminishing returns beyond a certain IQG-LLM reasoning threshold.

C3.1 Internal Knowledge Requirement in AE-LLM

Table 5 examines the impact of AE-LLM selection on QA performance when using GPT-4o as IQG-LLM with GNN-based retrieval for the best reasoning and external knowledge support. Replacing DeepSeek-R1 with LLaMA 70B as AE-LLM results in only minor performance variations, suggesting that AE-LLM with weaker reasoning does not significantly affect the final QA outcome. However, upgrading AE-LLM to GPT-4o leads to substantial improvements across all datasets. This indicates that AE-LLM benefits more from stronger internal knowledge rather than reasoning.

6 Conclusion

In this paper, we present iQUEST, an iterative, question-driven framework for multi-step reasoning in complex KBQA. It decomposes complex queries into sub-questions to guide structured reasoning, using a GNN for knowledge aggregation from relevant areas, which supports 2-hop forward-looking reasoning. By combining sub-questions and answers, iQUEST maintains a coherent reasoning path for more accurate results. Carefully designed ablation studies on four benchmark datasets and four LLMs demonstrate that generating question effectively reduces the reasoning burden, improving reasoning efficiency, knowledge extraction, and utilization. These results confirm the effectiveness of iQUEST in handling complex KBQA tasks.

Acknowledgment

Limitations

Despite the effectiveness of our approach, several limitations should be acknowledged. First, our framework employs one LLM for question formulation and guidance, followed by another LLM for answering, which, while improving performance, introduces additional computational overhead. This increased cost may impact scalability, particularly in real-time or resource-constrained applications. We provide a detailed runtime comparison in Appendix B

Second, our GNN model is limited to capturing information from 2-hop neighbors. While this is sufficient for many cases, it may be inadequate for certain domain-specific KGs where meaningful reasoning requires a broader context. Future work could explore more advanced path search strategies that allow consideration of multi-hop neighbors dynamically, potentially enhancing the model's ability to leverage deeper relational structures within KGs.

References

Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. 2013. Semantic parsing on freebase from question-answer pairs. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1533–1544.

Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: a collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, pages 1247–1250.

Ziyang Chen, Xiang Zhao, Jinzhi Liao, Xinyi Li, and Evangelos Kanoulas. 2022. Temporal knowledge graph question answering via subgraph reasoning. *Knowledge-Based Systems*, 251:109134.

Rong Dai, Xun Yang, Yan Sun, Li Shen, Xinmei Tian, Meng Wang, and Yongdong Zhang. 2023. Fedgamma: Federated learning with global sharpness-aware minimization. *IEEE Transactions on Neural Networks and Learning Systems*.

Rajarshi Das, Manzil Zaheer, Dung Thai, Ameya Godbole, Ethan Perez, Jay-Yoon Lee, Lizhen Tan, Lazaros Polymenakos, and Andrew Mccallum. 2021. Case-based reasoning for natural language queries over knowledge bases. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 9594–9611.

Yunfan Gao, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yi Dai, Jiawei Sun, and Haofen Wang. 2023. Retrieval-augmented generation for large language models: A survey. *arXiv preprint arXiv:2312.10997*.

Yu Gu, Sue Kase, Michelle Vanni, Brian Sadler, Percy Liang, Xifeng Yan, and Yu Su. 2021. Beyond iid: three levels of generalization for question answering on knowledge bases. In *Proceedings of the Web Conference 2021*, pages 3477–3488.

Yu Gu and Yu Su. 2022. ArcaneQA: Dynamic program induction and contextualized encoding for knowledge base question answering. In *Proceedings of the 29th International Conference on Computational Linguistics*, pages 1718–1731.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, et al. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*.

Will Hamilton, Zhitao Ying, and Jure Leskovec. 2017. Inductive representation learning on large graphs. *Advances in neural information processing systems*, 30.

