SKRAG: A Retrieval-Augmented Generation Framework Guided by Reasoning Skeletons over Knowledge Graphs

Xiaotong Xu[1,2,3], Yizhao Wang[1], Yunfei Liu[1], Shengyang Li[1,2,3]*
[1]Technology and Engineering Center for Space Utilization, Chinese Academy of Sciences
[2]Key Laboratory of Space Utilization, Chinese Academy of Sciences
[3]University of Chinese Academy of Sciences
xuxiaotong231@mails.ucas.ac.cn;
{yizhaowang, liuyunfei, shyli}@csu.ac.cn

Abstract

In specialized domains such as space science and utilization, question answering (QA) systems are required to perform complex multi-fact reasoning over sparse knowledge graphs (KGs). Existing KG-based retrieval-augmented generation (RAG) frameworks often face challenges such as inefficient subgraph retrieval, limited reasoning capabilities, and high computational costs. These issues limit their effectiveness in specialized domains. In this paper, we propose SKRAG, a novel Skeleton-guided RAG framework for knowledge graph question answering (KGQA). SKRAG leverages a lightweight language model enhanced with the Finite State Machine (FSM) constraint to produce structurally grounded reasoning skeletons, which guide accurate subgraph retrieval. The retrieved subgraph is then used to prompt a general large language model (LLM) for answer generation. We also introduce SSUQA, a KGQA dataset in the space science and utilization domain. Experiments show that SKRAG outperforms strong baselines on SSUQA and two general-domain benchmarks, demonstrating its adaptability and practical effectiveness.

1 Introduction

In some specialized domains such as space science and utilization, relevant questions often involve interdisciplinary knowledge, multi-fact reasoning, and complex relational structures. As a result, traditional keyword-based retrieval methods struggle to meet the demand for accurate and efficient knowledge acquisition, highlighting the need for intelligent question answering (QA) systems. Recently, Large Language Models (LLMs) have shown impressive capabilities in language understanding and generation tasks (Brown et al., 2020; Kojima et al., 2022; Ouyang et al., 2022; Huang and Chang, 2023; Achiam et al., 2023), offering new potential for QA. However, without external knowledge grounding, LLMs are prone to hallucinations and outdated information (Kasai et al., 2023; Huang et al., 2025), limiting their reliability in critical domains. To address these issues, Retrieval-Augmented Generation (RAG) framework has emerged as a promising solution. By integrating external knowledge sources into the generation process, RAG enhances the factual accuracy and interpretability of LLMs outputs (Shuster et al., 2021; Borgeaud et al., 2022; Gao et al., 2023; Liang et al., 2024).

Currently, Knowledge Graph-based RAG (KG-based RAG) methods have gained increasing attention. Knowledge graphs (KGs), which store structured, domain-specific knowledge as triples, are valuable external sources for RAG (Peng et al., 2024a; Sanmartin, 2024; Hu et al., 2024; Xu et al., 2025). Compared to unstructured text, KGs enhance the ability of LLMs to perform reasoning across entities and relations, improve retrieval accuracy, and reduce hallucinations (Pan et al., 2024). The quality of the subgraph retrieved from the KG as input to the final QA LLMs is crucial for both reasoning efficiency and answer accuracy of KG-based RAG. Mainstream methods typically rely on either direct or indirect interaction between LLMs and the KG to identify answer paths (Jiang et al., 2023; Sun et al., 2023; Luo et al., 2023), or use graph neural networks (GNNs) to propagate embeddings within the graph (He et al., 2024; Mavromatis and Karypis, 2024).

Although existing KG-based RAG frameworks have shown promising results in knowledge graph question answering (KGQA) tasks, they still face several critical challenges when applied to domains such as space science and utilization, where the problems are complex, reasoning paths are diverse, and the KGs are relatively sparse. Firstly, when dealing with semantically complex questions and multi-fact reasoning requirements, many step-by-step search methods over KGs-from an entity to its neighbors-often suffer from significant redundancy

* Corresponding author.

Findings of the Association for Computational Linguistics: EMNLP 2025, pages 13983–13994
November 4-9, 2025

and lack of explicit question modeling. This results in the omission of crucial information and inefficiency in reasoning. Secondly, GNNs operate by iteratively aggregating information from neighboring nodes to update node representations. However, in domain-specific KGs that are relatively sparse, many nodes have very few neighbors, leading to limited aggregation and suboptimal performance. Thirdly, in order to enable LLMs to effectively learn from KGs, some high-accuracy methods (Luo et al., 2023, 2024; Wang et al., 2024) rely on fine-tuning large-scale (with $\geq 7B$ parameters) pre-trained language models. However, these strategies often consume substantial local computational resources and require prolonged training time, which limits the efficiency and practicality of QA systems.

To tackle these challenges, we propose SKRAG, a Skeleton-guided RAG framework for KGQA. In particular, SKRAG fine-tunes a lightweight language model to generate explicit reasoning skeletons that capture core reasoning chains, especially in questions involving multiple question entities, thereby enabling accurate and efficient subgraph retrieval. To ensure these skeletons are executable and structurally aligned with the KG, we introduce the Finite State Machine (FSM) constraint mechanism, which guides generation by enforcing path-structure constraints. The subgraph retrieved under the guidance of the reasoning skeleton is then used to prompt a general LLM for final answer generation. By leveraging reasoning skeletons, SKRAG bridges complex reasoning, LLM generation, and KG structure, achieving strong performance with improved adaptability and efficiency.

Our contributions can be summarized as follows:

- We propose SKRAG, a novel and efficient framework that integrates a lightweight LLM with the FSM constraint to generate structurally reasoning skeletons, enabling precise subgraph retrieval and aligning semantic reasoning with KG structure.
- To better evaluate SKRAG's performance and promote the application of KGQA in the domain of space science and utilization, we develop a dedicated benchmark dataset named SSUQA through an efficient and automated construction process.
- Our proposed framework SKRAG achieves superior performance compared to popular baselines on SSUQA as well as two challenging general-domain KGQA benchmarks.

