Bridging the Gap: Integrating Knowledge Graphs into Large Language Models for Complex Question Answering

Anonymous ACL submission

Abstract

Large language models (LLMs) have performed impressively in various natural language processing tasks. However, their inherent hallucination phenomena seriously challenge their credibility in complex reasoning. Combining explainable knowledge graphs (KGs) with LLMs is a promising path to address this challenge. However, there is a huge representation gap between structured KGs and LLMs pre-trained from unstructured text, and how to make LLMs understand and utilize KGs for complex reasoning is a challenging topic. To tackle this challenge, we propose a comprehensive method: improving retrieval capabilities for KG by integrating reasoning processes and subgraph information and enhancing LLMs' understanding and utilization of KG through an efficient yet effective KG representation and KG-related tuning. Extensive experiments on two KGQA datasets and various LLMs demonstrate that our method outperforms existing strong KGQA methods[1].

1 Introduction

Recently, the emergence and application of large language models (LLMs) (OpenAI, 2022, 2023; Bubeck et al., 2023; Yang et al., 2023) have attracted widespread attention from researchers and the general public. It demonstrates remarkable reasoning capabilities, managing to solve complex reasoning problems through step-by-step thinking and planning (Wei et al., 2022; Khot et al., 2023). However, the reasoning of LLMs is not invariably reliable and may conflict with factual reality, a phenomenon known as hallucination (Wang et al., 2023; Huang et al., 2023). This will limit the application of LLMs in areas requiring high reliability, such as healthcare and science.

The knowledge graph (KG) stores high-quality common sense or domain-specific knowledge in structured triplets. Due to its reliability and interpretability, it is considered a promising method to improve the reliability of LLM reasoning (Pan et al., 2024). Therefore, researchers have never ceased their attempts to integrate KGs with language models (Zhang et al., 2019; Liu et al., 2020; Lewis et al., 2020; Sun et al., 2021). Among them, the knowledge graph question answering (KGQA) is the critical task to incorporate the knowledge of KG into reasoning models (Lan et al., 2021; Miller et al., 2016; Sun et al., 2018; Jiang et al., 2023b).

KGQA faces two main challenges: (1) How to retrieve specific knowledge from KGs to help reasoning precisely; (2) How to make the reasoning model understand and utilize the structured knowledge in KGs. For the first challenge, existing solutions include direct retrieval (Sun et al., 2019; Baek et al., 2023; Jiang et al., 2023b) and semantic parsing (Sun et al., 2020; Lan and Jiang, 2020; Gu and Su, 2022; Ye et al., 2022; Yu et al., 2023). Direct retrieval involves taking the question as a query and the knowledge triplets in the KG as candidates, using either sparse or dense retrieval techniques to identify several candidates most relevant to the query. Semantic parsing transforms the question into an executable structured query statement (e.g., SPARQL) and executes the query in KGs. However, individual knowledge in KGs has limited semantics, and direct retrieval makes it difficult to model the semantic relevance, especially in multi-hop question answering, where knowledge that is semantically weakly relevant to the question may instead be important intermediate knowledge. Semantic parsing faces the problem of non-executable or incorrectly executed generated queries (Yu et al., 2023). For the latter challenge, since current LLMs are primarily trained in unstructured text, they may not effectively comprehend and utilize knowledge in the structured form. Consequently, existing methods often convert KG content to natural language (He et al., 2024; Ye

[1]All the code, data and model checkpoints will be publicly available at https://anonymous.com

et al., 2024) or linearized triplets (Luo et al., 2024). However, natural language renders KG knowledge redundant, necessitating more tokens representing the KG, while linearization undermines the structural information inherent within the KG.

To address these two challenges, this paper introduces a novel retrieval-augmented method. Our proposed retrieval model combines chain-of-thought (CoT) (Wei et al., 2022) and subgraphs, where subgraphs enrich the semantic information of candidate knowledge, and CoT offers intermediate reasoning steps involved in multi-hop question answering, aiding the retrieval model in recalling useful intermediary knowledge. We then represent the KG in YAML format to reduce input redundancy and enhance the LLM's understanding of KGs by instruction tuning across three KG-level tasks and KG data pre-training. To further strengthen the reasoning capabilities of LLMs utilizing KGs, we generate explicit reasoning process data with larger open-source LLMs and train our reasoning models with these synthetic datasets. To evaluate the effectiveness of our proposed KGQA method, we conduct experiments on LLaMA2-7b-Chat on two KGQA datasets. Experimental results demonstrate our proposed method can perform better than existing strong baselines. Further analysis indicates the generalizability to other LLMs.

Overall, our main contributions include:

- We integrate the reasoning process and subgraph into knowledge retrieval, which aids in recalling useful intermediate knowledge for reasoning.
- We propose a novel and efficient KG representation method, the YAML format, which reduces token redundancy by approximately 25% compared to the traditional triple format. Combined with our proposed KG-related tuning, LLM is able to understand and utilize YAML-format KG to accomplish complex reasoning tasks.
- Extensive experiments show that our method outperforms the existing strong baselines in two challenging datasets.