Zhiqiang Hu, Lei Wang, Yihuai Lan, Wanyu Xu, Ee-Peng Lim, Lidong Bing, Xing Xu, Soujanya Poria, and Roy Lee. 2023. Llm-adapters: An adapter family for parameter-efficient fine-tuning of large language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pages 5254–5276.

Lei Huang, Weijiang Yu, Weitao Ma, Weihong Zhong, Zhangyin Feng, Haotian Wang, Qianglong Chen, Weihua Peng, Xiaocheng Feng, Bing Qin, et al. 2024. A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions. *ACM Transactions on Information Systems*.

Xiang Huang, Sitao Cheng, Yiheng Shu, Yuheng Bao, and Yuzhong Qu. 2023. Question decomposition tree for answering complex questions over knowledge bases. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 12924–12932.

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

Yunshi Lan, Gaole He, Jinhao Jiang, Jing Jiang, Wayne Xin Zhao, and Ji-Rong Wen. 2022. Complex knowledge base question answering: A survey. *IEEE Transactions on Knowledge and Data Engineering*, 35(11):11196–11215.

Zhenyu Li, Sunqi Fan, Yu Gu, Xiuxing Li, Zhichao Duan, Bowen Dong, Ning Liu, and Jianyong Wang. 2024. Flexkbqa: A flexible llm-powered framework for few-shot knowledge base question answering. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pages 18608–18616.

Ariana Martino, Michael Iannelli, and Coleen Truong. 2023. Knowledge injection to counter large language model (llm) hallucination. In *European Semantic Web Conference*, pages 182–185. Springer.

Shervin Minaee, Tomas Mikolov, Narjes Nikzad, Meysam Chenaghlu, Richard Socher, Xavier Amatriain, and Jianfeng Gao. 2024. Large language models: A survey. *arXiv preprint arXiv:2402.06196*.

Angela K Salmon and Maria Ximena Barrera. 2021. Intentional questioning to promote thinking and learning. *Thinking Skills and Creativity*, 40:100822.

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.

Jiashuo Sun, Chengjin Xu, Lumingyuan Tang, Saizhuo Wang, Chen Lin, Yeyun Gong, Lionel Ni, Heung-Yeung Shum, and Jian Guo. 2024. Think-on-graph: Deep and responsible reasoning of large language model on knowledge graph. In *The Twelfth International Conference on Learning Representations*.

Alon Talmor and Jonathan Berant. 2018. The web as a knowledge-base for answering complex questions. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 641–651.

Toyin Tofade, Jamie Elsner, and Stuart T Haines. 2013. Best practice strategies for effective use of questions as a teaching tool. *American journal of pharmaceutical education*, 77(7):155.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837.

Yilin Wen, Zifeng Wang, and Jimeng Sun. 2024. MindMap: Knowledge graph prompting sparks graph of thoughts in large language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 10370–10388.

Guanming Xiong, Junwei Bao, and Wen Zhao. 2024. Interactive-KBQA: Multi-turn interactions for knowledge base question answering with large language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 10561–10582.

Xi Ye, Semih Yavuz, Kazuma Hashimoto, Yingbo Zhou, and Caiming Xiong. 2022. Rng-kbqa: Generation augmented iterative ranking for knowledge base question answering. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 6032–6043.

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.

Peng Yixing, Quan Wang, Licheng Zhang, Yi Liu, and Zhendong Mao. 2024. Chain-of-question: A progressive question decomposition approach for complex knowledge base question answering. In *Findings of the Association for Computational Linguistics ACL 2024*, pages 4763–4776.

Haoyu Zhang, Jingjing Cai, Jianjun Xu, and Ji Wang. 2019. Complex question decomposition for semantic parsing. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4477–4486.

Ruilin Zhao, Feng Zhao, Long Wang, Xianzhi Wang, and Guandong Xu. 2024. Kg-cot: Chain-of-thought prompting of large language models over knowledge graphs for knowledge-aware question answering. In *Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence (IJCAI-24)*, pages 6642–6650. International Joint Conferences on Artificial Intelligence.

