

# KGFR: A Foundation Retriever for Generalized Knowledge Graph Question Answering

Yuanning Cui, Zequn Sun, Wei Hu, and Zhangjie Fu\*

**Abstract**—Large language models (LLMs) excel at reasoning but struggle with knowledge-intensive questions due to limited context and parametric knowledge. However, existing methods that rely on finetuned LLMs or GNN retrievers are limited by dataset-specific tuning and scalability on large or unseen graphs. We propose the **LLM–KGFR** collaborative framework, where an LLM works with a structured retriever, the *Knowledge Graph Foundation Retriever* (**KGFR**). KGFR encodes relations using LLM-generated descriptions and initializes entities based on their roles in the question, enabling zero-shot generalization to unseen KGs. To handle large graphs efficiently, it employs *Asymmetric Progressive Propagation* (APP)—a stepwise expansion that selectively limits high-degree nodes while retaining informative paths. Through node-, edge-, and path-level interfaces, the LLM iteratively requests candidate answers, supporting facts, and reasoning paths, forming a controllable reasoning loop. Experiments demonstrate that **LLM–KGFR** achieves strong performance while maintaining scalability and generalization, providing a practical solution for KG-augmented reasoning.

**Index Terms**—Question answering, knowledge graph, information retrieval, large language model, graph foundation model.

## I. INTRODUCTION

Large language models (LLMs) have shown impressive progress in natural language understanding and reasoning. Nevertheless, due to the finite scope of training corpora and the compression of knowledge into parameters, LLMs inevitably suffer from incomplete knowledge coverage and hallucinations [1]. To address this issue, external structured knowledge sources such as knowledge graphs (KGs) [2]–[6] offer an effective complement, providing factual grounding and enhancing reasoning reliability.

Despite their potential, integrating KGs into LLM-based question answering (QA) systems remains non-trivial. First, KGs are large and heterogeneous, making it infeasible to directly feed their content into LLMs due to token and memory constraints. Second, GNN-based approaches [7], [8] rely on KG-specific finetuning and full-graph message passing, which

\* Corresponding author

Yuanning Cui is with the School of Computer Science, Nanjing University of Information Science and Technology, Nanjing 210044, China (e-mail: yncui@nuist.edu.cn)

Zequn Sun is with the State Key Laboratory for Novel Software Technology, Nanjing University, Nanjing 210023, China (e-mail: sunzq@nju.edu.cn)

Wei Hu is with the State Key Laboratory for Novel Software Technology, Nanjing University, Nanjing 210023, China, and also with the National Institute of Healthcare Data Science, Nanjing University, Nanjing 210093, China (e-mail: whu@nju.edu.cn)

Zhangjie Fu is with the School of Computer Science, Nanjing University of Information Science and Technology, Nanjing 210044, China, and also with the Engineering Research Center of Digital Forensics, Ministry of Education, Nanjing University of Information Science and Technology, Nanjing 210044, China (e-mail: fzj@nuist.edu.cn).

limits both generalization and scalability on large graphs. Third, retrieval-augmented methods such as ToG [9] and RoG [10] depend on unstructured LLM queries or costly finetuning for relation-path generation, further constraining scalability and adaptability. These limitations motivate the following research question:

*How can we design a KG retrieval framework that generalizes to unseen graphs, scales to large KGs, and collaborates seamlessly with LLMs for reliable reasoning?*

To answer this question, we propose the **LLM–KGFR** collaborative framework, where **KGFR** (Knowledge Graph Foundation Retriever) serves as a question-conditioned graph retriever and works together with a frozen LLM. The overall framework is designed around three principles: generalization, scalability, and collaboration.

**First, generalization.** KGs differ in domains, vocabularies, and relation schemas, posing a key challenge for cross-dataset generalization. Many recent methods [7], [8] rely on dataset-specific LLM finetuning, where the model implicitly learns graph-specific patterns and thus struggles to transfer to new KGs without retraining. In contrast, our KGFR adopts a *question-conditioned initialization* that dynamically adapts entity embeddings according to the question context, assigning informative embeddings to mentioned entities and neutral ones to others. Furthermore, the LLM generates unified textual relation descriptions that are encoded as structured representations, enabling meaningful embeddings even for unseen relations and supporting cross-KG generalization without any tuning.

**Second, scalability.** Large KGs with millions of entities and edges make retrieval computationally intensive. Traditional GNNs expand neighborhoods uniformly, causing combinatorial growth and high memory usage around hub nodes. KGFR introduces an *Asymmetric Progressive Propagation* (APP) mechanism that expands layer by layer from topic entities while selectively constraining high-degree nodes to limit redundant growth. This asymmetric control retains informative links without inflating subgraphs, effectively balancing depth and breadth to achieve scalable retrieval on million-scale KGs.

**Third, collaboration.** While KGFR ensures efficient retrieval, effective reasoning relies on dynamic interaction with the LLM. To balance structural precision and semantic understanding, we design a *controller–retriever loop*: the LLM acts as a controller that reflects on intermediate results, reformulates or raises follow-up queries when information is missing, and issues new retrieval requests, while KGFR executes them and returns structured evidence through node-, edge-, and path-level interfaces. The LLM iteratively accesses candidate entities, supporting facts, and reasoning paths, while also

generating unified relation descriptions that refine KGFR’s retrieval space. This bidirectional collaboration enables controllable and interpretable reasoning beyond the capability of either component alone.

We evaluate our approach on seven QA benchmarks spanning diverse KG domains and reasoning tasks. Experimental results show that our framework consistently achieves strong accuracy, generalizes robustly to unseen datasets, and scales effectively to million-entity graphs.

Our main contributions are summarized as follows:

- **Generalization:** We design KGFR with LLM-guided relation initialization and question-conditioned propagation, enabling it to handle unseen entities and relations across heterogeneous KGs without retraining.
- **Scalability:** We propose an *Asymmetric Progressive Propagation (APP)* mechanism that selectively constrains high-degree expansions, effectively controlling subgraph growth and ensuring scalable retrieval on large graphs.
- **Collaboration:** We develop an *LLM–KGFR* framework, where KGFR offers multi-level retrieval and the LLM conducts iterative reasoning through reflection and reformulation.

The remainder of this paper is organized as follows. Section II reviews related work. Section IV introduces the Knowledge Graph Foundation Retriever (KGFR), and Section V presents the LLM–KGFR collaborative framework. Section VI reports experiments, and Section VII concludes the paper.

## II. RELATED WORK

*a) LLM reasoning with KGs:* LLMs often hallucinate on knowledge-intensive tasks [1], prompting the use of structured KGs for factual grounding and improved reliability. Existing studies fall into two categories: (1) constructing task-specific graphs from texts to enhance retrieval and summarization [11]–[14], and (2) leveraging open KGs as external evidence for natural-language questions. Within the first line, GFM-RAG [14] employs a graph foundation model to improve text retrieval and summarization, but its task objective, input–output form, and benchmarks differ substantially from KG-based QA, making it not directly comparable to our work. Representative methods in the second line include KD-CoT [15], which retrieves KG facts to guide chain-of-thought reasoning, and agent-style frameworks such as StructGPT [16] and ToG [9], where LLMs interact with KGs to explore reasoning paths. EffiQA [17] further introduces a compact plug-in retriever to efficiently explore entities and relations. LightPROF [18] encodes KG structures into soft prompts via a lightweight adapter. RoG [10] and GCR [19] adopt planning–retrieval–reasoning pipelines that return KG paths as structured evidence. To better exploit graph topology, GNN-RAG [8] integrates GNN-based retrieval, and G-Retriever [7] applies GNN-based prompt tuning; however, both rely on KG-specific training or finetuned LLMs, which hinders cross-KG generalization and scalability.

*b) KG foundation models:* Early graph foundation models (GFMs) follow a pretraining–finetuning paradigm to transfer across datasets [20]–[24]. Prompt-based GFMs [25]–[32]

further improve adaptability via universal prompt templates for diverse graph-level, edge-level, and node-level tasks. Recently, several KG-oriented foundation models have targeted relational reasoning and KG completion [33]–[36]. These approaches primarily focus on graph analytics or completion objectives, whereas our setting centers on natural-language QA over KGs with a collaborative LLM–retriever workflow.