2 Related Work

Knowledge graph question answering (KGQA) enables models to answer questions by integrating common sense or domain-specific knowledge from knowledge graphs. Current approaches to KGQA can be categorized into three types: embedding-based, semantic parsing-based and retrieval-augmented. Embedding-based methods project entities and relations from knowledge graphs into an embedding space, and utilize key-value memory networks (Miller et al., 2016), sequence modeling (He et al., 2021), or graph neural networks (Yasunaga et al., 2021) to learn the reasoning process between questions and the entities and relations. Semantic parsing-based methods utilize the semantic parsing model to convert questions into structured query language oriented towards the knowledge base (e.g. SPARQL), and then execute it to search answers from the knowledge graph (Sun et al., 2020; Lan and Jiang, 2020; Gu and Su, 2022; Ye et al., 2022; Yu et al., 2023). However, semantic parsing-based methods rely on retrieving answers from knowledge bases, overlooking the reasoning capabilities of models. Retrieval-augmented methods combine knowledge graphs with the intrinsic reasoning capabilities of models. They first retrieve question-relevant knowledge triples or subgraphs from the knowledge graphs, and then leverage this retrieved knowledge to enhance the factualness of the reasoning. Sun et al. (2018) propose the GraftNet which utilizes entity linking to retrieve subgraphs. Subsequently, many works adopt effective dense retrieval models as their retrieval modules, such as PullNet (Sun et al., 2019), SR (Zhang et al., 2022), DiFar (Baek et al., 2023), UniKGQA (Jiang et al., 2023b), etc. Today, natural language processing has entered the era of large language models, where retrieval-augmented generation (RAG) enables these models to effectively leverage external knowledge to accomplish various tasks (Lewis et al., 2020; Gao et al., 2024). Wang et al. (2023) retrieve knowledge from knowledge graphs to verify and correct the factual within chain-of-thought, resulting in the generation of more precision responses. Yu et al. (2023) utilize a larger-scale retriever to enhance retrieval performance and generate both semantic parsing expressions and inference results in the generation phase, compensating for their respective shortcomings by integrating the two approaches.

3 Methodology

In this section, we present our proposed KGQA method, which is based on the retrieval-augmentation generation paradigm. First, we introduce the overall inference process of our method, including the KG retrieval module and the KG reasoning module. Then, we detail the training processes for the two modules.

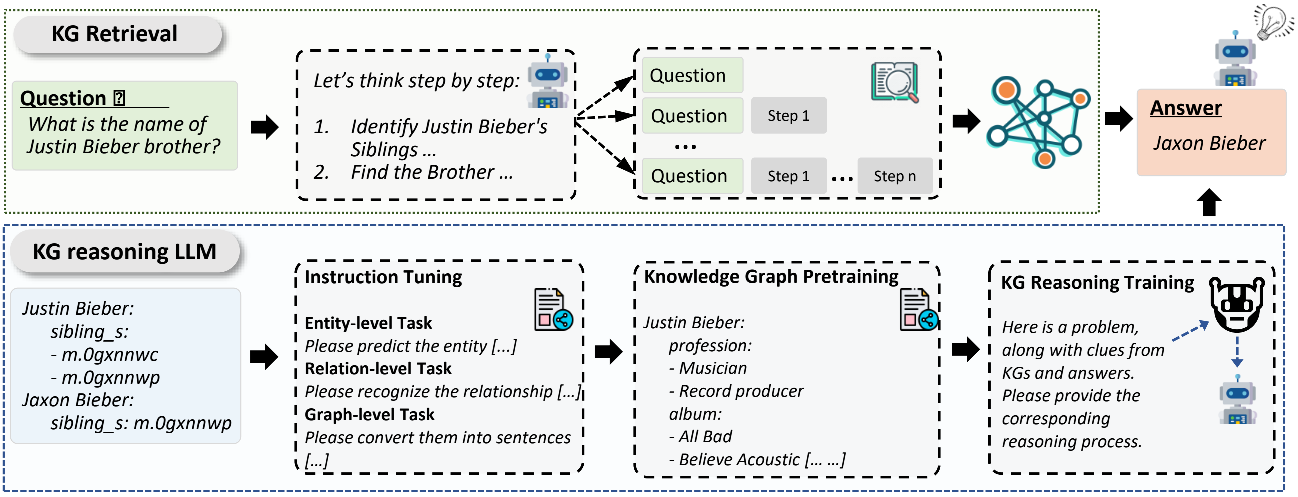

Figure 1: llustration of our KGQA method. It contains two modules, Knowledge Graph Retrieval Model and Knowledge Graph Reasoning LLM.

Prompt 1: Generating CoT for Retrieval

Please think step by step and then answer the given question.

Here are some examples:
Input: <Demonstration Question>
CoT: Let's think step by step. <Demonstration CoT>
Output: <Demonstration Answer>

Input: <Question>
CoT: Let's think step by step.

3.1 Overview

As Fig. 1 shows, our KGQA method includes two modules: KG retrieval model and KG reasoning LLM. Given a question q and a knowledge graph $\mathcal{G} = \{t_i\}_i^n$, where $t_i = (e_h^i, r^i, e_t^i) \in \mathcal{E} \times \mathcal{R} \times \mathcal{E}$ is a knowledge triple; $\mathcal{E}, \mathcal{R}$ are the set of entities and relationships; e_h, r, e_t are the head entity, relationship and tail entity, respectively. After we complete training the KG retrieval model R_ϕ and the KG reasoning LLM $\mathcal{M}_\theta$, in the inference stage, the LLM $\mathcal{M}_\theta$ first plans the problem and generates a reasoning process with chain-of-thought (CoT) prompting:

$$\{c^1, ..., c^j\} = \mathcal{M}_\theta(p_{cot} \oplus q), \qquad (1)$$

where c^j is the j-th step reasoning process and p_{cot} is the CoT prompting as shown in Prompt 1, $\oplus$ means the concatenation operator. Then, we progressively concatenate the reasoning process with the question as queries to retrieve knowledge: $q^j = q \oplus c^1 \oplus ... \oplus c^j$ $(q^0 = q)$. For each candidate knowledge t, we integrate the surrounding subgraph information $\mathcal{G}_t = \{(e_h, r, e_t) | e_h = e_h^t \vee e_t = e_t^t\}$. The retrieval can be formalized as follows:

$$\mathcal{T} = \text{Top}_k \sum_j f(R_\phi(q^j), R_\phi(t \oplus \mathcal{G}_t)), \qquad (2)$$

where f is the similarity function between the query representation and the candidate representation (e.g. cosine similarity or dot-product similarity), $\mathcal{T}$ is the set of top-k candidates retrieved that are most relevant to the query.

Prompt 2: Utilizing KG to Reason

Please think step by step and then answer the given question. Please keep the answer as simple as possible and return all the possible answers as a list. If there are hints, please combine this information to answer.

Here are some examples:
Input: <Demonstration Question>
Hints: <Demonstration Knowledge Graph>
CoT: Let's think step by step. <Demonstration CoT>
Output: <Demonstration Answer>

Input: <Question>
Hints: <Knowledge Graph>
CoT: Let's think step by step.

After retrieval, the candidate set is transformed into YAML format and serves as part of the input for the KG reasoning LLM, which reasons and outputs the final answer through Prompt 2.

3.2 Knowledge Retrieval with Chain-of-thoughts and Subgraphs

Retrieving relevant and useful knowledge from knowledge graphs is critical for the performance of KGQA. Benefiting from the increasingly advanced dense retrieval, we can obtain relevant knowledge through direct retrieval, without the need for elaborate techniques such as semantic parsing and entity linking (Baek et al., 2023). However, the semantic expression of individual knowledge in knowledge graphs is limited, and the semantic relationship between knowledge and questions is not directly related in multi-hop question answering. Therefore, we consider incorporating neighboring knowledge information and reasoning processes when retrieving knowledge.

We employ the contrastive learning to train our retrieval model, the training loss is:

$$\mathcal{L} = -\log \frac{\exp(f(R_\phi(q^j), R_\phi(t^+ \oplus \mathcal{G}_{t^+})))}{\sum_{t \in \tau} \exp(f(R_\phi(q^j), R_\phi(t \oplus \mathcal{G}_t)))}, \tag{3}$$

where τ contains all triplets in the same batch, t^+ is the positive sample and others are negative samples. In our method, we take all the knowledge triples on the path from the entity in the question to the answer entity in the knowledge graph as positive samples, and randomly sample from the remaining triples as negative samples.

Different from the inference stage, We only use the LLaMA-7b-Chat model, which has not been specifically trained for knowledge graph tasks, to generate the reasoning process for training (This method allows for the complete decoupling of the training of the retrieval and reasoning models, enabling them to be trained independently and in parallel). To address the inconsistency in CoT quality during training and inference, we employ rationalization prompting (Prompt 3 [2]) during training, providing the answer in the prompt so that the LLM can generate a reasonable reasoning process based on the answer.

Prompt 3: Generating CoT for Training

Here is a problem, along with (clues from a knowledge graph and) the answer. Please provide the corresponding reasoning process.

Here are some examples:
Input: <Demonstration Question>
(**Clues:** <Demonstration Knowledge Graph>)
Answer: <Demonstration Answer>
Output: <Demonstration CoT>

Input: <Question>
(**Clues:** <Knowledge Triples>)
Answer: <Answer>
Output:

[2]Prompt 3 applies to both retrieval training and reasoning training, and KG information is only provided during reasoning training (in section 3.4).

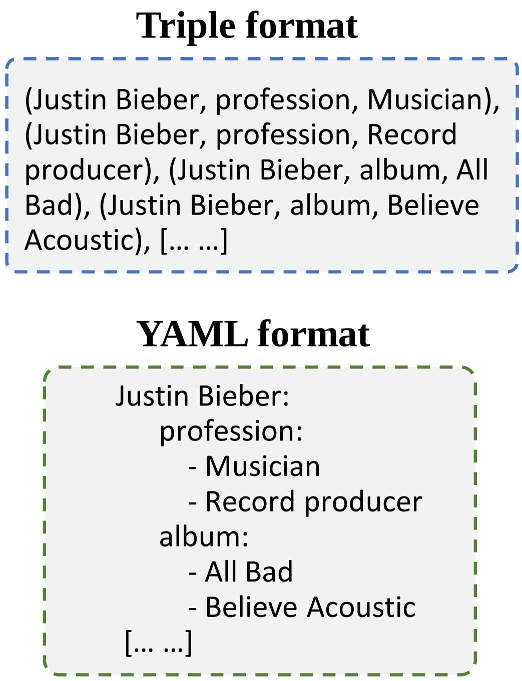

Figure 2: An example of triple and YAML format KG.

3.3 Utilizing Knowledge Graphs Effectively and Efficiently in LLMs

Knowledge graphs are essentially structured knowledge, while LLMs are typically pretrained on unstructured text. To bridge this gap and enable LLMs to better understand and utilize the structured knowledge, we propose a simplified representation for knowledge graphs. Additionally, we employ instruction tuning and continual pre-training to ensure that LLMs internalize both the knowledge and this representation form.