2 Related Work

Traditional KGQA. Traditional KGQA methods mainly fall into two categories: embedding-based and GNN-based methods. Early embedding methods (Miller et al., 2016; Saxena et al., 2020) map entities and relations into low-dimensional vectors using static KG embeddings. TransferNet (Shi et al., 2021) enhances this by introducing dynamic embeddings and explicit path propagation, along with a differentiable relation activation mechanism. GNN-based methods (He et al., 2021; Mavromatis and Karypis, 2022; Zhang et al., 2022) leverage message passing to capture deep entity associations for multi-hop reasoning. UniKGQA (Jiang et al., 2022) further improves this by dynamically adjusting propagation weights and adopting multi-step optimization to better handle complex semantic reasoning.

KG-based RAG. With the rise of LLMs, KG-based RAG methods have shown significant advantages in semantic understanding, reasoning, and interpretability by combining the generative power of LLMs with the structured knowledge of KGs. Wu et al. (2023) proposed the Retrieve-Rewrite-Answer framework, which transforms subgraphs into question-relevant textual descriptions to enhance the reasoning capabilities of LLMs. KD-CoT (Wang et al., 2023) and Chain-of-Question (Peng et al., 2024b) incorporate external KGs to enhance the chain-of-thought (CoT) reasoning ability of LLMs, using structured triples to guide intermediate reasoning steps and improve interpretability and accuracy. GNN-RAG (Mavromatis and Karypis, 2024) and G-Retriever (He et al., 2024) leverage GNNs to retrieve semantically relevant subgraphs from large-scale KGs, which are then passed to LLMs as contextual input. StructGPT (Jiang et al., 2023) and ToG (Sun et al., 2023) treat LLMs as autonomous agents that iteratively interact with KGs to explore reasoning paths. RoG (Luo et al., 2023) proposes a Plan-Retrieve-Generate framework, where a planning module guides the LLM to perform retrieval and faithful reasoning.

3 Methodology

In this section, we present the proposed framework, SKRAG. This framework generates question reasoning skeletons that are faithful to the structure of the KG and make them act as navigational guides to accurately direct the retrieval of subgraphs pertinent to the input question. As illustrated in Figure

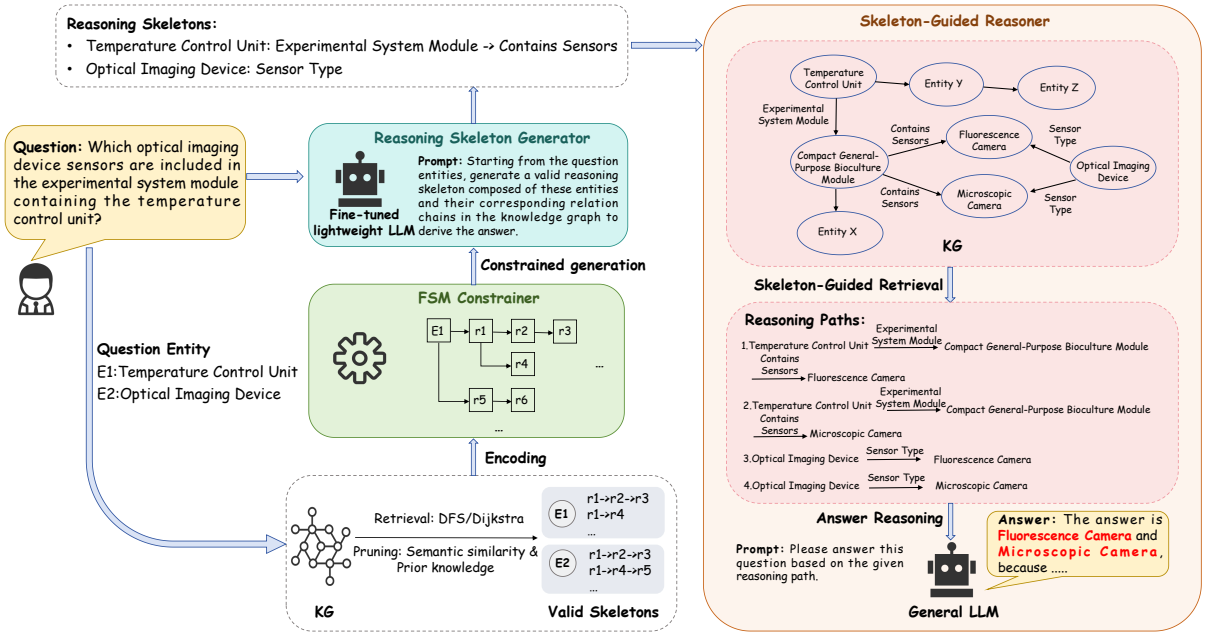

Figure 1: The overall framework of SKRAG, which consists of three core modules: (1) Reasoning Skeleton Generator: Generates reasoning skeletons using a fine-tuned lightweight LLM; (2) FSM Constrainer: Applies FSM constraints to regulate the generation of reasoning skeletons; (3) Skeleton-Guided Reasoner: Leverages the generated skeletons to guide subgraph retrieval and infer the final answer.

1, SKRAG consists of three key components: Reasoning Skeleton Generator, FSM Constrainer, and Skeleton-Guided Reasoner. The following subsections provide a detailed explanation of each module.

3.1 Preliminaries

Knowledge Graphs (KGs) organize rich information in a structured form and can be represented by a set of triples: $\mathcal{G} = \{(e_h, r, e_t) \mid e_h, e_t \in \mathcal{E},\ r \in \mathcal{R}\}$, where $\mathcal{E}$ and $\mathcal{R}$ represent the set of entities and relations, respectively. Each triple (e_h, r, e_t) represents a fact by connecting a head entity h and a tail entity t through a directed relation r.

Reasoning Skeletons represent the starting point and the inferential process of a question, which consist of a topic entity and the relation chain within the path originating from this entity: $s = (e_0 : r_1 \rightarrow r_2 \rightarrow \cdots \rightarrow r_l)$, where $e_0 \in \mathcal{E}$ denotes the starting entity, $r_i \in \mathcal{R}$ denotes the i-th relation in the relation chain and l denotes the length of the relation chain.

Reasoning Paths are the instances of a reasoning skeleton s in KGs: $p_s = e_0 \xrightarrow{r_1} e_1 \xrightarrow{r_2} \ldots \xrightarrow{r_l} e_l$, where $e_0 \in \mathcal{E}$ denotes the starting entity, $e_i \in \mathcal{E}$ denotes the i-th entity and $r_i \in \mathcal{R}$ denotes the i-th relation in the relation chain.