## III. PRELIMINARIES

*a) Knowledge graph question answering:* KGQA is the task of answering natural language questions based on the facts in a given KG. The background KG is formulated as  $\mathcal{G} = (\mathcal{E}, \mathcal{R}, \mathcal{T})$ , where  $\mathcal{E}$ ,  $\mathcal{R}$ , and  $\mathcal{T}$  are the sets of entities, relations, and facts, respectively. A fact, formed as  $(s, r, o) \in \mathcal{T}$ , represents a directed edge between entities, where  $s$  and  $o$  are the subject and object entities from  $\mathcal{E}$ , respectively, and  $r$  is the relation between them. A question is formed as  $(q, \mathcal{A}_q, \mathcal{P}_q)$ , where  $q$  is the natural language question,  $\mathcal{A}_q \subseteq \mathcal{E}$  is the set of answer entities, and  $\mathcal{P}_q \subseteq \mathcal{E}$  is the set of topic entities. Each question contains at least one topic entity, which anchors the natural language question to the KG. The KG facts about the topic entities in the question are the basis for answering the question. For complex questions, KGQA methods should be capable of reasoning over the subgraphs surrounding the topic entities, filtering out irrelevant information, and aggregating useful relational facts.

*b) Generalized KGQA:* Generalized KGQA aims to answer questions over both seen and unseen KGs. In the generalized KGQA task, during the reasoning phase, a KG  $\mathcal{G}' = (\mathcal{E}', \mathcal{R}', \mathcal{T}')$  is given, which may differ from the one used in the training set. A generalized KGQA question is formalized as  $(q, \mathcal{A}_q, \mathcal{P}_q, \mathcal{H}_q)$ , where  $q$  and  $\mathcal{P}_q \subseteq \mathcal{E}'$  are defined in the same way as in KGQA.  $\mathcal{H}_q$  represents a set of candidate answers, and  $\mathcal{A}_q \subseteq \mathcal{H}_q$  is the set of correct answers. For example, in multiple-choice questions,  $\mathcal{H}_q$  is the set of all options, while  $\mathcal{A}_q$  contains the correct options. KGQA can be seen as a special case of generalized KGQA, where the candidate answers are all entities, i.e.,  $\mathcal{H}_q = \mathcal{E}$ . The original KG is a directed graph. To enhance its connectivity, following the convention [37], [38], we incorporate inverse relations and facts into the KG. Specifically, for every fact  $(s, r, o) \in \mathcal{T}$ , we introduce a reverse fact  $(o, r^{-1}, s)$ , where  $r^{-1}$  is the inverse relation of  $r$ . In the following sections, we assume that inverse relations and facts are included in  $\mathcal{R}$  and  $\mathcal{T}$ .

## IV. KNOWLEDGE GRAPH FOUNDATION RETRIEVER

Figure 1 presents an overview of the framework. We propose the Knowledge Graph Foundation Retriever (KGFR), a question-conditioned retriever for generalized and scalable knowledge retrieval without finetuning. For **generalization**, KGFR employs language-guided initialization, encoding relations from LLM-generated descriptions and initializing entities by their roles in the question. For **scalability**, it performs asymmetric progressive propagation from topic entities while constraining high-degree nodes. For **collaboration**, it exposes node-, edge-, and path-level interfaces that interact with the LLM for iterative reasoning and reflection.

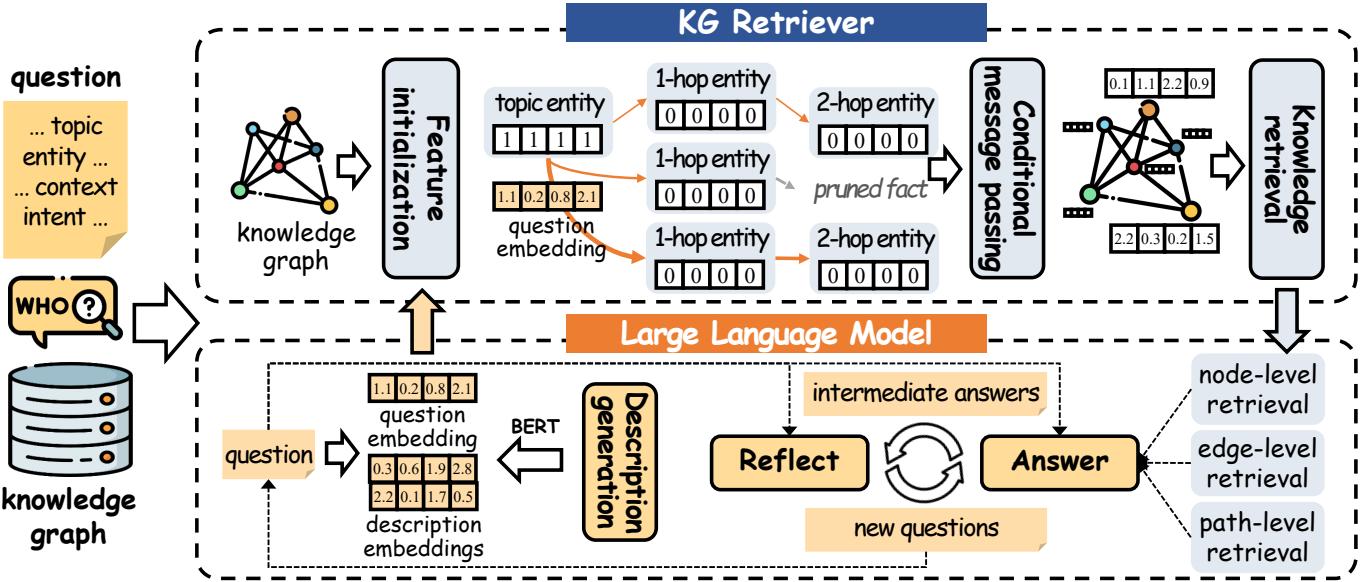


Fig. 1: Framework of LLM-KGFR. Given a question and a KG, we encode the question and relations using BERT and perform asymmetric progressive propagation from the topic entity. The KGFR then conducts multi-level (node, edge, path) retrieval to iteratively generate and refine answers with the LLM until the final answer is confirmed.

#### A. Relation Initialization via Unified Descriptions

A key challenge to *generalization* in KGQA is the inconsistent naming and formatting of relations across KGs. Semantically equivalent relations often appear under different identifiers or conventions (e.g., Freebase /location/country/capital, Wikidata P36, or natural-language forms such as “is the capital of”), making relation names unreliable indicators of semantics. As a result, retrievers trained on one KG struggle to transfer to others with distinct relation vocabularies.

To address this, KGFR leverages LLMs to generate *unified textual descriptions* for relations from their names and a few example triples. These descriptions abstract away dataset-specific identifiers and capture consistent relation semantics in natural language, which are then encoded as initial relation embeddings. An illustrative prompt is shown in Figure 2.

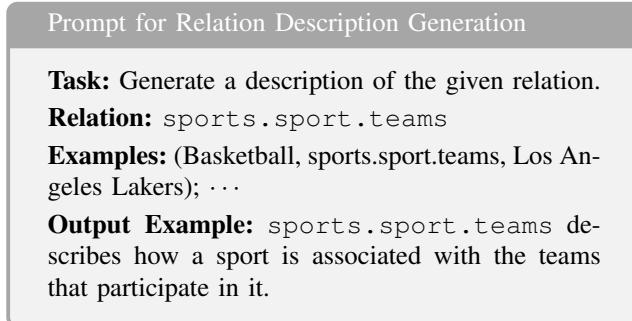


Fig. 2: Illustrative prompt used for generating unified textual descriptions of relations.

By normalizing relations into such unified descriptions, KGFR aligns heterogeneous schemas with natural language

questions, enabling question-conditioned message passing and robust cross-KG generalization.

#### B. Question-Conditioned Message-Passing

Our KGFR’s message-passing mechanism addresses two key challenges: enabling natural language understanding and ensuring strong generalization to unseen graphs. Specifically, we integrate the BERT-encoded question representations into both relation embedding initialization and attention computation, enabling the KGFR to process linguistic inputs. To ensure generalization to unseen KGs, we employ non-learnable entity vectors [34]–[36], [39], [40] during this process.

**Feature initialization.** Given a question  $q$ , a KG  $\mathcal{G} = \{\mathcal{E}, \mathcal{R}, \mathcal{T}\}$ , and a topic entity set  $\mathcal{P}_q$ , we first initialize the features of relations and entities.

For relations, we initialize their features using the textual embeddings of their descriptions:

$$\mathbf{r}^{(0)} = \text{BERT}(u_r), \quad (1)$$

where  $\text{BERT}()$  denotes the BERT encoder [41], and  $u_r$  is the textual description of the relation  $r$ .