Chang Zong, Yuchen Yan, Weiming Lu, Jian Shao, Yongfeng Huang, Heng Chang, and Yueting Zhuang. 2024. Triad: A framework leveraging a multi-role LLM-based agent to solve knowledge base question answering. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 1698–1710, Miami, Florida, USA.

A   Complete ablation results

Table 6 presents a comprehensive performance comparison of various model combinations in multi-hop reasoning and generalization tasks across four benchmark datasets: CWQ, WebQSP, WebQuestion, and GrailQA. The results highlight the effectiveness of different architectures and model integration strategies, particularly focusing on the impact of Knowledge Graphs (KG), Iterative Question Generation (IQG), Answer Extraction (AE) and Graph Neural Networks (GNN)-based information retrieval. Additionally, the evaluation encompasses four large language models (LLaMA 3B, LLaMA 70B, DeepSeek-R1 (70B), and GPT-4o), comparing their ability to enhance reasoning capabilities and generalization performance.

Model Combination	Multi-hop Reasoning			Generalization
	CWQ	WebQSP	WebQuestion	GrailQA
LLaMA 3B (without KG)	10.18	46.50	47.86	18.51
LLaMA 3B (with KG)	14.81	30.51	40.00	29.06
LLaMA 3B (AE) + LLaMA 3B (IQG)	16.65	31.97	41.56	29.73
LLaMA 3B (AE) + LLaMA 70B (IQG)	19.44	37.74	46.58	31.29
LLaMA 3B (AE) + DeepSeek-R1 (IQG)	19.86	38.56	48.11	32.75
LLaMA 3B (AE) + GPT-4o (IQG)	20.14	40.42	48.65	32.87
LLaMA 3B (A) + GPT-4o + GNN	23.66	43.73	50.11	34.19
LLaMA 70B (without KG)	35.85	69.66	58.42	32.48
LLaMA 70B (with KG)	42.65	80.43	71.11	56.35
LLaMA 70B (AE) + LLaMA 3B (IQG)	43.33	81.85	72.15	56.43
LLaMA 70B (AE) + LLaMA 70B (IQG)	46.15	82.64	73.41	57.61
LLaMA 70B (AE) + DeepSeek-R1 (IQG)	47.13	83.15	74.36	58.57
LLaMA 70B (AE) + GPT-4o (IQG)	47.24	83.07	74.61	58.46
LLaMA 70B (A) + GPT-4o + GNN	50.30	84.36	76.65	60.45
DeepSeek-R1 (without KG)	30.15	75.71	67.24	30.15
DeepSeek-R1 (with KG)	50.85	78.79	74.01	57.68
DeepSeek-R1 (AE) + LLaMA 3B (IQG)	51.02	79.13	74.18	57.91
DeepSeek-R1 (AE) + LLaMA 70B (IQG)	51.78	79.43	74.35	58.14
DeepSeek-R1 (AE) + DeepSeek-R1 (IQG)	52.05	79.08	74.56	58.05
DeepSeek-R1 (AE) + GPT-4o (IQG)	52.44	80.43	74.63	58.23
DeepSeek-R1 + GPT-4o + GNN	55.64	83.21	77.87	60.27
GPT-4o (without KG)	40.14	71.04	63.30	33.62
GPT-4o (with KG)	63.34	84.25	77.72	67.42
GPT-4o (AE) + LLaMA 3B (IQG)	64.95	85.70	78.46	67.69
GPT-4o (AE) + LLaMA 70B (IQG)	67.46	87.01	79.78	68.76
GPT-4o (AE) + DeepSeek-R1 (IQG)	68.16	87.85	80.11	69.45
GPT-4o (AE) + GPT-4o (IQG)	68.42	88.10	80.20	69.30
GPT-4o (A) + GPT-4o(IQG) + GNN	73.85	88.93	81.23	73.52

Table 6: Performance comparison of different model combinations in multi-hop reasoning and generalization tasks.