Knowledge Graph Question Answering (KGQA) aims to answer questions by integrating and reasoning over knowledge derived from KGs. Given a natural language question q and a KG $\mathcal{G}$, the task of KGQA is to reason a set of answers $a_i \in \mathcal{A}_q$ which correspond to the entities in $\mathcal{G}$ and can correctly answer q.

3.2 Reasoning Skeleton Generator

We propose the reasoning skeleton that explicitly decouples multi-entity reasoning flows, ensuring both retrieval precision and completeness.The skeleton enables a concise representation of the reasoning process, substantially reducing redundancy and facilitating more efficient constraint decoding.

LLMs show great potential in generating reasoning skeletons due to their strong language understanding and generation abilities. However, their limited awareness of KG structure and the semantics of entities and relations hampers effectiveness. To address this, we fine-tune a lightweight LLM to better capture these semantics, enabling faithful skeleton generation with low computational cost. To ensure that the reasoning skeletons generated by LLMs closely approximate the ground-truth skeletons, we aim to minimize the discrepancy between

the reasoning skeletons generated by the LLMs and the ground-truth reasoning skeletons.

Construction of Supervised Data. Given a question q of the training set, its question entity set $\mathcal{Q}$ and its answer set $\mathcal{A}$, we could find the shortest reasoning path set $\mathcal{P}^* = \left\{p = e_q \xrightarrow{r_1} e_1 \xrightarrow{r_2} \ldots \xrightarrow{r_l} e_a \mid e_q \in \mathcal{Q},\ e_a \in \mathcal{A}\right\}$, from which we could extract the shortest reasoning skeleton set $\mathcal{S}_i^* = \bigcup_{e_a \in \mathcal{A}} \{s \mid s$ is a shortest skeleton from e_q^i to $e_a\}$ for each question entity $e_q^i \in \mathcal{Q}$. Since there may exist multiple shortest skeletons from each question entity to the answer entity, we generate candidate combinations of the shortest reasoning skeletons by applying the Cartesian product to the shortest skeletons of all question entities: $\mathcal{C}^* = \mathcal{S}_1^* \times \mathcal{S}_2^* \times \cdots \times \mathcal{S}_m^*$, where m denotes the number of question entities in the question. Therefore, each supervision instance corresponds to a pairing of the question q and a skeleton combination c_k: Supervised Data $= \{(q, c_k) \mid c_k \in \mathcal{C}^*\}$. To facilitate the understanding of LLMs, we format the skeleton combinations into sentence strings with the following structure: $<$ SK $><$ ENTITY $> e_q^1 <$ /ENTITY $>$: $r_1^1 \rightarrow r_2^1 \ldots <$ SEP $><$ ENTITY $> e_q^2 <$ /ENTITY $>$: $r_1^2 \rightarrow r_2^2 \ldots <$ SEP $> \ldots <$ /SK $>$. Due to the instruction-following capability of LLMs (Wei et al., 2021), we design a simple instruction template to prompt the LLM to generate reasoning skeletons, as illustrated in Figure 4.

Reasoning Skeleton Generation Optimization. We approximate candidate combinations of all reasoning skeletons that connect the question entities to the answer entities $\mathcal{C}$ by using candidate combinations of the shortest reasoning skeletons $\mathcal{C}^* \subseteq \mathcal{C}$ (Zhang et al., 2022), so the posterior distribution $Q(c)$ can be formally approximated as:

$$Q(c) \simeq Q(c \mid q, \mathcal{G}) \simeq \begin{cases} \frac{1}{|\mathcal{C}^*|}, & \text{if } c \in \mathcal{C}^*, \\ 0, & \text{otherwise,} \end{cases} \quad (1)$$

where we assume a uniform distribution over $\mathcal{C}^*$. Therefore, the KL divergence can be calculated as (Luo et al., 2023):

$$\begin{aligned} \mathcal{L} &= D_{\mathrm{KL}}\left(Q(c) \| P_\theta(c \mid q)\right) \\ &\simeq D_{\mathrm{KL}}\left(Q(c \mid q, \mathcal{G}) \| P_\theta(c \mid q)\right) \\ &\simeq -\frac{1}{|\mathcal{C}^*|} \sum_{c \in \mathcal{C}^*} \log P_\theta(c \mid q). \end{aligned} \quad (2)$$

Our optimization objective is to minimize the above KL divergence by maximizing the probability of LLMs generating faithful reasoning skeleton combinations. Therefore, the optimization of $\mathcal{L}$ can be achieved as:

$$\begin{aligned} \arg\max_\theta \frac{1}{|\mathcal{C}^*|} \sum_{c \in \mathcal{C}^*} \log P_\theta(c \mid q) = \\ \frac{1}{|\mathcal{C}^*|} \sum_{c \in \mathcal{C}^*} \log \prod_{i=1}^{m} P_\theta\left(s_i \mid q\right), \end{aligned} \quad (3)$$

where $P_\theta(c \mid q)$ denotes the prior distribution of generating faithful reasoning skeleton combination c, and $P_\theta\left(s_i \mid q\right)$ denotes the probability of each skeleton in c generated by LLMs.

3.3 FSM Constrainer

Although fine-tuning enhances the LLM's ability to generate KG-based reasoning skeletons, its linguistic flexibility may still lead to semantically plausible but structurally invalid outputs (Valmeekam et al., 2023). To address this, we introduce the FSM constrainer that enforces structural validity by restricting generation to follow actual KG connectivity patterns.

FSM is a mathematical structure used to model system state transitions (Russell and Norvig, 2016). It consists of a set of states, transition rules, an initial state, and accepting states, enabling orderly switching between states based on input symbols. In natural language processing tasks, the FSM is often employed to impose structural constraints on the generation process, ensuring that the output adheres to predefined valid structures and thereby enhancing the accuracy and controllability of the generated content (Cabrera et al., 2024). In our method, we encode valid reasoning skeletons from the KG into the FSM to constrain the generation process of LLMs.