For entities, KGFR adopts a lightweight yet robust initialization scheme widely used in recent KG foundation models [34]–[36]. Instead of encoding millions of entity names—which is computationally expensive and often inconsistent across KGs—topic entities are assigned fixed one-vectors  $\mathbf{1}_d$ , and all others zero-vectors  $\mathbf{0}_d$ , where  $d$  is the embedding dimension. This initialization avoids dependency on noisy textual features, maintains efficiency on large-scale KGs, and has been empirically shown to support generalization to unseen graphs.

**Question-conditioned message-passing.** Next, we propagate messages through the graph to update the embeddings of

the relations and entities. In the  $i$ -th layer, we first update the embeddings of the relations as follows:

$$\mathbf{r}^{(i+1)} = \mathbf{W}_1^{(i)} \left[ \mathbf{r}^{(i)} ; \text{BERT}(q) \right], \quad (2)$$

where  $\mathbf{W}_1^{(i)} \in \mathbb{R}^{d \times 2d}$  is a learnable weight matrix, and  $[;]$  is the concatenation operation. In this way, we retain the message from the previous layer while injecting question-conditioned information at each layer. Then, we adopt a progressive propagation [40] to update the embeddings of the entities as follows:

$$\mathbf{e}^{(i+1)} = \sum_{(s,r,e) \in \mathcal{N}_{\mathcal{P}_q}^{(i)}(e)} \mathbf{W}_2^{(i)} \text{MSG}^{(i)}(s, r, q), \quad (3)$$

where  $\mathbf{W}_2^{(i)} \in \mathbb{R}^{d \times d}$  is a learnable weight matrix, and  $\mathcal{N}_{\mathcal{P}_q}^{(i)}(e)$  is the set of facts that both includes entity  $e$  and is within the  $i$ -hop neighbors of the topic entities.  $\text{MSG}^{(i)}()$  is the message function:

$$\text{MSG}^{(i)}(s, r, q) = \alpha_{s;r;q}^{(i)} \left( \mathbf{s}^{(i)} + \mathbf{r}^{(i)} \right), \quad (4)$$

where  $\mathbf{s}^{(i)}$  and  $\mathbf{r}^{(i)}$  are the  $i$ -th layer embeddings of  $s$  and  $r$ , respectively.  $\alpha_{s;r;q}^{(i)}$  is the attention of the edge, which is calculated by

$$\alpha_{s;r;q}^{(i)} = f \left( \mathbf{W}_3^{(i)} g \left( \mathbf{W}_4^{(i)} \mathbf{s}^{(i)} + \mathbf{W}_5^{(i)} \mathbf{r}^{(i)} + \mathbf{W}_6^{(i)} \mathbf{q} \right) \right), \quad (5)$$

where  $f()$  and  $g()$  denote the activation functions sigmoid and ReLU, respectively.  $\mathbf{W}_3^{(i)} \in \mathbb{R}^{1 \times d_{\text{attn}}}$ ,  $\mathbf{W}_4^{(i)}$ ,  $\mathbf{W}_5^{(i)}$ , and  $\mathbf{W}_6^{(i)} \in \mathbb{R}^{d_{\text{attn}} \times d}$  are learnable weight matrices, where  $d_{\text{attn}}$  is a hyperparameter that reduces dimensionality.  $\mathbf{q} = \text{BERT}(q)$ .

### C. Asymmetric Progressive Propagation

To ensure scalability on large KGs, we design the *Asymmetric Progressive Propagation* (APP) mechanism, which integrates two complementary principles: **progressive expansion** and **asymmetric pruning**.

**Progressive expansion.** Natural-language questions often imply a multi-hop reasoning process. APP follows this intuition by starting from the topic entities  $\mathcal{P}_q$  and expanding one hop at a time. At each hop, the newly reached edges are merged with previously retrieved ones to form a progressively enlarged retrieval subgraph. This progressive expansion keeps the retrieval scope localized around relevant entities instead of the entire KG, enabling efficient large-scale deployment.

**Asymmetric pruning.** Naïve propagation tends to suffer from high-degree relations that cause uncontrolled growth. For instance, entities such as *China* may participate in relations like (*China*, *citizens*, ?) that connect to millions of nodes, most of which are irrelevant. However, pruning the entity itself would also remove useful edges such as (*China*, *capital*, ?) or (*China*, *official language*, ?). APP therefore performs pruning at the  $(s, r, ?)$  level: when a relation type yields excessive neighbors, further expansion along that relation is suppressed, while other relations of  $s$  remain available for propagation. This asymmetric rule effectively controls hub-induced explosion without discarding informative reasoning paths.

**Formalization.** Let  $\mathcal{N}_q^{(i)}(e)$  denote the set of edges reachable from entity  $e$  at hop  $i$ . Progressive propagation is defined as

$$\mathcal{N}_q^{(i+1)}(e) = \mathcal{N}_q^{(i)}(e) \cup \bigcup_{(e,r,o) \in \mathcal{T}} \text{Expand}(e, r), \quad (6)$$

where  $\text{Expand}(e, r)$  inserts edges  $(e, r, o)$  into the retrieval frontier. Define  $C_{e,r} = \{o \mid (e, r, o) \in \mathcal{T}\}$  as the candidate neighbor set. Then

$$\text{Expand}(e, r) = \begin{cases} \{(e, r, o) \mid o \in C_{e,r}\}, & |C_{e,r}| \leq \lambda, \\ \{(e, r, o) \mid o \in \mathcal{S}_i\}, & |C_{e,r}| > \lambda, \end{cases} \quad (7)$$

where  $\mathcal{S}_i = \bigcup_{i=0}^i \left\{ x \mid \exists (u, r, v) \in \mathcal{N}_q^{(i)}(u), x \in \{u, v\} \right\}$  is the cumulative set of entities reached up to hop  $i$ .  $\lambda$  is a threshold controlling the maximum number of neighbors expanded per relation, preventing high-degree nodes from overwhelming the propagation.

APP provides three properties: (1) localized reasoning via progressive expansion from topic entities; (2) preservation of useful paths while pruning noisy high-degree relations; (3) control of hub-induced explosion for a tractable, semantically sufficient subgraph. These properties enable APP to scale to large KGs while remaining consistent with stepwise reasoning in natural-language questions.

### D. Pre-training Objective

Since we have already incorporated the question as a condition in the question-conditioned message-passing process, we directly read the entity embeddings from the last layer to compute their scores:

$$c_{e|q} = \mathbf{W}_7 \mathbf{e}^{(L)}, \quad (8)$$

where  $\mathbf{W}_7 \in \mathbb{R}^{1 \times d}$  is a learnable weight matrix, and  $L$  is the number of message-passing layers.

Given the background KG  $\mathcal{G} = (\mathcal{E}, \mathcal{R}, \mathcal{T})$  and a set of question-answering training data  $\mathcal{D} = \{(q_1, \mathcal{A}_{q_1}, \mathcal{P}_{q_1}), \dots, (q_n, \mathcal{A}_{q_n}, \mathcal{P}_{q_n})\}$ , we introduce a variant of the multi-class log-loss function [42] to pre-train the KGFR:

$$\mathcal{L} = \sum_{(q_i, \mathcal{A}_{q_i}, \mathcal{P}_{q_i}) \in \mathcal{D}} \left[ \underbrace{\log \sum_{x \in \mathcal{E}} \exp(c_{x|q_i})}_{\text{All candidates (denominator)}} - \underbrace{\log \sum_{a \in \mathcal{A}_{q_i}} \exp(c_{a|q_i})}_{\text{Positive set (numerator)}} \right]. \quad (9)$$

## V. COLLABORATIVE QUESTION ANSWERING WITH LLMs

While the previous section establishes KGFR as a general and scalable retriever, effective question answering requires it to operate in concert with a language model capable of reasoning over retrieved evidence. This section introduces the **LLM–KGFR collaborative framework**, where the LLM and KGFR jointly perform QA through a controller–executor interaction loop (Figure 1). The LLM interprets questions, formulates relation descriptions, and decides when additional retrieval is needed, while KGFR executes APP-based propagation and returns structured evidence at node, edge, and path levels. Through iterative retrieval, reasoning, and reflection, this collaboration enables large language models to reason over vast KGs efficiently and transparently.

### A. KGFR-based Knowledge Retrieval

To enable collaborative QA, KGFR exposes structured retrieval interfaces that let the LLM acquire information at different granularities. Rather than returning raw neighbors or invoking additional models, KGFR organizes its outputs into three complementary levels: **node-level** candidate entities, **edge-level** supporting facts, and **path-level** connections to topic entities. These retrievals operate on the *retrieval subgraph*  $\mathcal{G}_q^{(L)} = (\mathcal{S}_L, \mathcal{N}_q^{(L)})$  produced by APP after  $L$  hops, where  $\mathcal{S}_L$  is the set of reached entities and  $\mathcal{N}_q^{(L)}$  the set of reached edges. This design allows the LLM to flexibly request coarse-to-fine evidence depending on the reasoning stage, while preserving scalability and generalization.