YAML Format KG In general, the retrieved knowledge triples may exhibit many literal similarities, such as having the same head entity or relation across multiple triples. If we linearize these triples directly as input for the reasoning LLM, it will result in significant token redundancy, thereby impacting the efficiency of the model's inference. Therefore, we try to represent the knowledge graph in a more efficient format. Our approach uses the YAML format, a data serialization language with a simple syntax. As shown in Figure 2, YAML uses indentation to represent hierarchical relationships. We treat different head entities as the first-level relationship, different relationships under the same head entity as the second level, and different tail entities under the same head entity and relationship as the final level.

KG Instruction For general-purpose LLMs, representing knowledge graphs in YAML format is unfamiliar and infrequently encountered in their pre-training corpora. Therefore, to enable LLMs to understand knowledge graphs in YAML, we design three types of graph-related instruction-tuning tasks: (1) **entity-level tasks**, where the LLM is required to reason the entity according to neighbors; (2) **relationship-level tasks**, where the task is to reason the relationship between entities; (3) **graph-level tasks**, where the LLM needs to understand the semantic of knowledge graphs and converts to natural language. We design three different instructions for each type of task (shown in Table 1) and denote the instruction prompt as $\mathcal{I}$. For entity-level and relationship-level instruction tasks, we automatically construct them based on the data in the knowledge graph without the need for additional manual annotation. For graph-level instruction tasks, we utilize existing high-quality KG-to-text datasets (Gardent et al., 2017). The training loss of KG instruction is:

$$\mathcal{L}_{instruct} = -\sum_{l}^{L} y^l logp(\hat{y}^l|\mathcal{I}(x), y^{<l}), \quad (4)$$

where (x, y) is the input-output pair, L is the length of y, y^l is y's l-th token, $y^{<l}$ means tokens before l-th token, $\hat{y}^l$ is the predicted l-th token.

Continual KG Pre-training To further learn the structured knowledge embedded in knowledge graphs, we propose the continual KG pre-training method. We serialize the entire knowledge graph in YAML format and train it by the next token prediction:

$$\mathcal{L}_{pretrain} = -\sum_{l}^{L} x^l logp(\hat{x}^l|x^{<l}), \quad (5)$$

where x is the pretraining data.

3.4 KG-based Reasoning Training

In Section 3.3, we enhance the LLM's understanding of the specialized structured representation of KG, without explicitly teaching the LLM to use KG for reasoning. In practical scenarios, we need to address two issues: (1) How to utilize KG for multi-hop reasoning; (2) How to manage the retrieved noisy knowledge that lacks crucial task-related information or contains irrelevant redundant information. To address these two issues, we use a retrieval model that has not been fine-tuned for KGQA tasks to retrieve noisy knowledge, and a more powerful LLM to generate high-quality reasoning processes for questions based on retrieved knowledge and answers with Prompt 3. After obtain the knowledge and reasoning processes, we train our reasoning LLM with the loss function defined in Equation 4.

4 Experiments

4.1 Baseline Methods

We compare our method with the following competitive KBQA baselines.

NSM (He et al., 2021) proposes a teacher-student framework where the teacher model learns supervision signals for intermediate reasoning processes through forward and backward reasoning, which are then conveyed to the student model for multi-hop inference.

Transfernet (Shi et al., 2021) utilizes the graph attention mechanism to capture the relevance among questions, entities, and relationships, guiding a step-by-step traversal on the knowledge graph towards the answer.

SR+NSM (+E2E) (Zhang et al., 2022) proposes a effective subgraph retriever to retrieve the most relevant relation-path for reasoning and then utilizes the NSM to reason. **E2E** denotes further jointly finetuning the SR+NSM.

QGG (Lan and Jiang, 2020) is a semantic parsing based approach that incorporates constraints and extends relational paths in the process of generating query graphs.

UniKGQA (Jiang et al., 2023b) unifies the retriever and reasoning module into a single model.

DECAF (Yu et al., 2023) proposes a method for joint generating semantic parsing forms and direct answers, significantly improving the executability of semantic parsing forms.

StructGPT (Jiang et al., 2023a) utilizes LLMs' tool-using capabilities to interactive between LLMs and knowledge bases, which facilitates multi-hop reasoning through iterative interactions.

KD-CoT (Wang et al., 2023) retrieves relevant knowledge from the KG during the reasoning process, progressively verifying and correcting facts in the reasoning process.

RoG (Luo et al., 2024) RoG leverages the powerful generative and planning capabilities of LLMs to generate reasoning paths. It retrieves corresponding knowledge from knowledge graphs based on these paths and synthesizes various reasoning paths to deduce the final answer. RoG is based on LLaMA2-7b-chat.

Task	Instruction
Entity	Please predict the entity represented by <mask> based on the one-hop relationships in the knowledge graph. \n Input: {Input} Based on the one-hop relationships in the knowledge graph, infer the entity represented by <mask>. \n Input: {Input} Make a prediction about the masked entity, using the one-hop relationships in the knowledge graph as a reference. \n Input: {Input}
Relationship	Please recognize the relationship between the two entities.\n Knowledge Graph: {KG} \n Input: {Input} Please predict the relationship between the two entities.There are some one-hop information of these entities: {KG} \n Input: {Input} Make a prediction about the relationship, using the one-hop relationships in the knowledge graph as a reference. \n {KG} \n Input: {Input}
Graph2text	Please deeply understand the following knowledge graph, and then convert them into a coherent sentence. \n Input: {Input} Given these knowledge graph, please deeply write a paragraph that integrates the information contained in them. \n Input: {Input} Compose an informative report using the information from these knowledge graph. \n Input: {Input}
Text2graph	Please extract all entities and relationships in the sentence. \n Input: {Input} Given the sentence, please extract a knowledge graph that integrates the information contained in them. \n Input: {Input} Please deeply understand the following sentence, and then generate a knowledge graph. \n Input: {Input}

Table 1: Instructions of knowledge graph related tasks.