Retrieval of Valid Skeletons. In practical QA scenarios, given a question q of the test set and its question entity set $\mathcal{Q}$, we retrieve neighborhood reasoning paths within an h-hop range from each question entity. If $h < 4$, we adopt the Depth-First Search (DFS) algorithm to extract the neighborhood paths. However, when $h \geq 4$, the increased number of hops may lead to excessive path redundancy; therefore, we adopt the Dijkstra algorithm to reduce redundant paths. The process can be formulated as:

$$\mathcal{P}_i = \begin{cases} \mathrm{DFS}\left(e_q^i, \mathcal{G}, h\right), & h < 4, \\ \mathrm{Dijkstra}\left(e_q^i, \mathcal{G}, h\right), & h \geq 4, \end{cases} \quad (4)$$

where $e_q^i \in \mathcal{Q}$ denotes the i-th question entity. Then we could extract the valid reasoning skeleton set $\mathcal{S}_i = \left\{ \left(e_q^i : r_1^{(k)} \rightarrow r_2^{(k)} \rightarrow \cdots \rightarrow r_l^{(k)} \right) \mid k \in K \right\}$ from $\mathcal{P}_i$.

Pruning of Valid Skeletons. To reduce storage and improve the quality of reasoning skeletons encoded in the FSM, we apply a pruning strategy that combines semantic similarity and prior knowledge. For each question entity e_q^i, we compute a fused score for each skeleton $s \in \mathcal{S}_i$ as:

$$\begin{aligned} \text{Score}(s) = \lambda \cdot \cos(\text{S-BERT}(q), \text{S-BERT}(s)) \\ + (1 - \lambda) \cdot \text{Prior}(s), \end{aligned} \tag{5}$$

where S-BERT denotes the Sentence-BERT model, $\text{Prior}(s) \in \{0, 1\}$ represents a prior score indicating whether skeleton s appears in the training set, and $\lambda \in [0, 1]$ is a tunable weight balancing semantic relevance and prior knowledge. After computing the scores, we select the top-k highest-scoring skeletons to form a pruned skeleton set $\mathcal{S}_i^p$. Since multiple skeletons may exist when starting from each question entity, we finally obtain candidate combinations of reasoning skeletons to be stored in the FSM by applying the Cartesian product over the pruned skeletons of all question entities: $\mathcal{C}^p = \mathcal{S}_1^p \times \mathcal{S}_2^p \times \cdots \times \mathcal{S}_m^p$. Each combination $c \in \mathcal{C}^p$ is formatted as a string $z =< \text{SK} >< \text{ENTITY} > e_q^1 < /\text{ENTITY} >: r_1^1 \rightarrow r_2^1 \ldots < \text{SEP} >< \text{ENTITY} > e_q^2 < /\text{ENTITY} >: r_1^2 \rightarrow r_2^2 \ldots < \text{SEP} > \ldots < /\text{SK} >$.

Constrained Generation with FSM. Given a set of valid skeleton strings $\mathcal{Z} = \{z_1, z_2, \ldots, z_n\}$, where each skeleton z_i is a tokenized sequence of token IDs $z_i = [t_1, t_2, \ldots, t_k]$, we construct the FSM to enforce generation constraints. The FSM is formalized as a prefix-to-next-token transition mapping: $\{(t_1, \ldots, t_{i-1}) \mapsto t_i\}, \quad \forall i \in \{1, \ldots, k\}$. This mapping defines, for each prefix, the set of permissible next tokens. Once the full sequence is matched, the corresponding terminal state transitions to an empty set, signaling the end of a valid skeleton. During decoding, this FSM is used to restrict the token generation space at each time step. Let the current decoding prefix be $x_{1:t-1}$, and let $\mathbf{l}_t \in \mathbb{R}^{|\mathcal{V}|}$ denote the vocabulary logits predicted by the model. The FSM determines the valid continuation set as $\mathcal{V}_t^{\text{valid}} = \text{FSM}(x_{1:t-1})$. We then apply a masking operation over the logits to filter out invalid tokens:

$$\tilde{\mathbf{l}}_t[v] = \begin{cases} \mathbf{l}_t[v], & \text{if } v \in \mathcal{V}_t^{\text{valid}}, \\ -\infty, & \text{otherwise.} \end{cases} \tag{6}$$

This masked logit vector $\tilde{\mathbf{l}}_t$ is passed to the next step of decoding, ensuring that only FSM-consistent tokens are considered. In doing so, the model's generation is strictly constrained to the space of syntactically valid reasoning skeletons, thereby improving structural correctness and reducing the risk of generating semantically invalid or unexecutable skeletons.

3.4 Skeleton-Guided Reasoner

Skeleton-Guided Retrieval. We leverage the constraint-generated reasoning skeletons to efficiently guide the retrieval of relevant subgraphs from the KG. Given a question q and a reasoning skeleton $s = (e_q : r_1 \rightarrow r_2 \rightarrow \cdots \rightarrow r_l)$, we retrieve the reasoning paths p from KG $\mathcal{G}$. The retrieval process can be conducted by searching for paths in $\mathcal{G}$ guided by the reasoning skeleton, based on the question entity and its relation chain in the skeleton. These paths start from the question entity e_q and follow the relation chain $r_1 \rightarrow r_2 \rightarrow \cdots \rightarrow r_l$, which can be formalized as:

$$\begin{aligned} \mathcal{P} = \Big\{ p(e_q, e_*) \;\Big|\; p(e_q, e_*) = e_q \xrightarrow{r_1} e_1 \xrightarrow{r_2} \cdots \\ \xrightarrow{r_l} e_{a*},\ p(e_q, e_*) \in \mathcal{G} \Big\}. \end{aligned} \tag{7}$$

Answer Reasoning. The answers to complex questions often depend on integrated reasoning over multiple reasoning paths. To address this, our module fully exploits the powerful inference capabilities of general LLMs by incorporating a set of retrieved reasoning paths $\mathcal{P} = \{p_1, p_2, \ldots, p_n\}$ into the input prompt. By providing $\mathcal{P}$ as formatted context, the LLM is guided to perform interpretable and constraint-aware reasoning, which helps mitigate hallucinations and enhances the accuracy of the final answer $\mathcal{A}$ generated in response to q. The detailed prompt can be found in Figure 5.

4 Domain-specific Dataset: SSUQA

To support comprehensive evaluation and advance research in KGQA within the domain of space science and utilization, we build a valuable dataset named SSUQA based on the domain-specific KG (Liu et al., 2023). The dataset is constructed through an efficient automated method leveraging

Type	Path			Multi-entity	Aggregation
	1-hop	2-hop	3-hop		
Count	3,500	5,000	5,000	4,000	500

Table 1: Distribution of SSUQA.