**Node-level retrieval.** KGFR ranks entities within the current subgraph and returns the top- $k$  candidates:

$$\mathcal{C}_q = \text{Top}_k \{ e : c_{e|q} \mid e \in \mathcal{S}_L \}, \quad (10)$$

where  $c_{e|q}$  is the question-conditioned score of entity  $e$ . This narrows the search space of a large KG into a concise, ranked set of plausible answers for the LLM to examine.

**Edge-level retrieval.** To provide factual grounding, KGFR retrieves the most relevant edges for an entity  $e$  within the subgraph:

$$\mathcal{I}_{q;e} = \text{Top}_n \{ (s, r, e) : \alpha_{s;r;q}^{\max} \mid (s, r, e) \in \mathcal{N}_q^{(L)} \}, \quad (11)$$

where  $\alpha_{s;r;q}^{\max} = \max_{1 \leq i \leq L} \alpha_{s;r;q}^{(i)}$  is the maximum attention weight across message-passing hops. Let  $\mathcal{I}_q = \bigcup_{e \in \mathcal{C}_q} \mathcal{I}_{q;e}$  be the set of relevant edges.

**Path-level retrieval.** To reveal multi-hop reasoning chains, KGFR computes the shortest paths in  $\mathcal{G}_q^{(L)}$  from each important entity to every topic entity in  $\mathcal{P}_q = \{e_q^{(1)}, e_q^{(2)}, \dots\}$ :

$$\mathcal{P}_q^{\text{path}} = \bigcup_{e \in \mathcal{C}_q} \bigcup_{e_q \in \mathcal{P}_q} \text{SP}(e, e_q; \mathcal{G}_q^{(L)}), \quad (12)$$

where SP returns directed shortest paths between  $e$  and  $e_q$  within the subgraph  $\mathcal{G}_q^{(L)}$ . This deterministic retrieval avoids training path-generating LLMs while ensuring efficiency, interpretability, and transferability to unseen KGs.

Together, node-, edge-, and path-level retrieval grant the LLM modular access to KG evidence—candidates to narrow the space, supporting facts to ground decisions, and paths to expose reasoning chains—enabling scalable and robust collaborative QA.

### B. LLM-based Generation and Reflection

The LLM in the LLM–KGFR framework plays a controlling role in driving the QA process, complementing KGFR’s structured retrieval with language-based reasoning. Its role spans three dimensions: generating and aligning relation representations, reflecting on intermediate answers, and adaptively refining retrieval.

**Generation and fact verbalization.** Since language and graph representations differ substantially, effective communication requires a unified representational space bridging the two modalities. First, the LLM generates *relation descriptions* for each relation (Section IV-A); these unified textual descriptions serve as anchors for initializing relation embeddings in

KGFR. Separately, the LLM also induces *verbalization templates* for each relation. Subsequently, when KGFR retrieves factual triples  $(s, r, o)$ , these templates are applied to verbalize the structured facts into natural sentences, allowing the evidence to be seamlessly integrated into subsequent reasoning steps. Together, this two-part design—unified relation descriptions for initialization and relation-specific verbalization templates for natural-language rendering—enables KGFR to align with linguistic semantics and helps the LLM accurately interpret graph-based retrieval results.

**Answer generation and reflection.** Given the candidate entities, supporting facts, and reasoning paths retrieved by KGFR, the LLM synthesizes a coherent natural-language answer through contextual reasoning and factual aggregation. After each generation round, it performs a *reflection step* to evaluate the sufficiency and consistency of the produced answer. If the retrieved evidence adequately supports the conclusion and no contradictions are detected, the reasoning cycle terminates. Otherwise, the LLM analyzes which parts of the reasoning chain remain uncertain or underspecified, identifies missing entities or relations, and formulates a targeted follow-up query to KGFR.

**Adaptive question rewriting and entity focusing.** When retrieval is incomplete, the LLM can rewrite the original question into auxiliary sub-questions  $\mathcal{Q}_q$  to guide further retrieval [9], [43], [44]. It also identifies key entities from the current reasoning context and directs KGFR to concentrate subsequent retrieval on these entities. The rewritten questions and selected entities are then fed into the next iteration, enabling progressively refined reasoning.

### C. Reasoning Pipeline

For clarity, we describe the complete reasoning workflow of the LLM–KGFR collaboration below. Algorithm 1 outlines the full reasoning pipeline of the framework, which proceeds in two stages given a question  $q$ , topic entities  $\mathcal{P}_q$ , and a KG  $\mathcal{G} = (\mathcal{E}, \mathcal{R}, \mathcal{T})$ .

**Initial retrieval.** The LLM first generates unified textual descriptions for all relations, which are encoded by a frozen BERT encoder to initialize relation embeddings (Section IV-A). Entities are initialized using KGFR’s initialization strategy described in Section IV-B. KGFR then performs question-conditioned propagation with APP and produces three types of evidence within the retrieval subgraph: (i) node-level candidate entities  $\mathcal{C}_q$ , (ii) edge-level important facts  $\mathcal{I}_q$ , and (iii) path-level connections  $\mathcal{P}_q^{\text{path}}$  obtained by shortest paths from retrieved entities to topic entities. This stage builds a compact, question-specific subgraph without any dataset-specific training, ensuring both efficiency and generalization.

**Iterative retrieval.** Based on the retrieved evidence, the LLM synthesizes answers and performs reflection. If the evidence is insufficient, the LLM may (i) rewrite the question into sub-questions  $\mathcal{Q}_q$  to trigger further retrieval, or (ii) identify key entities to focus additional edge- or path-level exploration. This reasoning loop continues until the answer is confirmed or a maximum number of steps is reached.

TABLE I: Dataset statistics. “S” and “C” denote “Support” and “Counter”, respectively.

Answer types	Datasets	Entity	Relation	Fact	Training	Development	Testing	KGs
Entities	WebQSP	1,298,306	6,094	3,791,303	2,848	250	1,639	Freebase
	CWQ	2,259,510	6,649	7,269,449	27,639	3,519	3,531	Freebase
	GrailQA	430,781	6,484	2,408,034	-	-	1,000	Freebase
S / C	ExplaGraphs	6,331	28	9,771	-	-	2,766	ConceptNet
Choices	CSQA	34,869	16	257,767	-	-	1,241	ConceptNet
	OBQA	22,961	16	173,102	-	-	500	ConceptNet
	MedQA	3,364	15	15,265	-	-	1,273	UMLS & DrugBank

**Algorithm 1:** Reasoning Pipeline of LLM–KGFR

---

**Input:** Question  $q$ , topic entities  $\mathcal{P}_q$ , and KG  $\mathcal{G} = (\mathcal{E}, \mathcal{R}, \mathcal{T})$ .

**Output:** Predicted answers  $\mathcal{A}_{q;\text{pred}}$ .

```

/* Stage 1: Initial retrieval */ 
1 Encode descriptions with BERT to initialize relation
   embeddings;
2 Initialize entity embeddings;
3 Run question-conditioned APP propagation;
4 Retrieve candidate entities  $\mathcal{C}_q$ ;
5 Retrieve important facts  $\mathcal{I}_q$  around  $\mathcal{P}_q \cup \mathcal{C}_q$ ;
6 Compute path-level connections  $\mathcal{P}_q^{\text{path}}$  via shortest
   paths;
/* Stage 2: Iterative retrieval */ 
7 for  $\text{step} \leftarrow 1$  to  $\text{max\_steps}$  do
8   Generate answer candidates  $\mathcal{A}_{q;\text{pred}}$  (LLM);
9   Reflect on evidence sufficiency and answer
      consistency;
10  if evidence insufficient then
11    if LLM produces sub-questions  $\mathcal{Q}_q$  then
12      Trigger new retrieval for  $\mathcal{Q}_q$ ;
13    if LLM identifies key entities then
14      Focus additional edge retrieval around
         selected entities;
15  if answer confirmed then
16    break;
17 return final answers  $\mathcal{A}_{q;\text{pred}}$ .

```

---

## VI. EXPERIMENTS AND RESULTS

Our evaluation is guided by four research questions (RQs):

- **RQ1 (Performance):** How effectively does LLM-KGFR perform on KGQA compared with existing methods?
- **RQ2 (Generalization):** Can LLM-KGFR generalize to unseen KGs and QA datasets with heterogeneous schemas?
- **RQ3 (Scalability and Efficiency):** Does LLM-KGFR enable efficient inference on large KGs while preserving accuracy?
- **RQ4 (Module Effectiveness):** What is the contribution of each proposed module to the overall performance?