Dataset	WebQSP	CWQ
#Train	2,848	27,639
#Valid	250	3,519
#Test	1,639	3,519
#Max hop	2	5

Table 2: Characteristics of datasets

4.2 Datasets and Evaluation Metrics

To evaluate the effectiveness of our proposed KGQA method, we conduct experiments on two popular and challenging dataset: *WebQuestionsSP* (WebQSP) (Yih et al., 2015) and *Complex WebQuestions 1.1* (CWQ) (Talmor and Berant, 2018). Both two datasets are created from the Freebase knowledge graph (Bollacker et al., 2008). We report more details in Table 2.

Following previous work (Jiang et al., 2023b), we take the Hits@1 and F1 as evaluation metrics for WebQSP and CWQ. Hits@1 is a metric for measuring the accuracy of the top-1 answer. For generative LLMs, we consider the first answer generated as the top-1 answer. Given a question may have multiple answers, F1 balances precision and recall of the predicted answers, and is used to assess the overall coverage of the model's predictions.

4.3 Experimental Details

In our main experiments, we take LLaMA2-7b-Chat [3] as the reasoning backbone model and BGE-1.5-en-base [4] as the retrieval backbone model. We finetune the retrieval model on the training set of WebQSP and CWQ for 5 epochs. The learning rate is set to 1e-5 and the batch size is set to 64. We search for a path in Freebase that starts with a question entity and ends with an answer entity (limiting the length of the path to no more than 5), treating all entities in the path as positive samples of the query, and randomly sampling 6 triples as negative samples. We construct 270k entity-level and 540k relationship-level instruction data from Freebase, and the WebNLG dataset (Gardent et al., 2017) as graph-level instruction data. We tune the reasoning model for 2 epochs with the learning rate set to 2e-6 and batch size set to 64. Then, we perform continual pre-training on the Freebase data using the same setting. For KG-based reasoning training, we use the WebQSP and CWQ training sets as queries to retrieve knowledge from KG using BGE-1.5-en-base. Then, we employ Llama2-70b-Chat [5] to generate high-quality reasoning processes, which are subsequently used to train our reasoning model. The training is conducted for 5 epochs with the learning rate set to 2e-6 and batch size set to 64. In the inference stage, the first 3 samples from the WebQSP training set are added as demonstrations before each question. For each question, we use our retriever to retrieve the top-20 triples most relevant to it. For generation, we adopt top-p sampling with the temperature set to 0.85 and p set to 0.9, and the generation length is 512 tokens. To enhance inference speed, model inference is based on the vLLM library (Kwon et al., 2023).

4.4 Main Results

Table 3 shows the results of our KGQA model and other baselines on WebQSP and CWQ. Firstly, general-purpose LLMs do not perform well on KGQA tasks, with neither LLaMA2-7b-Chat nor the ChatGPT able to match the performance of KGQA-specific models, especially in the more challenging CWQ dataset. This means that LLMs still have significant room for improvement in their ability to understand and utilize structured knowl-

[3] https://huggingface.co/meta-llama/Llama-2-7b-chat-hf
[4] https://huggingface.co/BAAI/bge-base-en-v1.5
[5] https://huggingface.co/meta-llama/Llama-2-70b-chat-hf

Models	WebQSP		CWQ	
	Hits@1	**F1**	**Hits@1**	**F1**
NSM	68.7	62.8	47.6	42.4
TransferNet	71.4	-	48.6	-
SR+NSM	68.9	64.1	50.2	47.1
SR+NSM+E2E	69.5	64.1	49.3	46.3
QGG	73.0	73.8	36.9	37.4
UniKGQA	77.2	72.2	51.2	49.0
DECAF	82.1	**78.8**	-	-
LLaMA2-7b-chat	59.5	34.0	34.0	22.7
StructGPT	69.6	-	-	-
ChatGPT	75.6	-	48.9	-
KD-CoT	68.6	52.5	55.7	-
RoG	85.7	70.8	62.6	**56.2**
Ours	**91.5**	74.0	**68.7**	55.6

Table 3: Experimental results of our KGQA method and strong baselines on the two dataset. **Bold** and underline denote the best and the second best result, respectively.

Models	Hits@1	Precision	Recall	F1
Ours	68.7	**56.4**	63.0	**55.6**
- w/o SubKG-R	65.4	52.9	59.7	52.2
- w/o CoT-R	66.1	52.8	60.3	52.5
- w/o KG-IT	68.0	55.8	62.3	55.1
- w/o KG-PT	**69.4**	53.8	**63.9**	54.1
- w/o KG-RT	42.6	34.0	37.0	32.3

Table 4: Ablation results on CWQ. **R** denotes retrieval, **IT** denotes instruction tuning, **PT** denotes continual pretraining and **RT** denotes reasoning training.