Chain-of-Thought (CoT) (Wei et al., 2022) prompting with LLMs. Compared with labor-intensive manual annotation, our method enables scalable and high-quality data generation. The construction process consists of two main stages: Triple Reasoning Paths Generation and CoT-based Prompt Construction.

Triple Reasoning Paths Generation. This stage aims to automatically mine logically valid reasoning paths from the KG to support question generation. We start from each entity and use DFS to extract all 1-hop to 3-hop paths. Based on these, we define three types of constraints:

- **Path Constraints:** Limit hop counts and include both single-answer paths and merged paths sharing the same head and relation to support multi-answer questions.
- **Multi-entity Constraints:** Combine paths from different heads that lead to overlapping tail entities to introduce multi-entity reasoning.
- **Aggregation Constraints:** Use paths with multiple tail entities, with the answer being the count of these entities.

To reduce cost and ensure semantic diversity, we encode each path using Sentence-BERT (Reimers and Gurevych, 2019) and apply K-means clustering (Hartigan and Wong, 1979) for representative path selection.

CoT-based Prompt Construction. In this stage, we guide LLMs to iteratively transform complex reasoning paths into natural language questions using CoT prompting. The detailed prompt can be found in Figure 6. The process includes:

- **Entity Type Identification.** Identify entity roles and semantics to help LLMs understand KG structure and support later generation.
- **Triple-to-Text Conversion.** Convert reasoning paths into natural language to enhance semantic understanding.
- **Iterative Sub-question Generation.** Extract key segments to form sub-questions, which are progressively merged into a complete and fluent question.

In total, we construct a dataset of 18,000 QA pairs in SSUQA. The detailed question distribution is shown in Table 1.

Methods	SSUQA	
	Hits@1	F1
NSM	56.4	-
TransferNet	63.5	-
Qwen-Plus	72.6	-
DeepSeek-V3	80.3	-
StructGPT	62.9	-
RoG(Qwen2.5-1.5B-Instruct + DeepSeek-V3)	93.0	84.3
GNN-RAG(DeepSeek-V3)	87.8	79.9
GCR(Qwen2.5-1.5B-Instruct + DeepSeek-V3)	81.0	70.6
SubgraphRAG(DeepSeek-V3)	91.7	**88.3**
SKRAG(Ours, Qwen2.5-1.5B-Instruct + DeepSeek-V3)	**97.3**	87.8

Table 2: Performance comparison of different methods on the SSUQA dataset. Bold represents the best result.

5 Experiment

5.1 Experimental Settings

Datasets and Evaluation Metrics. To evaluate the effectiveness of SKRAG, we conduct experiments on SSUQA and two widely-used general-domain KGQA benchmarks: WebQuestionsSP (WebQSP) (Yih et al., 2016) and ComplexWebQuestions (CWQ) (Talmor and Berant, 2018). Both WebQSP and CWQ use Freebase (Bollacker et al., 2008) as the underlying KG. The details of three datasets are described in Appendix A. Following prior work, we use Hits@1 and F1 as evaluation metrics. Hits@1 reflects the proportion of questions with a correct top-ranked answer, while F1 balances precision and recall to assess both accuracy and coverage.

Baselines. We compare SKRAG with four major categories of baseline methods: (1) Embedding-based methods, including EmbedKGQA (Saxena et al., 2020) and TransferNet (Shi et al., 2021); (2) GNN-based methods, including NSM (He et al., 2021), SR+NSM (Zhang et al., 2022), ReaRev (Mavromatis and Karypis, 2022), and UniKGQA (Jiang et al., 2022); (3) LLM-based methods, including ChatGPT, ChatGPT+CoT, Qwen-Plus

Type	Methods	WebQSP		CWQ	
		Hits@1	F1	Hits@1	F1
Embedding	EmbedKGQA	66.6	-	45.9	-
	TransferNet	71.4	-	48.6	-
GNN	NSM	68.7	62.8	47.6	42.4
	SR+NSM	68.9	64.1	50.2	47.1
	ReaRev	76.4	70.9	52.9	47.8
	UniKGQA	77.2	72.2	51.2	49.0
LLM	ChatGPT	66.8	-	39.9	-
	ChatGPT+CoT	75.6	-	48.9	-
KG-based RAG	StructGPT	72.6	-	-	-
	ToG (ChatGPT)	76.2	-	58.9	-
	RoG(LLaMA2-Chat-7B)	85.7	70.8	62.6	56.2
	GNN-RAG	85.7	71.3	66.8	**59.4**
	SKRAG (Ours)	**90.1**	**75.3**	**67.2**	54.6

Table 3: Performance comparison of different methods on WebQSP and CWQ datasets. Bold represents the best result.

(Yang et al., 2024), and DeepSeek-V3 (Liu et al., 2024); and (4) KG-based RAG methods, including StructGPT (Jiang et al., 2023), ToG (Sun et al., 2023), RoG (Luo et al., 2023), GNN-RAG (Mavromatis and Karypis, 2024), GCR (Luo et al., 2024) and SubgraphRAG (Li et al., 2024).

Implementation Details. For SKRAG, we adopt the lightweight Qwen2.5-1.5B-Instruct[1] as the backbone for reasoning skeleton generation. The model is instruction fine-tuned for 3 epochs on WebQSP and CWQ, generating the top-10 reasoning skeletons. Given the relative sparsity of the domain-specific KG, we fine-tune the model for 2 epochs on SSUQA, generating the top-5 reasoning skeletons. For the answer generation phase, we employ DeepSeek-V3 as the general LLM. Additionally, we utilize the paraphrase-multilingual-MiniLM-L12-v2[2] as our Sentence-BERT model to measure semantic similarity. More detailed settings are provided in the Appendix B.