We evaluate LLM-KGFR on seven KGQA benchmarks covering diverse domains, scales, and question styles. The source code is available at <https://github.com/yncui-nju/KGFR>.

## A. Settings

a) **Datasets:** We evaluate LLM-KGFR on seven datasets that span factual, compositional, and commonsense reasoning, as summarized in Table I.

WebQSP [45] contains 4,737 natural language questions requiring up to 2-hop reasoning, while CWQ [46] includes 34,699 complex questions involving up to 4 hops. Both are built upon Freebase as the background KG. Following prior work [8], [10], [47]–[50], we first extract local subgraphs centered on topic entities for each dataset. These subgraphs are then merged into dataset-specific background KGs used during inference, whose overall scales remain comparable to those adopted in previous large-KG QA studies [51]–[53].

GrailQA [54] further evaluates generalization to unseen domains and compositional query structures. It is also based on Freebase and undergoes the same subgraph extraction and merging procedure. Following [55], we do not perform additional training on GrailQA and directly evaluate the retriever pre-trained on CWQ in a zero-shot setting.

ExplaGraphs [7], [56] provides explanation graphs for determining whether two arguments are supportive or contradictory. CommonSenseQA (CSQA) [57] and OpenBookQA (OBQA) [58] are multiple-choice QA datasets built on ConceptNet [59], with 1,241 and 500 test questions (five and four options, respectively) following [60], [61]. MedQA [62] is a biomedical multiple-choice dataset (four options) whose background KG integrates UMLS [63] and DrugBank [64].

b) **Implementation details:** KGFR adopts a 3-layer message-passing backbone. The threshold of the asymmetric pruning module is set to  $\lambda=100$ . We use BGE-Large-EN-v1.5<sup>1</sup> as the BERT encoder to encode both the question and the relation descriptions, keeping encoder parameters frozen during all experiments. The hidden dimension and attention head count are set to  $d=1024$  and  $d_{\text{attn}}=4$ , respectively. Both the top- $k$  candidate size and neighbor selection  $n$  are set to 20. The LLM reasoning loop is limited to three iterations. Optimization uses the Adam optimizer with a learning rate of  $1e-4$ , a maximum of 200 epochs, and early stopping with patience 5. In the main evaluation, we adopt four LLMs—Qwen-max,<sup>2</sup> GPT-4o-mini,<sup>3</sup> GPT-4-Turbo,<sup>4</sup> and GPT-4<sup>5</sup>—and report results on WebQSP and CWQ. For cost considerations, subsequent analyses

<sup>1</sup><https://huggingface.co/BAAI/bge-large-en-v1.5>

<sup>2</sup><https://qwenlm.github.io/blog/qwen2.5-max/>

<sup>3</sup><https://openai.com/index/gpt-4o-mini-advancing-cost-efficient-intelligence/>

<sup>4</sup><https://openai.com/index/hello-gpt-4o/>

<sup>5</sup><https://openai.com/index/gpt-4/>

TABLE II: Results on WebQSP and CWQ. The best score on each metric is in **bold**, and the second best score is underlined.

Types	Methods	WebQSP			CWQ		
		F1	Hit	H@1	F1	Hit	H@1
Embedding	KV-Mem [65]	38.6	46.7	-	-	21.1	-
	EmbedKGQA [38]	-	66.6	-	-	45.9	-
	TransferNet [66]	-	71.4	-	-	48.6	-
	Rigel [67]	-	73.3	-	-	48.7	-
GNN	GraftNet [50]	60.4	66.4	-	32.7	36.8	-
	PullNet [48]	-	68.1	-	-	45.9	-
	NSM [49]	62.8	68.7	-	42.4	47.6	-
	SR + NSM (+E2E) [68]	64.1	69.5	-	47.1	50.2	-
	NSM + h [69]	67.4	74.3	-	44.0	48.8	-
	SQALER [70]	-	76.1	-	-	-	-
	UniKGQA [47]	72.2	77.2	-	49.1	51.2	-
	ReaRev+LMsr [71]	72.8	77.5	-	49.7	53.3	-
	KD-CoT [15]	52.5	68.6	-	-	55.7	-
KG + LLM	StructGPT [16]	-	72.6	-	-	54.3	-
	KB-BINDER [72]	-	74.4	-	-	-	-
	ToG + GPT-4 [9]	-	82.6	-	-	69.5	-
	RoG [10]	70.8	85.7	80.0	56.2	62.6	57.8
	PoG [55]	-	87.3	-	-	<b>75.0</b>	-
	EffiQA [17]	-	82.9	-	-	69.5	-
	LightPROF [18]	-	83.8	-	-	59.3	-
	G-Retriever [7]	-	73.8	-	-	-	-
GNN + LLM	GNN-RAG [8]	71.3	85.7	80.6	59.4	66.8	61.7
	LLM-KGFR (Qwen-max)	<u>74.7</u>	<b>90.3</b>	<u>83.2</u>	61.6	71.8	<u>63.6</u>
Ours	LLM-KGFR (GPT-4o-mini)	69.0	<u>89.4</u>	80.0	53.7	<u>72.3</u>	62.1
	LLM-KGFR (GPT-4-turbo)	<b>76.2</b>	88.7	<b>83.4</b>	<b>62.3</b>	70.8	<b>63.9</b>
	LLM-KGFR (GPT-4)	73.4	89.0	81.4	<u>61.9</u>	72.2	63.0

adopt Qwen-max and GPT-4o-mini. For ExplaGraphs, CSQA, OBQA, and MedQA, we use the frozen KGFR pre-trained on CWQ. Model pre-training was done on a workstation featuring two Intel Xeon Gold CPUs, four NVIDIA A800 (80 GB) GPUs, and Ubuntu 18.04 LTS, whereas all evaluation was carried out on a server with four NVIDIA A6000 (48 GB) GPUs. The retriever contains 28 MB parameters.

c) *Baselines:* We compare LLM-KGFR with four categories of methods:

(i) **Embedding-based.** KV-Mem [65] uses a key-value memory network for KGQA. EmbedKGQA [38] leverages pre-trained embeddings for multi-hop reasoning. TransferNet [66] improves reasoning within the relation set. Rigel [67] enhances reasoning for questions involving multiple entities.

(ii) **GNN-based.** GraftNet [50] employs a convolutional GNN. PullNet [48] builds on GraftNet and learns to retrieve nodes via shortest paths to answers. NSM [49] adapts GNNs for KGQA, and NSM+h [69] improves multi-hop reasoning. SQALER [70] selects which facts to retrieve during GNN reasoning. SR+NSM (+E2E) [68] proposes relation-path retrieval. ReaRev+LMsr [71] explores diverse reasoning paths in a multi-stage manner.

(iii) **KG-enhanced LLM-based.** KD-CoT [15] augments chain-of-thought prompting with KG facts. StructGPT [16] retrieves KG facts for RAG. KB-BINDER [72] improves reasoning via logical forms. ToG [9] selects relevant facts step-by-step with a strong LLM. RoG [10] finetunes an LLM to generate relation paths for planning. PoG [55] performs self-

correcting planning via sub-goals and reflection. EffiQA [17] creates sub-questions and pseudo answers with LLMs to guide a lightweight plug-in retriever over the KG. LightPROF [18] encodes KG structure into soft prompts via a lightweight adapter.

(iv) **GNN-enhanced LLM-based.** G-Retriever [7] uses GNN-based prompt tuning to assist LLMs. GNN-RAG [8] integrates a GNN for KG retrieval and finetunes an LLM for question answering.

d) *Evaluation protocol:* We follow previous works [8]–[10] to adopt F1, Hit, and H@1 as the evaluation metrics. F1 measures the overall quality of answers by balancing the precision and recall of the predictions. Hit checks for the presence of at least one correct answer among the final predictions. H@1 calculates the proportion of instances where one correct answer is the first predicted entity. We also adopt accuracy as a metric in CSQA, OBQA, MedQA, and ExplaGraphs.