B Comparison of Relative Runtime

B.1 Overview

We present a detailed runtime comparison between our *iterative question-guided framework* and ToG [2], a representative baseline that also leverages LLMs for question decomposition and reasoning over knowledge graphs. Unlike our method, ToG does not incorporate explicit question-guided prompting, making it a suitable point of comparison in both efficiency and effectiveness.

B.2 Evaluation Setup

Both methods were evaluated under identical experimental settings using `GPT-4o` via Azure OpenAI. For each hop category (1-hop, 2-hop, and 3-hop), we randomly sampled 100 questions. The n-hop category indicates that the shortest path between the topic and answer entities in the knowledge graph is n.

B.3 Overall Performance Comparison

Table 7 summarizes the results in terms of the average number of LLM calls, runtime per query (in seconds), and accuracy.

Our approach incurs higher latency due to increased LLM calls, particularly for complex multi-hop queries. However, the added reasoning steps improve the completeness of path exploration, yielding significantly better accuracy, especially in 2-hop and 3-hop settings. In contrast, ToG frequently terminates prematurely in harder cases, sacrificing accuracy for speed.

B.4 Controlled Runtime Analysis (Correct Chains Only)

To isolate runtime differences independent of correctness variance, we conducted an additional analysis limited to cases where both methods successfully produced complete and correct reasoning chains. Results are shown in Table 8.

The difference in LLM calls aligns with our method's structure: it introduces an explicit question-guided step at each reasoning hop, resulting in approximately 3 (1-hop: 1 questioning + 1 reasoning + 1 summary), 5 (2-hop), and 7 (3-hop) calls. In contrast, ToG performs around 2, 3, and 4 calls respectively. Empirical averages match the theoretical design, with our method using 0.81, 2.20, and 3.15 more calls per query for 1-hop, 2-hop, and 3-hop questions, respectively.

Hop	Method	LLM Calls	Runtime (s)	Hit@1
1-hop	ToG	2.32	3.41	87.0
	iQUEST	3.09 (+0.77)	4.65 (+1.24)	88.0 (+1.0)
2-hop	ToG	2.56	3.96	72.0
	iQUEST	5.86 (+3.30)	7.54 (+3.58)	78.0 (+6.0)
3-hop	ToG	3.88	6.03	37.0
	iQUEST	7.93 (+4.05)	10.69 (+4.66)	46.0 (+9.0)

Table 7: Overall runtime and accuracy comparison between ToG and our method.

Hop	Method	LLM Calls	Runtime (s)
1-hop	ToG	2.51	3.60
	iQUEST	3.12 (+0.61)	4.82 (+1.22)
2-hop	ToG	3.65	5.23
	iQUEST	6.05 (+2.40)	8.06 (+2.83)
3-hop	ToG	4.87	7.36
	iQUEST	8.32 (+3.45)	11.04 (+3.68)

Table 8: Runtime comparison on correctly answered cases only.

C   Handling Missing or Incomplete Knowledge Graph Facts

In real-world scenarios, knowledge graphs (KGs) are often incomplete or contain outdated information. To address this challenge, our system employs two complementary strategies to ensure robust question answering despite such limitations:

1. **Leveraging LLM Internal Knowledge:** When the required fact is not available in the KG, IQUEST defers to the pretrained knowledge of the large language model (LLM) by directly querying it. This fallback mechanism proves effective in approximately half of such cases, as demonstrated in the example provided in Response 2.

2. **Retrieving Indirect Supporting Facts:** In cases where the exact fact is missing, IQUEST searches for semantically related facts that can provide indirect support. For example, even if the exact timezone of a location is not found, knowing that *Utah* is in the *United States* allows the model to infer a plausible timezone.