5.2 Results

Main Results. Tables 2 and 3 present the performance of our model SKRAG and various baselines on the SSUQA, WebQSP and CWQ datasets. The results demonstrate that SKRAG consistently achieves the best performance across most evaluation metrics on all three datasets. Among the KG-based RAG methods, SKRAG significantly outperforms prior methods, particularly on SSUQA and WebQSP. On SSUQA, when employing the same LLM combination, our method achieves superior results compared to RoG and GCR. This indicates that the QA performance of these approaches heavily relies on the parameter scale of the KG-specialized LLM, whereas our method attains superior performance even with smaller models. Similarly, our approach outperforms SubgraphRAG by 5.6% in Hits@1, further demonstrating its advantage on relatively sparse, domain-specific knowledge graphs. On WebQSP, it surpasses strong baselines like RoG and GNN-RAG by 5% in Hits@1 and 4% in F1, while being more resource-efficient. For CWQ, which relies on a denser and more structurally complex KG, the GNN mechanism tends to exhibit greater advantages; nevertheless, SKRAG still achieves comparable performance. In summary, SKRAG demonstrates strong performance across both specialized domain QA datasets with sparse KGs and general domain QA datasets with dense KGs, showcasing its versatility and effectiveness in diverse KGQA scenarios.

In contrast, traditional KGQA methods and general LLMs perform worse, due to limited reasoning

[1] https://huggingface.co/Qwen/Qwen2.5-1.5B-Instruct

[2] https://huggingface.co/sentence-transformers/paraphrase-multilingual-MiniLM-L12-v2

Method	SSUQA		WebQSP		CWQ	
	Hits@1	Recall	Hits@1	Recall	Hits@1	Recall
SKRAG	**97.3**	**97.2**	**90.1**	**84.2**	**67.2**	**64.9**
SKRAG *w/o* fine-tuned LLM	81.7	81.4	53.9	45.7	42.7	40.2
SKRAG *w/o* FSM constraint	94.4	94.3	78.4	74.7	53.9	51.3
SKRAG *w/o* reasoning	94.6	94.2	49.2	36.5	40.7	37.3

Table 4: Ablation studies of SKRAG.

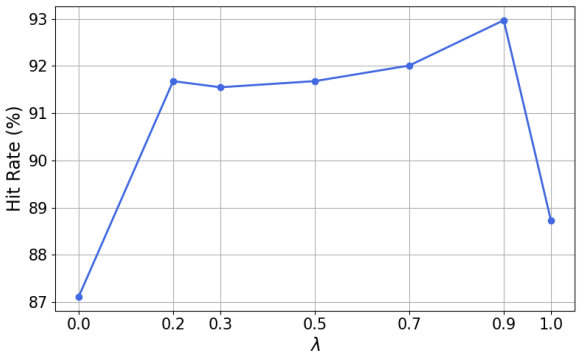

Figure 2: Skeleton Hit Rate under Different λ.

capabilities and insufficient structural understanding. This underscores the challenges LLMs face in complex KG reasoning and the need for enhanced integration with structured knowledge.

Ablation Study. We conduct ablation studies on three datasets to assess the contributions of SKRAG's core modules. We compare three variants: (1) *w/o* fine-tuned LLM, which uses an unfine-tuned LLM to generate reasoning skeletons; (2) *w/o* FSM constraint, which removes the FSM constraint mechanism; and (3) *w/o* reasoning, which replaces LLM reasoning with majority voting over top-5 frequent tail entities. The experimental results are presented in Table 4. From the results, it is evident that all three core modules of SKRAG are essential. Without fine-tuning the reasoning skeleton generator, the LLM lacks the understanding of the KG, which makes it difficult to accurately identify the relevant reasoning skeletons for the question. When the FSM constraint is removed, the LLM loses structural guidance, often generating reasoning skeletons that cannot be executed on the KG, resulting in subgraph retrieval failures. Finally, removing LLM reasoning reduces answer selection to tail entity frequency, ignoring semantic relevance and often misidentifying high-frequency entities while overlooking correct low-frequency ones.

Effect of Balancing Semantic Similarity and Structural Priors. To assess the effect of semantic similarity and structural priors in pruning reasoning skeletons, we analyze the parameter λ in Equation 5, which represents the proportion of their influence. Figure 2 presents the correct hit rate of reasoning skeletons generated under FSM constraints for WebQSP at different values of λ. Results show that at $\lambda = 0$ (using only prior skeleton scoring), the hit rate is 87.11%, indicating that structural priors alone cannot fully capture semantic relevance. The highest hit rate of 92.97% occurs at $\lambda = 0.9$. When $\lambda = 1$ (only semantic similarity), performance drops to 88.72%, suggesting ignoring structural priors harms coherence with the KG. Thus, semantic similarity and structural priors complement each other, and their combination effectively filters high-quality reasoning skeletons, allowing FSM to better constrain generation.

Effect of FSM Search Space Size. We analyze the impact of FSM search space size (top-k valid reasoning skeletons) on generation accuracy across three datasets, shown in Figure 3. For WebQSP, the hit rate peaks at 92.97% at $k = 50$ but declines beyond that, indicating that too many candidates in dense KGs introduce noise and reduce quality. CWQ shows an earlier drop, from 78.7% at $k = 25$ to 77.89% at $k = 50$, likely due to its complex structure and many potential skeletons. Conversely, SSUQA, with a sparse domain-specific KG, reaches its highest hit rate of 97.72% at $k = 75$, as increasing candidates doesn't add much noise and can boost performance. These results suggest tuning the number of candidate skeletons based on KG density and complexity to balance accuracy and search space.

6 Conclusion

In this paper, we propose SKRAG, a novel KG-based RAG framework tailored for KGQA in sce-

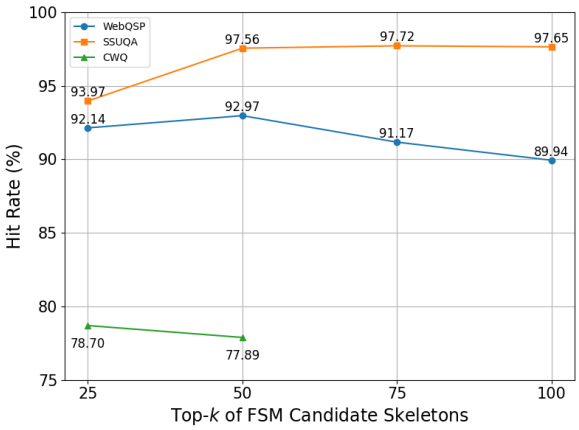

Figure 3: Effect of Top-k Skeleton Candidates on Hit Rate.

narios involving complex multi-fact questions over relatively sparse KGs, exemplified by but not limited to the domain of space science and utilization. SKRAG integrates a fine-tuned lightweight LLM with the FSM constraint to generate reasoning skeletons for efficient subgraph retrieval. We also introduce SSUQA, a benchmark dataset for the domain of space science and utilization. Experimental results demonstrate that SKRAG outperforms strong baselines on SSUQA and two general KGQA datasets, validating its adaptability and effectiveness across diverse KGQA scenarios.