## B. RQ1: Performance

a) *Main results:* Table II presents the main experimental results. We can observe the following findings: (i) LLM-KGFR outperforms existing methods on most metrics across the four LLMs, particularly when using Qwen-max and GPT-4-turbo as the LLMs. This indicates that our approach is versatile across state-of-the-art LLMs. (ii) Among the various models, GPT-4-turbo exhibits the strongest performance, followed by Qwen-max and GPT-4, with GPT-4o-mini ranking last. Our method is

influenced by both the LLM and KGFR, and we anticipate that our method can improve as more advanced models become available in the future. Notably, even the smallest GPT-4o-mini model achieves commendable scores. (iii) The WebQSP dataset is relatively simple, with the maximum hop of 2, whereas CWQ is more challenging. Comparing the results on both datasets, GPT-4o-mini shows a larger gap in F1 scores on CWQ compared to the other two variants. This difference arises because more complex questions demand higher levels of language understanding and decision-making capabilities from the LLMs. (iv) Despite GPT-4o-mini’s lower F1 score on CWQ, it achieves a high Hit rate. This can be attributed to the more advanced models, Qwen-max and GPT-4-turbo, which tend to be more conservative in their responses. In contrast, GPT-4o-mini tends to include more potential candidates.

Overall, these results confirm that LLM–KGFR consistently outperforms prior KGQA systems, scales across diverse LLMs, and remains robust on complex reasoning tasks.

*b) Comparison under unified LLM settings:* To ensure fairness, we align all methods under the same LLM backbones. We use two representative models—Llama2-7B and GPT-4—covering most baselines. Finetuned or LoRA-based models (e.g., KD-CoT, RoG, G-Retriever, GNN-RAG) use Llama2-7B. Frozen-LLM methods include ToG, PoG, EffiQA (GPT-4) and LightPROF (frozen Llama2-7B with a trainable adapter). Table III reports the results. For LightPROF, we report its original Llama2-7B results here, while Table II reports Llama3-8B. Without any finetuning, LLM-KGFR shows strong performance. Under frozen Llama2-7B, it clearly outperforms the frozen and LoRA-finetuned baselines. With GPT-4, it reports high F1 and competitive Hit scores—slightly above ToG and EffiQA, and close to PoG. These results indicate that our gains mainly come from the retrieval–reasoning synergy rather than LLM finetuning or model size.

TABLE III: Comparison under unified LLM settings

Method	WebQSP		CWQ	
	F1	Hit	F1	Hit
<i>Llama2-7B (LoRA / Finetuned)</i>				
KD-CoT (LoRA)	52.5	68.6	—	55.7
G-Retriever (LoRA)	—	73.8	—	—
RoG (Finetuned)	—	85.7	56.2	62.6
GNN-RAG (Finetuned)	71.3	85.7	59.4	66.8
<i>Llama2-7B (Frozen)</i>				
G-Retriever (w/o LoRA)	—	70.5	—	—
LightPROF	—	71.2	—	48.5
<b>LLM-KGFR</b>	<b>55.7</b>	<b>80.8</b>	<b>44.6</b>	<b>58.1</b>
<i>GPT-4 (Frozen)</i>				
ToG	—	82.6	—	69.5
PoG	—	87.3	—	75.0
EffiQA	—	82.9	—	69.5
<b>LLM-KGFR</b>	<b>73.4</b>	<b>89.0</b>	<b>61.9</b>	<b>72.2</b>

### C. RQ2: Generalization Evaluation

We conduct experiments to validate the generalization capabilities of LLM-KGFR.

#### a) Cross-dataset transfer between WebQSP and CWQ:

To explore the generalization and knowledge transfer capabilities of LLM-KGFR, we pre-train two KGFRs using WebQSP and CWQ, respectively. We then test the two models on both datasets, resulting in four knowledge transfer settings. Figure 3 shows the Hit scores w.r.t. various values of  $k$ .

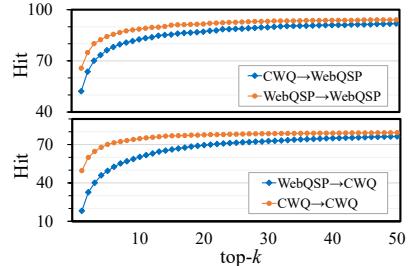


Fig. 3: KGFR only top- $k$  retrieval results between WebQSP and CWQ

We observe a slight performance gap when comparing results from training and testing on the same dataset versus different datasets, and this gap continues to narrow as  $k$  increases. Notably, the gap is smaller in the “CWQ → WebQSP” case compared to “WebQSP → CWQ”. This discrepancy can be attributed to the significantly larger training set for CWQ, which consists of 27,639 samples, compared to only 2,848 samples in WebQSP. Furthermore, we conduct experiments on the complete LLM-KGFR based on these combinations, and the results are shown in Table IV. This gap further narrows after collaborating with an LLM.

TABLE IV: Generalization between WebQSP and CWQ

LLMs	Training → Testing	F1	Hit	H@1
		CWQ → WebQSP	72.9	86.4
Qwen-max	WebQSP → WebQSP	74.7	90.3	83.2
	WebQSP → CWQ	57.6	70.2	62.1
GPT-4o-mini	CWQ → CWQ	61.6	71.8	63.6
	CWQ → WebQSP	68.9	88.6	79.2
	WebQSP → WebQSP	69.0	89.4	80.0
	WebQSP → CWQ	52.3	70.5	61.2
	CWQ → CWQ	53.7	72.3	62.1

*b) Generalization to unseen dataset:* We further validate LLM-KGFR’s generalization on GrailQA [54] (an entirely unseen dataset during pre-training). The evaluation setup and the LLM are the same as PoG [55]. The results are shown in Table V. LLM-KGFR maintains strong knowledge transfer and achieves competitive performance on GrailQA, demonstrating its robust generalization capabilities. These results further confirm that LLM-KGFR can effectively handle novel questions in unseen domains.

TABLE V: Performance on the GrailQA dataset

Methods	Overall	I.I.D	Compositional	Zero-shot
ToG	68.7	70.1	56.1	72.7
PoG	76.5	76.3	62.1	81.7
<b>LLM-KGFR</b>	<b>80.2</b>	<b>79.4</b>	<b>74.8</b>	<b>82.4</b>

c) *Generalization to multiple-choice QA*: We evaluate the multiple-choice QA performance using the CSQA, OBQA, and MedQA datasets. The background KGs for CSQA and OBQA are sourced from ConceptNet, while MedQA uses UMLS and DrugBank. We contrast a direct LLM-answering baseline with LLM-KGFR. For LLM-KGFR, we incorporate the options as known information and ask the LLM to return the most likely answer option. The results are displayed in Figure 4. LLM-KGFR achieves consistent improvements across all LLMs and datasets compared to direct answering.

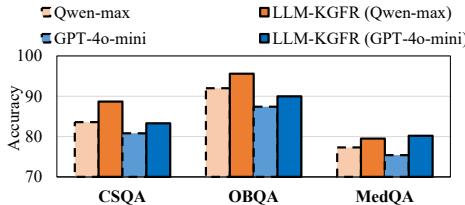


Fig. 4: Acc. of multiple-choice QA

We further evaluate LLM-KGFR against state-of-the-art GNN-based approaches to establish comprehensive benchmarks. Table VI compares the performance on three challenging QA datasets (CSQA, OBQA, and MedQA) using the results from the PaperWithCode leaderboard<sup>6</sup>. The results reveal that LLM-KGFR achieves superior performance across all benchmarks, particularly showing remarkable gains on the biomedical MedQA dataset. This advantage stems from three key factors: (i) The KGFR’s robust knowledge retrieval capabilities that surpass traditional graph propagation methods. (ii) The LLM’s strong language understanding and reasoning capabilities. (iii) Their synergistic interaction that dynamically adapts to different question types. The consistent superiority across both general-domain (CSQA/OBQA) and specialized (MedQA) benchmarks demonstrates LLM-KGFR’s versatility compared to previous GNN-based approaches.

TABLE VI: Accuracy comparison with GNN-based methods across multiple QA datasets

Methods	CSQA	OBQA	MedQA
QA-GNN [61]	76.5	82.8	38.0
DEKCOR [73]	83.3	82.4	–
DRAGON [74]	72.0	76.0	–
GSC [75]	79.1	87.4	–
GrapeQA [76]	74.9	90.0	39.5
GNR [77]	–	89.6	–
LLM-KGFR (GPT-4o-mini)	83.3	90.0	<b>80.2</b>
LLM-KGFR (Qwen-max)	<b>88.6</b>	<b>95.6</b>	79.5

d) *Generalization to commonsense reasoning*.: ExplaGraphs is a dataset for generative commonsense reasoning, evaluating whether arguments are supportive or contradictory. Its background graphs are small enough to fit entirely within a single prompt, making it ideal for testing structured retrieval effects beyond context-length limits. We evaluate four variants: (i) a pure LLM incorporating the background KG into prompts, (ii) our full LLM-KGFR that retrieves relevant

entities and verbalizes corresponding facts into natural sentences, (iii) LLM-KGFR (w/o retrieval), and (iv) LLM-KGFR (w/o verbalization). Although the answers are not entities, we still retrieve and rank relevant entities, and the relation verbalization module converts retrieved facts into coherent natural-language evidence. Results are shown in Table VII. Qwen-max already surpasses existing baselines, while our structured retrieval and fact verbalization further boost performance. Even though the graphs can be fully included in prompts, LLM-KGFR still shows consistent gains, indicating that improvements stem from structured retrieval and semantic verbalization rather than context truncation. Ablation shows that removing verbalization drops accuracy from 94.3 to 93.9, and removing retrieval further to 93.3, confirming that both components are complementary and essential.