Despite employing these strategies, their effectiveness remains limited. In the following section, we provide a detailed analysis of the impact of missing or incorrect knowledge graph facts.

D   Error Analysis: Missing Entities in the Knowledge Graph

Some sub-questions fail during the reasoning process due to missing entities or incomplete evidence in the knowledge graph (KG). For instance, for the question *"About the school that publishes the Harvard Review, what are its school colors?"*, the generated sub-question retrieves only an unnamed entity ID (`m.01066g18`), which lacks sufficient semantic information to support further reasoning steps.

To better understand the impact of such failures, we manually analyzed and categorized errors based on whether the relevant entity is missing in the KG or exists but lacks a semantic name. The results, broken down by dataset and whether iQUEST answered the question correctly or incorrectly, are summarized in Table 9.

These results show that while LLMs like iQUEST are sometimes able to answer questions using prior knowledge, the incompleteness of the KG, either through missing entities or insufficient semantic labeling, remains a significant obstacle. Addressing this issue is critical for improving the

Table 9: Percentage of sub-question failures due to missing or unnamed entities in the KG.

Dataset	Entity Missing in KG		Entity Exists but Name Missing	
	Correct	Incorrect	Correct	Incorrect
CWQ	12%	17%	6%	13%
WebQSP	6%	4%	2%	4%
WebQuestion	3%	5%	1%	6%
GrailQA	3%	5%	14%	9%

robustness and generalization of KGQA systems.

E Failure Case Analysis

Original Question:
Which child actor played in the movie whose soundtrack is Forrest Gump: Original Motion Picture?

Reasoning Process via Sub-question Decomposition:
To answer this complex question using a knowledge graph, the system first attempts to decompose it into simpler sub-questions. The goal is to resolve intermediate entities (such as the movie) before identifying the actor. The process unfolds as follows:

1. **Sub-question 1:** *What movie is associated with the soundtrack Forrest Gump: Original Motion Picture Score?*
   → No direct answer found in the knowledge graph.

2. **Sub-question 2:** *What is the soundtrack title of the movie Forrest Gump?*
   → No direct answer found.

3. **Sub-question 3:** *What movie has the soundtrack titled Forrest Gump: Original Motion Picture?*
   → No direct answer found.

4. **Sub-question 4:** *What movie features the soundtrack titled Forrest Gump: Original Motion Picture?*
   → Still no answer retrieved.

5. **Sub-question 5:** *What movies did Asa Earl Carter act in?*
   → This sub-question appears off-topic and indicates potential drift in the reasoning path.

6. **Sub-question 6:** *What is the movie associated with the soundtrack Forrest Gump: Original Motion Picture?*
   → No answer obtained, despite being a paraphrase of earlier attempts.

7. **Sub-question 7:** *What movie has the soundtrack titled "Forrest Gump: Original Motion Picture"?*
   → No result returned.

Final Answer Synthesis:
Despite no explicit answer found in the knowledge graph for any sub-question, the system leverages world knowledge to synthesize an answer. It recognizes that *Forrest Gump: Original Motion Picture* is the official soundtrack of the movie *Forrest Gump*.

Further, among the limited child actors featured in *Forrest Gump*, **Haley Joel Osment** is identified as the actor who portrayed Forrest Gump's son in a brief but notable appearance. While not a main character, he is the only prominent child actor in the film.

Answer:
Haley Joel Osment

Analysis:
This failure case highlights a key limitation of our method: when a generated sub-question fails to retrieve an answer from the knowledge graph, the reasoning process may stall or fall into repetitive, unproductive loops. This leads to the generation of redundant sub-questions and ineffective reasoning paths.

Moreover, retrieving irrelevant information from the graph can mislead the LLM to generate off-topic sub-questions, such as "*What movies did Asa Earl Carter act in?*", which are unrelated to the original question.

These observations underscore a major challenge: the incompleteness of the knowledge graph remains a significant bottleneck for our method's effectiveness.