Limitations

Despite the promising results, our current work still has several limitations:

- The proposed method has only been evaluated using a combination of Qwen2.5-1.5B-Instruct and DeepSeek-V3 as the underlying LLMs. To enhance the persuasiveness and generalizability of our method, future work will explore incorporating a broader range of LLMs for further validation.

- Aside from the newly constructed SSUQA dataset, our experiments have so far focused on only two general-domain datasets for KGQA. In the future, we plan to extend our evaluation to a wider variety of domain-specific KGQA datasets, to comprehensively assess the applicability and robustness of our method across diverse real-world scenarios.

Acknowledgments

This work was supported by the Science and Application Data Center of the Space Application System, China Manned Space Program (No. Y6140711WN), and the "Microgravity Multidisciplinary Database" of the National Basic Science Data Center (No. NBSDC-DB-17).

References

Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, and 1 others. 2023. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*.

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.

Sebastian Borgeaud, Arthur Mensch, Jordan Hoffmann, Trevor Cai, Eliza Rutherford, Katie Millican, George Bm Van Den Driessche, Jean-Baptiste Lespiau, Bogdan Damoc, Aidan Clark, and 1 others. 2022. Improving language models by retrieving from trillions of tokens. In *International conference on machine learning*, pages 2206–2240. PMLR.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, and 1 others. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Eduardo Faria Cabrera, Marcel Rodrigues de Barros, and Anna Helena Reali Costa. 2024. Improving llms' reasoning and planning with finite-state machines. In *Brazilian Conference on Intelligent Systems*, pages 110–124. Springer.

Yunfan Gao, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yixin Dai, Jiawei Sun, Haofen Wang, and Haofen Wang. 2023. Retrieval-augmented generation for large language models: A survey. *arXiv preprint arXiv:2312.10997*, 2:1.

John A Hartigan and Manchek A Wong. 1979. Algorithm as 136: A k-means clustering algorithm. *Journal of the royal statistical society. series c (applied statistics)*, 28(1):100–108.

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

Xiaoxin He, Yijun Tian, Yifei Sun, Nitesh Chawla, Thomas Laurent, Yann LeCun, Xavier Bresson, and Bryan Hooi. 2024. G-retriever: Retrieval-augmented generation for textual graph understanding and question answering. *Advances in Neural Information Processing Systems*, 37:132876–132907.

Yuntong Hu, Zhihan Lei, Zheng Zhang, Bo Pan, Chen Ling, and Liang Zhao. 2024. Grag: Graph retrieval-augmented generation. *arXiv preprint arXiv:2405.16506*.

Jie Huang and Kevin Chen-Chuan Chang. 2023. Towards reasoning in large language models: A survey. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 1049–1065.

Lei Huang, Weijiang Yu, Weitao Ma, Weihong Zhong, Zhangyin Feng, Haotian Wang, Qianglong Chen, Weihua Peng, Xiaocheng Feng, Bing Qin, and 1 others. 2025. A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions. *ACM Transactions on Information Systems*, 43(2):1–55.

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.

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*.

Jungo Kasai, Keisuke Sakaguchi, Ronan Le Bras, Akari Asai, Xinyan Yu, Dragomir Radev, Noah A Smith, Yejin Choi, Kentaro Inui, and 1 others. 2023. Realtime qa: What's the answer right now? *Advances in neural information processing systems*, 36:49025–49043.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large language models are zero-shot reasoners. *Advances in neural information processing systems*, 35:22199–22213.

Mufei Li, Siqi Miao, and Pan Li. 2024. Simple is effective: The roles of graphs and large language models in knowledge-graph-based retrieval-augmented generation. *arXiv preprint arXiv:2410.20724*.

Lei Liang, Mengshu Sun, Zhengke Gui, Zhongshu Zhu, Zhouyu Jiang, Ling Zhong, Yuan Qu, Peilong Zhao, Zhongpu Bo, Jin Yang, and 1 others. 2024. Kag: Boosting llms in professional domains via knowledge augmented generation. *arXiv preprint arXiv:2409.13731*.

Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, and 1 others. 2024. Deepseek-v3 technical report. *arXiv preprint arXiv:2412.19437*.

Yunfei Liu, Shengyang Li, Chen Wang, Xiong Xiong, Yifeng Zheng, Linjie Wang, and Shiyi Hao. 2023. Ssuie 1.0: A dataset for chinese space science and utilization information extraction. In *CCF International Conference on Natural Language Processing and Chinese Computing*, pages 223–235. Springer.

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*.

Linhao Luo, Zicheng Zhao, Chen Gong, Gholamreza Haffari, and Shirui Pan. 2024. Graph-constrained reasoning: Faithful reasoning on knowledge graphs with large language models. *arXiv preprint arXiv:2410.13080*.

Costas Mavromatis and George Karypis. 2022. Rearev: Adaptive reasoning for question answering over knowledge graphs. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 2447–2458.

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. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1400–1409.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, and 1 others. 2022. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744.

Shirui Pan, Linhao Luo, Yufei Wang, Chen Chen, Jiapu Wang, and Xindong Wu. 2024. Unifying large language models and knowledge graphs: A roadmap. *IEEE Transactions on Knowledge and Data Engineering*, 36(7):3580–3599.

Boci Peng, Yun Zhu, Yongchao Liu, Xiaohe Bo, Haizhou Shi, Chuntao Hong, Yan Zhang, and Siliang Tang. 2024a. Graph retrieval-augmented generation: A survey. *arXiv preprint arXiv:2408.08921*.

Yixing Peng, Quan Wang, Licheng Zhang, Yi Liu, and Zhendong Mao. 2024b. Chain-of-question: A progressive question decomposition approach for complex knowledge base question answering. In *ACL (Findings)*.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 3982–3992.

Stuart J Russell and Peter Norvig. 2016. *Artificial intelligence: a modern approach*. pearson.