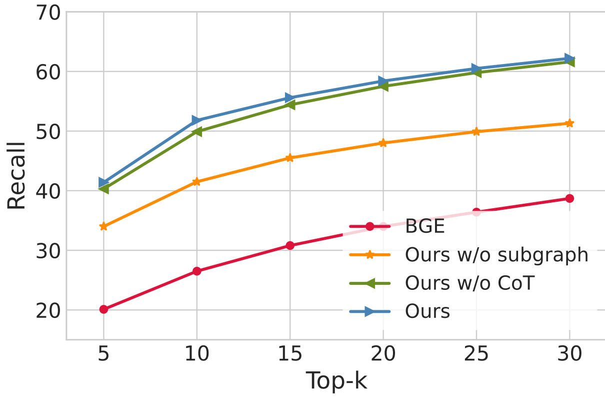

Figure 3: Comparison of recall ability of different retrieval models.

edge graphs for complex reasoning. Our approach improves Hits@1 by 15-20% on the two KGQA tasks compared to these strong general-purpose LLMs. Currently, the state-of-the-art (SOTA) models for KGQA are RoG and DECAF, which are based on retrieval-augmentation and semantic parsing respectively, with backbone models that have over a billion parameters. In terms of the Hits@1 metric, our method comprehensively surpasses the existing SOTA, especially in the WebQSP dataset, where we achieve a breakthrough of more than 90% for the first time. Compared to RoG, our method shows a significant improvement in 6% Hits@1 on both WebQSP and CWQ. Overall, our method is comparable to the SOTA models in terms of the F1 score. On WebQSP, it falls short of DECAF but outperforms RoG by 3%, and on CWQ, it is on par with RoG.

5 Analysis and Discussion

5.1 Ablation Study

We conduct ablation experiments on CWQ to analyze the contributions of KG retrieval module and KG reasoning module. As shown in the experimental results in Table 4, each module in our method is indispensable. The most crucial component is KG reasoning training; without it, the model's performance plummets from 68.7% to 42.6% in Hits@1. This indicates that even if LLMs encode KG information and understand its semantics, it is in vain if LLMs fail to utilize KG for reasoning. The second key component is the retrieval module. Experiments show that the roles of subgraph information and the reasoning process are complementary, and their combined use maximizes effectiveness. Lacking either can lead to a 3% reduction in the model's performance. Compared to the reasoning process, subgraph information is more crucial, indicating that effectively encoding the semantic information of KG in the retrieval model remains the key issue. Finally, command fine-tuning and continued pre-training also have a positive impact on model performance. Instruction tuning can improve the model's performance by about 0.7% across all metrics. Continued pre-training enhances the model's understanding of KG semantics, which helps to filter out irrelevant knowledge, thereby improving the model's precision and F1 score.

5.2 Retrieval Evaluation

The performance of retrieval-augmented KGQA models is largely dependent on the quality of the retrieval process (Jiang et al., 2023b). We expect retrieval models to exhibit exceptional recall capabilities to cover as much useful intermediate knowledge as possible. This is because while reasoning

LLMs may learn to filter out irrelevant information through training, they struggle to compensate for the absence of crucial information. Therefore, we compare the recall ability of *our* retrieval model, *ours w/o subgraph*, *ours w/o CoT*, and the *BGE* model (results are shown in Figure 3). It is evident that our retrieval model has a higher recall rate from top-5 to top-30 than the other three models, significantly surpassing the original BGE model. Comparing the performance of our model without CoT and without subgraph information, we find that subgraph information is more crucial for the retrieval model, consistent with the results of the ablation study in Section 5.1.

5.3 The Efficiency of YAML Format KG

As analyzed in Section 3.3, adopting the YAML format with simple syntax to represent KGs instead of the traditional triplet format can reduce token redundancy. To quantitatively assess how much redundancy YAML can eliminate, we have calculated the average number of KG tokens required per question by selecting knowledge graphs constructed from knowledge retrieved by our search engine on both WebQSP and CWQ datasets. For WebQSP, using triples to represent the KG requires an average of 532.6 tokens per question; if we use the YAML format, the average token drops to 384.2, thus reducing token redundancy by nearly 28%. For CWQ, replacing triples with YAML reduces the average token count of KGs from 534.3 to 401.4, a compression of nearly 25%. In a scenario where budget resources are constrained, minimizing the representation of tokens in a knowledge graph by using YAML allows those resources to be repurposed towards combining additional examples or recalling more retrieved information, aiming to achieve further performance enhancements.

5.4 Applying to Other Models

To verify the generalizability of our proposed method, we apply our method on two other different models, CodeLLaMA-7b-Instruct[6] (Rozière et al., 2024) and Phi2-3b[7] (Li et al., 2023). As shown in Figure 4, our method has significantly improved the performance of these two models on the KGQA task. For Phi2 and CodeLLaMA, our method has achieved an average improvement of 30% and 40% on the two datasets, respectively. Although CodeLLaMA is slightly inferior to LLaMA2-7b-chat, it still achieves performance comparable to RoG. Phi2, with only half the number of parameters compared to the other two models, lags significantly behind in performance, only reaching the level of UniKGQA and ChatGPT.

[6]https://huggingface.co/codellama/CodeLlama-7b-Instruct-hf

[7]https://huggingface.co/microsoft/phi-2

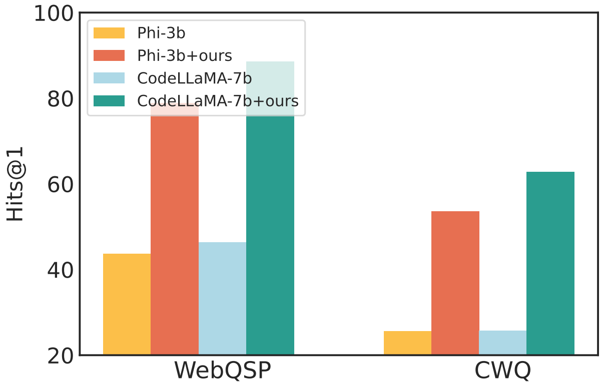

Figure 4: Experimental results on Phi2 and CodeLLaMA models.