TABLE VII: Accuracy on ExplaGraphs

Methods	Acc.
Zero-shot [7]	56.5
Zero-CoT [78]	57.0
CoT-BAG [79]	57.9
KAPING [80]	62.3
GraphToken [81]	85.1
G-Retriever [7]	85.2
G-Retriever (LoRA) [7]	87.1
Qwen-max	92.1
LLM-KGFR (Qwen-max)	<b>94.3</b>
LLM-KGFR (Qwen-max w/o verbalization)	93.9
LLM-KGFR (Qwen-max w/o retrieval)	93.3

#### D. RQ3: Scalability and Efficiency

a) *Scalability Analysis*: We propose two strategies to ensure scalability: progressive expansion (PE) and asymmetric pruning (AP) in Section IV-C. Here, we conduct experiments to analyze their effects on the message propagation range and to verify whether they can maintain accuracy. Specifically, we design five variants based on whether PE and AP are enabled, as well as the threshold  $\lambda$ . For each variant, we calculate the average number of entities and facts (facts are treated as directed edges) involved in propagation for each question. We then evaluate their H@1 scores under the condition without LLM collaboration (w/o LLM). The experimental results are shown in Table VIII. “OOM” denotes Out-of-Memory.

TABLE VIII: Effects of Progressive Propagation (PE) and Asymmetric Pruning (AP) under different thresholds  $\lambda$

Dataset	PE	AP	$\lambda$	Entity	Fact	H@1 (w/o LLM)
WebQSP	×	×	-	1.3m	7.6m	OOM
	✓	✗	-	0.6m	3.4m	OOM
	✓	✓	1000	0.2m	1.5m	65.4
	✓	✓	100	0.1m	0.6m	65.7
CWQ	✓	✗	10	9.8k	44.8k	60.2
	✗	✗	-	2.3m	14.5m	OOM
	✓	✗	-	0.8m	5.1m	OOM
	✓	✓	1,000	0.3m	1.6m	49.8
	✓	✓	100	0.1m	0.6m	49.7
	✓	✓	10	7.9k	38.8k	38.4

<sup>6</sup><https://paperswithcode.com/>

Without PE or AP, propagation expands to millions of nodes and edges, easily causing OOM errors. PE alone halves the size by expanding neighborhoods progressively. With both PE and AP, the graph shrinks by 80–90%, as AP prunes redundant high-degree nodes while keeping key links. Smaller  $\lambda$  increases pruning but risks missing reasoning paths; reducing it from 1000 to 10 cuts 99% of the range with accuracy loss. A moderate  $\lambda=100$  balances accuracy and efficiency, maintaining near-optimal H@1 within 10% of full-scale cost.

*b) Efficiency Analysis:* We conduct experiments to analyze the inference efficiency of LLM-KGFR. We compare the LLM calls, token usage and inference time with state-of-the-art black-box models, while excluding LLM-tuning-based approaches that require costly LLM finetuning for each dataset. Our message-passing component completes most retrievals within 0.5 second, while our LLM generation phase typically requires only 2–6 total operations per question (combining generation and reflection steps). As shown in Table IX, LLM-KGFR substantially outperforms existing methods across all efficiency metrics. These efficiency improvements are most evident when handling complex queries, where our KGFR retrieval mechanism eliminates the need for LLM calls required by LLM-based retrieval.

TABLE IX: Efficiency comparison with baseline methods

Datasets	Methods	LLM calls	Tokens	Time (s)
CWQ	ToG	22.6	9,669.4	96.5
	PoG	13.3	8,156.2	23.3
	LLM-KGFR	2.4	3,145.1	9.2
WebQSP	ToG	15.9	7,018.9	63.1
	PoG	9.0	5,517.7	16.8
	LLM-KGFR	2.1	2,725.2	7.6

#### E. RQ4: Further Analysis

*a) Ablation Study:* We conduct an ablation study to evaluate the contribution of each module. Specifically, six variants of LLM-KGFR (Qwen-max) are constructed by selectively removing key components: “w/o LLM ( $k=10$ )” removes the LLM and directly ranks the top-10 entities; “w/o description” replaces generated relation descriptions with raw relation names; “w/o node-/edge-/path-level retrieval” respectively disable different hierarchical retrieval interfaces; and “w/o reflection” removes the reflection loop. Table X shows that the complete LLM-KGFR achieves the highest F1 and H@1 scores across both datasets, demonstrating that all modules positively contribute to overall performance. Removing the LLM causes a dramatic F1 drop, confirming that language-guided reasoning and answer synthesis are indispensable. The “w/o description” variant performs close to the full model since raw relation names still carry partial semantic cues from pre-training. Among retrieval interfaces, the node-level retrieval yields the largest performance degradation when removed, validating it as the core of KGFR’s reasoning process. Edge- and path-level retrievals also bring steady gains by enriching multi-hop context and refining fact-level evidence. Finally, removing the reflection module slightly reduces accuracy, since the pre-retrieval stage already offers strong candidate and fact

grounding, though reflection further enhances stability and prevents premature termination in complex cases.

TABLE X: Ablation results

Methods	WebQSP			CWQ		
	F1	Hit	H@1	F1	Hit	H@1
LLM-KGFR (Qwen-max)	74.7	90.3	83.2	61.6	71.8	63.6
w/o LLM ( $k = 10$ )	29.5	88.5	65.7	18.0	74.7	49.7
w/o description	71.6	85.9	81.2	61.2	69.6	62.9
w/o node-level retrieval	62.7	73.8	68.6	51.3	56.4	54.0
w/o edge-level retrieval	69.3	87.3	81.4	58.9	70.1	62.4
w/o path-level retrieval	66.2	86.1	80.2	57.3	69.8	62.2
w/o reflection	71.9	89.5	81.4	59.7	69.9	62.0

*b) Effect of encoder choice:* To further analyze the influence of the textual encoder used in relation and question representation, we fix the LLM to Qwen-max and replace the default BGE-Large-EN-v1.5 with several alternative BERT-based encoders of different sizes and training objectives, including BERT-base/large, SentenceBERT-base/large, and BGE-base/large. Table XI reports the results on WebQSP and CWQ.

TABLE XI: Performance comparison of LLM-KGFR with different BERT encoders.

Encoder	WebQSP			CWQ		
	F1	Hit	H@1	F1	Hit	H@1
BERT-base	70.8	84.0	80.2	58.5	67.9	59.6
BERT-large	72.6	89.7	82.6	60.7	71.0	62.2
SentenceBERT-base	71.4	85.3	80.3	59.4	68.1	60.3
SentenceBERT-large	74.1	88.4	82.4	61.2	71.4	62.4
BGE-base	70.4	85.6	79.9	57.6	68.9	60.0
BGE-large	<b>74.7</b>	<b>90.3</b>	<b>83.2</b>	<b>61.6</b>	<b>71.8</b>	<b>63.6</b>

We observe that encoder capacity plays a key role in retrieval quality. Larger encoders (e.g., BERT-large and BGE-large) consistently outperform their base counterparts across both datasets, indicating that a stronger language encoder yields more informative question and relation representations. Sentence-level contrastive training (as in SentenceBERT and BGE) also provides moderate gains over vanilla BERT, suggesting that semantic alignment between question and relation text further enhances KGFR’s reasoning accuracy. Overall, these results confirm that the encoder’s representation quality directly impacts the generalization ability and overall performance of the retriever.