Diego Sanmartin. 2024. Kg-rag: Bridging the gap between knowledge and creativity. *arXiv preprint arXiv:2405.12035*.

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

Jiaxin Shi, Shulin Cao, Lei Hou, Juanzi Li, and Hanwang Zhang. 2021. Transfernet: An effective and transparent framework for multi-hop question answering over relation graph. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 4149–4158.

Kurt Shuster, Spencer Poff, Moya Chen, Douwe Kiela, and Jason Weston. 2021. Retrieval augmentation reduces hallucination in conversation. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 3784–3803.

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. 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.

Karthik Valmeekam, Matthew Marquez, Sarath Sreedharan, and Subbarao Kambhampati. 2023. On the planning abilities of large language models-a critical investigation. *Advances in Neural Information Processing Systems*, 36:75993–76005.

Junjie Wang, Mingyang Chen, Binbin Hu, Dan Yang, Ziqi Liu, Yue Shen, Peng Wei, Zhiqiang Zhang, Jinjie Gu, Jun Zhou, and 1 others. 2024. Learning to plan for retrieval-augmented large language models from knowledge graphs. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pages 7813–7835.

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*.

Jason Wei, Maarten Bosma, Vincent Y Zhao, Kelvin Guu, Adams Wei Yu, Brian Lester, Nan Du, Andrew M Dai, and Quoc V Le. 2021. Finetuned language models are zero-shot learners. *arXiv preprint arXiv:2109.01652*.

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. *Advances in neural information processing systems*, 35:24824–24837.

Yike Wu, Nan Hu, Sheng Bi, Guilin Qi, Jie Ren, Anhuan Xie, and Wei Song. 2023. Retrieve-rewrite-answer: A kg-to-text enhanced llms framework for knowledge graph question answering. *arXiv preprint arXiv:2309.11206*.

Derong Xu, Xinhang Li, Ziheng Zhang, Zhenxi Lin, Zhihong Zhu, Zhi Zheng, Xian Wu, Xiangyu Zhao, Tong Xu, and Enhong Chen. 2025. Harnessing large language models for knowledge graph question answering via adaptive multi-aspect retrieval-augmentation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 39, pages 25570–25578.

An Yang, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chengyuan Li, Dayiheng Liu, Fei Huang, Haoran Wei, and 1 others. 2024. Qwen2. 5 technical report. *arXiv preprint arXiv:2412.15115*.

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.

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. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 5773–5784.

A Datasets

WebQSP and CWQ are both based on the Freebase KG, which contains approximately 164.6 million facts and 24.9 million entities, making it a highly dense graph. Questions in WebQSP typically require up to 2-hop reasoning. Furthermore, CWQ is an extension of WebQSP, where questions are made more complex by expanding the question entities or adding additional constraints, involving more complex queries with up to 4-hop reasoning. In comparison, the domain-specific KG used in SSUQA consists of 30,338 triples, making it relatively sparse. Detailed dataset statistics are provided in Table 5.

To reduce the scale of the KG for computational efficiency, we follow previous work (He et al., 2021) and construct a subgraph for each question in the datasets by extracting all triples within the maximum reasoning hops from the question entities.

Datasets	Train	Dev	Test	Max hop
SSUQA	11,600	3,200	3,200	3
WebQSP	2,826	-	1,628	2
CWQ	27,639	3,519	3,531	4

Table 5: Statistics of datasets.

B Implementation Details

During the fine-tuning of the skeleton generator, the batch size is set to 2 and the learning rate is set to 2e-5. The training is conducted on four NVIDIA RTX 3090 GPUs, each with 24 GB of memory.

When comparing with LLM-based methods, we adopt zero-shot ChatGPT and ChatGPT+CoT for WebQSP and CWQ. For SSUQA, we directly input the constructed subgraphs into Qwen-Plus and DeepSeek-V3 for answer generation without prior retrieval. For a fair comparison on the SSUQA dataset, we reproduced RoG and GCR using the same LLM combination (Qwen2.5-1.5B-Instruct + DeepSeek-V3) as used in our method.

During the construction of SSUQA, we manually design 2-3 CoT demonstrations and utilize DeepSeek-V3 to generate the corresponding natural language questions, guiding the LLM more effectively.

C Prompt Templates

Figures 4, 5, and 6 respectively show the prompt templates for skeleton generation, question answering, and question generation.

Prompt Template for Skeleton Generation

The reasoning skeleton should consist of the question entities and their respective relation chains that are relevant to deriving the answer. Starting from the question entities, generate an appropriate reasoning skeleton in the knowledge graph to answer the question:
<Question>

Figure 4: The prompt template for skeleton generation.

Prompt Template for Question Answering

Based on the reasoning paths retrieved from the knowledge graph, please answer the question. Please return formatted answers as a list, each prefixed with "ans:".

Reasoning Paths:
<Reasoning Paths>

Question:
<Question>

Figure 5: The prompt template for question answering.

Prompt Template for Question Generation

Please generate relevant questions based on the given set of triples. First, identify the entity types in each triple. Then, convert each triple into a complete natural language statement based on the identified entity types. Next, incrementally generate question phrases from the statement, and finally formulate the final question by querying the tail entity of the triple. Please follow the provided example and complete the question generation task in the same format.

Demonstration:
Input:
(Fluid mechanics panoramic camera image, Experimental payload, Fluid physics science experiment cabinet), (Fluid physics science experiment cabinet, Space science and utilization research field, Microgravity physical science)
Output:
Entity type: Fluid mechanics panoramic camera image - Experimental data, Fluid physics science experiment cabinet - Experimental payload, Microgravity physical science - Space science and utilization research field.
Triple to text: The experimental payload related to the fluid mechanics panoramic camera image is the fluid physics science experiment cabinet. The space science and utilization research field to which the fluid physics science experiment cabinet belongs is microgravity physical science.
Sub-question phrase 1: The experimental payload related to the fluid mechanics panoramic camera image
Sub-question phrase 2: The space science and utilization research field to which the experimental payload related to the fluid mechanics panoramic camera image belongs
Question: Which space science and utilization research field does the experimental payload related to the fluid mechanics panoramic camera image belong to?
Answer: Microgravity physical science

Input:
<A set of triples>

Figure 6: The prompt template for question generation.