We observe that the performance differences among the original three models on KGQA tasks are not significant. The original Phi2 and codellama exhibit a mere 1% difference on KGQA tasks; however, when combined with our approach, this margin increases to approximately 10%. Our method amplifies these differences, which may be due to understanding and exploiting KG to reason is a new skill for general-purpose LLMs. Phi2, with its smaller model size, may not allocate sufficient capacity to learn this skill. This phenomenon offers new insights for selecting a foundational model for KGQA in practice: firstly, within resource limits, choose models with larger parameters to fully learn and utilize KG capabilities; secondly, choose models with stronger reasoning abilities.

6 Conclusion

In this paper, we propose a method combining explainable knowledge graphs with large language models to enhance complex reasoning capabilities. Our method includes a KG retrieval model and a KG reasoning model. We integrate reasoning processes and subgraph information for better KG retrieval. We employ a novel KG representation and KG-related tuning for the reasoning model to learn to understand and reason with KG. Experimental results on two challenging KGQA tasks show that our method outperforms existing strong baselines and the SOTA model.

Limitations

Although our proposed method has made significant progress in KGQA, there are still some limitations:

- Due to computational resource constraints, we only conduct experiments on LLMs below 10B parameters, lacking investigation into larger models (such as LLaMA2-13B and 70B), other architectures (such as RWKV and Mixtral families).
- Our method fine-tunes LLMs with full-parameter, which is impractical in many low-resource settings. In future work, we plan to utilize efficient fine-tuning techniques such as LoRA, and compare its effectiveness with the current results.
- We only validate the efficacy of our method on two KGQA tasks. To more convincingly demonstrate that our approach enables LLMs to leverage KG for reasoning, we will incorporate additional tasks and datasets in our future work.

References

Jinheon Baek, Alham Fikri Aji, Jens Lehmann, and Sung Ju Hwang. 2023. Direct fact retrieval from knowledge graphs without entity linking. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 10038–10055, Toronto, Canada. Association for Computational Linguistics.

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*, SIGMOD '08, page 1247–1250, New York, NY, USA. Association for Computing Machinery.

Sébastien Bubeck, Varun Chandrasekaran, Ronen Eldan, Johannes Gehrke, Eric Horvitz, Ece Kamar, Peter Lee, Yin Tat Lee, Yuanzhi Li, Scott Lundberg, Harsha Nori, Hamid Palangi, Marco Tulio Ribeiro, and Yi Zhang. 2023. Sparks of artificial general intelligence: Early experiments with gpt-4.

Yunfan Gao, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yi Dai, Jiawei Sun, Qianyu Guo, Meng Wang, and Haofen Wang. 2024. Retrieval-augmented generation for large language models: A survey.

Claire Gardent, Anastasia Shimorina, Shashi Narayan, and Laura Perez-Beltrachini. 2017. Creating training corpora for NLG micro-planners. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long Papers*, pages 179–188. Association for Computational Linguistics.

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, Gyeongju, Republic of Korea. International Committee on Computational Linguistics.

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*, WSDM '21, page 553–561, New York, NY, USA. Association for Computing Machinery.

Xiaoxin He, Yijun Tian, Yifei Sun, Nitesh V. Chawla, Thomas Laurent, Yann LeCun, Xavier Bresson, and Bryan Hooi. 2024. G-retriever: Retrieval-augmented generation for textual graph understanding and question answering.

Lei Huang, Weijiang Yu, Weitao Ma, Weihong Zhong, Zhangyin Feng, Haotian Wang, Qianglong Chen, Weihua Peng, Xiaocheng Feng, Bing Qin, and Ting Liu. 2023. A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions.

Jinhao Jiang, Kun Zhou, Zican Dong, Keming Ye, Xin Zhao, and Ji-Rong Wen. 2023a. 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, Singapore. Association for Computational Linguistics.

Jinhao Jiang, Kun Zhou, Xin Zhao, and Ji-Rong Wen. 2023b. UniKGQA: Unified retrieval and reasoning for solving multi-hop question answering over knowledge graph. In *The Eleventh International Conference on Learning Representations*.

Tushar Khot, Harsh Trivedi, Matthew Finlayson, Yao Fu, Kyle Richardson, Peter Clark, and Ashish Sabharwal. 2023. Decomposed prompting: A modular approach for solving complex tasks. In *The Eleventh International Conference on Learning Representations*.

Woosuk Kwon, Zhuohan Li, Siyuan Zhuang, Ying Sheng, Lianmin Zheng, Cody Hao Yu, Joseph E. Gonzalez, Hao Zhang, and Ion Stoica. 2023. Efficient memory management for large language model serving with pagedattention. In *Proceedings of the ACM SIGOPS 29th Symposium on Operating Systems Principles*.

Yunshi Lan, Gaole He, Jinhao Jiang, Jing Jiang, Wayne Xin Zhao, and Ji-Rong Wen. 2021. A survey on complex knowledge base question answering: Methods, challenges and solutions. In *Proceedings* of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21*, pages 4483–4491. International Joint Conferences on Artificial Intelligence Organization. Survey Track.

Yunshi Lan and Jing Jiang. 2020. Query graph generation for answering multi-hop complex questions from knowledge bases. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 969–974, Online. Association for Computational Linguistics.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. In *Advances in Neural Information Processing Systems*, volume 33, pages 9459–9474. Curran Associates, Inc.

Yuanzhi Li, Sébastien Bubeck, Ronen Eldan, Allie Del Giorno, Suriya Gunasekar, and Yin Tat Lee. 2023. Textbooks are all you need ii: phi-1.5 technical report.

Weijie Liu, Peng Zhou, Zhe Zhao, Zhiruo Wang, Qi Ju, Haotang Deng, and Ping Wang. 2020. K-bert: Enabling language representation with knowledge graph. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(03):2901–2908.

Linho Luo, Yuan-Fang Li, Reza Haf, and Shirui Pan. 2024. Reasoning on graphs: Faithful and interpretable large language model reasoning. In *The Twelfth International Conference on Learning Representations*.

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, Austin, Texas. Association for Computational Linguistics.

OpenAI. 2022. Chatgpt: Optimizing language models for dialogue. *Open AI, blog*.

OpenAI. 2023. Gpt-4 technical report. *ArXiv*, abs/2303.08774.

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*, page 1–20.

Baptiste Rozière, Jonas Gehring, Fabian Gloeckle, Sten Sootla, Itai Gat, Xiaoqing Ellen Tan, Yossi Adi, Jingyu Liu, Romain Sauvestre, Tal Remez, Jérémy Rapin, Artyom Kozhevnikov, Ivan Evtimov, Joanna Bitton, Manish Bhatt, Cristian Canton Ferrer, Aaron Grattafiori, Wenhan Xiong, Alexandre Défossez, Jade Copet, Faisal Azhar, Hugo Touvron, Louis Martin, Nicolas Usunier, Thomas Scialom, and Gabriel Synnaeve. 2024. Code llama: Open foundation models for code.

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, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Haitian Sun, Tania Bedrax-Weiss, and William Cohen. 2019. PullNet: Open domain question answering with iterative retrieval on knowledge bases and text. 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 2380–2390, Hong Kong, China. Association for Computational Linguistics.

Haitian Sun, Bhuwan Dhingra, Manzil Zaheer, Kathryn Mazaitis, Ruslan Salakhutdinov, and William Cohen. 2018. Open domain question answering using early fusion of knowledge bases and text. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4231–4242, Brussels, Belgium. Association for Computational Linguistics.

Yawei Sun, Lingling Zhang, Gong Cheng, and Yuzhong Qu. 2020. Sparqa: Skeleton-based semantic parsing for complex questions over knowledge bases. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(05):8952–8959.

Yu Sun, Shuohuan Wang, Shikun Feng, Siyu Ding, Chao Pang, Junyuan Shang, Jiaxiang Liu, Xuyi Chen, Yanbin Zhao, Yuxiang Lu, Weixin Liu, Zhihua Wu, Weibao Gong, Jianzhong Liang, Zhizhou Shang, Peng Sun, Wei Liu, Xuan Ouyang, Dianhai Yu, Hao Tian, Hua Wu, and Haifeng Wang. 2021. Ernie 3.0: Large-scale knowledge enhanced pre-training for language understanding and generation.

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, New Orleans, Louisiana. Association for Computational Linguistics.

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.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, brian ichter, Fei Xia, Ed Chi, Quoc V Le, and Denny Zhou. 2022. Chain-of-thought prompting elicits reasoning in large language models. In

Advances in Neural Information Processing Systems, volume 35, pages 24824–24837. Curran Associates, Inc.

Zhengyuan Yang, Linjie Li, Kevin Lin, Jianfeng Wang, Chung-Ching Lin, Zicheng Liu, and Lijuan Wang. 2023. The dawn of lmms: Preliminary explorations with gpt-4v(ision).

Michihiro Yasunaga, Hongyu Ren, Antoine Bosselut, Percy Liang, and Jure Leskovec. 2021. QA-GNN: Reasoning with language models and knowledge graphs for question answering. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 535–546, Online. Association for Computational Linguistics.

Ruosong Ye, Caiqi Zhang, Runhui Wang, Shuyuan Xu, and Yongfeng Zhang. 2024. Language is all a graph needs. In *Findings of the Association for Computational Linguistics: EACL 2024*, pages 1955–1973, St. Julian's, Malta. Association for Computational Linguistics.

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, Dublin, Ireland. Association for Computational Linguistics.

Wen-tau Yih, Ming-Wei Chang, Xiaodong He, and Jianfeng Gao. 2015. Semantic parsing via staged query graph generation: Question answering with knowledge base. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1321–1331, Beijing, China. Association for Computational Linguistics.

Donghan Yu, Sheng Zhang, Patrick Ng, Henghui Zhu, Alexander Hanbo Li, Jun Wang, Yiqun Hu, William Yang Wang, Zhiguo Wang, and Bing Xiang. 2023. DecAF: Joint decoding of answers and logical forms for question answering over knowledge bases. In *The Eleventh International Conference on Learning Representations*.

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, Dublin, Ireland. Association for Computational Linguistics.

Zhengyan Zhang, Xu Han, Zhiyuan Liu, Xin Jiang, Maosong Sun, and Qun Liu. 2019. ERNIE: Enhanced language representation with informative entities. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 1441–1451, Florence, Italy. Association for Computational Linguistics.