*c) Effect of LLM scale and architecture:* In the main experiments, we primarily adopt commercial black-box LLMs (e.g., GPT-4, GPT-4-turbo, Qwen-max). To further investigate how the scale and architecture of the LLM affect overall performance, we replace these models with open-source LLMs of different sizes, including Llama3-8B, Llama3-70B, Qwen2.5-7B, and Qwen2.5-72B. Table XII reports the results on WebQSP and CWQ. The results show a consistent trend: larger LLMs yield stronger accuracy and answer consistency, while smaller ones remain competitive. Specifically, the 70B and 72B models achieve improvements of about 6–8 points in F1 over their 7B and 8B counterparts, demonstrating that LLM-KGFR can effectively exploit richer linguistic representations from larger models. Meanwhile, the solid performance

of the smaller models confirms that our retriever–reasoner collaboration remains effective even with lightweight LLMs, highlighting LLM-KGFR’s scalability and robustness across diverse LLM backbones.

TABLE XII: Performance comparison of LLM-KGFR with open-source LLMs of different scales.

LLM Backbone	WebQSP			CWQ		
	F1	Hit	H@1	F1	Hit	H@1
LLM-KGFR (Llama3-8B)	66.8	82.4	75.7	47.7	61.9	52.5
LLM-KGFR (Llama3-70B)	73.1	90.1	82.0	52.5	67.8	58.1
LLM-KGFR (Qwen2.5-7B)	68.4	83.3	76.5	43.6	57.2	48.8
LLM-KGFR (Qwen2.5-72B)	<b>72.9</b>	<b>89.8</b>	<b>82.1</b>	<b>55.4</b>	<b>66.1</b>	<b>57.7</b>

#### d) Incorporating LLM-based Retrieval Augmentation:

We introduce KGFR-based retrieval methods in Section V. They can collaborate with LLM-based retrievers to further enhance reasoning. Here, we follow [8] to create some variants of LLM-KGFR that incorporate finetunable LLM-based retrieval [10]. Specifically, during the pre-retrieval, we integrate the results from the LLM-based retrieval into the prompt to strengthen reasoning, with the results shown in Table XIII. We observe further performance improvements, which indicate that the information from the KGFR and LLM retrieval is complementary. It also demonstrates that LLM-KGFR can integrate LLM-based retrieval for further enhancement.

TABLE XIII: Results of LLM-KGFR with LLM-based retrieval augmentation (RA)

Methods	WebQSP			CWQ		
	F1	Hit	H@1	F1	Hit	H@1
GNN-RAG	71.3	85.7	80.6	59.4	66.8	61.7
GNN-RAG + RA	73.5	90.7	82.8	60.4	68.7	62.8
LLM-KGFR (Qwen-max)	74.7	90.3	83.2	61.6	71.8	63.6
LLM-KGFR (Qwen-max) + RA	76.3	89.9	83.8	62.9	72.7	64.2
LLM-KGFR (GPT-4o-mini)	69.0	89.4	80.0	53.7	72.3	62.1
LLM-KGFR (GPT-4o-mini) + RA	72.4	92.0	83.1	54.6	72.2	62.4

## VII. CONCLUSIONS

We present LLM-KGFR, a collaborative framework that unifies LLMs with a KG retriever for generalized and scalable KGQA. Through language-guided initialization and asymmetric progressive propagation, KGFR achieves efficient retrieval and strong cross-KG generalization without finetuning. With multi-level retrieval and reflection-based reasoning, the framework enables controllable, interpretable question answering. Extensive experiments on seven benchmarks verify that LLM-KGFR consistently outperforms existing methods in accuracy, efficiency, and transferability. Future work will extend this paradigm to broader structured and unstructured knowledge sources for more comprehensive reasoning.

## REFERENCES

- [1] L. Huang, W. Yu, W. Ma, W. Zhong, Z. Feng, H. Wang, Q. Chen, W. Peng, X. Feng, B. Qin, and T. Liu, “A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions,” *CoRR*, vol. abs/2311.05232, 2023.
- [2] Y. Chen, H. Li, G. Qi, T. Wu, and T. Wang, “Outlining and filling: Hierarchical query graph generation for answering complex questions over knowledge graphs,” *IEEE Trans. Knowl. Data Eng.*, vol. 35, no. 8, pp. 8343–8357, 2023.
- [3] S. Hu, L. Zou, J. X. Yu, H. Wang, and D. Zhao, “Answering natural language questions by subgraph matching over knowledge graphs,” *IEEE Trans. Knowl. Data Eng.*, vol. 30, no. 5, pp. 824–837, 2018.
- [4] X. Wang, F. Luo, Q. Wu, and Z. Bao, “How context or knowledge can benefit healthcare question answering?” *IEEE Trans. Knowl. Data Eng.*, vol. 35, no. 1, pp. 575–588, 2023.
- [5] Y. Lan, G. He, J. Jiang, J. Jiang, W. X. Zhao, and J. Wen, “Complex knowledge base question answering: A survey,” *IEEE Trans. Knowl. Data Eng.*, vol. 35, no. 11, pp. 11 196–11 215, 2023.
- [6] S. Pan, L. Luo, Y. Wang, C. Chen, J. Wang, and X. Wu, “Unifying large language models and knowledge graphs: A roadmap,” *IEEE Trans. Knowl. Data Eng.*, vol. 36, no. 7, pp. 3580–3599, 2024.
- [7] X. He, Y. Tian, Y. Sun, N. V. Chawla, T. Laurent, Y. LeCun, X. Bresson, and B. Hooi, “G-Retriever: Retrieval-augmented generation for textual graph understanding and question answering,” *CoRR*, vol. abs/2402.07630, 2024.
- [8] C. Mavromatis and G. Karypis, “GNN-RAG: Graph neural retrieval for large language model reasoning,” *CoRR*, vol. abs/2405.20139, 2024.
- [9] J. Sun, C. Xu, L. Tang, S. Wang, C. Lin, Y. Gong, H. Shum, and J. Guo, “Think-on-Graph: Deep and responsible reasoning of large language model with knowledge graph,” *CoRR*, vol. abs/2307.07697, 2023.
- [10] L. Luo, Y. Li, G. Haffari, and S. Pan, “Reasoning on Graphs: Faithful and interpretable large language model reasoning,” in *ICLR*, 2024.
- [11] D. Edge, H. Trinh, N. Cheng, J. Bradley, A. Chao, A. Mody, S. Truitt, and J. Larson, “From local to global: A graph RAG approach to query-focused summarization,” *CoRR*, vol. abs/2404.16130, 2024.
- [12] Z. Guo, L. Xia, Y. Yu, T. Ao, and C. Huang, “LightRAG: Simple and fast retrieval-augmented generation,” *CoRR*, vol. abs/2410.05779, 2024.
- [13] B. J. Gutierrez, Y. Shu, Y. Gu, M. Yasunaga, and Y. Su, “HippoRAG: Neurobiologically inspired long-term memory for large language models,” in *NeurIPS*, 2024.
- [14] L. Luo, Z. Zhao, G. Haffari, D. Phung, C. Gong, and S. Pan, “GFM-RAG: Graph foundation model for retrieval augmented generation,” *CoRR*, vol. abs/2502.01113, 2025.
- [15] K. Wang, F. Duan, S. Wang, P. Li, Y. Xian, C. Yin, W. Rong, and Z. Xiong, “Knowledge-driven CoT: Exploring faithful reasoning in LLMs for knowledge-intensive question answering,” *CoRR*, vol. abs/2308.13259, 2023.
- [16] J. Jiang, K. Zhou, Z. Dong, K. Ye, X. Zhao, and J. Wen, “StructGPT: A general framework for large language model to reason over structured data,” in *EMNLP*, 2023, pp. 9237–9251.
- [17] Z. Dong, B. Peng, Y. Wang, J. Fu, X. Wang, X. Zhou, Y. Shan, K. Zhu, and W. Chen, “EffiQA: Efficient question-answering with strategic multi-model collaboration on knowledge graphs,” in *COLING*, 2025, pp. 7180–7194.
- [18] T. Ao, Y. Yu, Y. Wang, Y. Deng, Z. Guo, L. Pang, P. Wang, T. Chua, X. Zhang, and Z. Cai, “Lightprof: A lightweight reasoning framework for large language model on knowledge graph,” in *AAAI*, 2025, pp. 23 424–23 432.
- [19] L. Luo, Z. Zhao, C. Gong, G. Haffari, and S. Pan, “Graph-constrained Reasoning: Faithful reasoning on knowledge graphs with large language models,” *CoRR*, vol. abs/2410.13080, 2024.
- [20] P. Velickovic, W. Fedus, W. L. Hamilton, P. Liò, Y. Bengio, and R. D. Hjelm, “Deep graph infomax,” in *ICLR*, Addis Ababa, Ethiopia, 2019.
- [21] W. Hu, B. Liu, J. Gomes, M. Zitnik, P. Liang, V. S. Pande, and J. Leskovec, “Strategies for pre-training graph neural networks,” in *ICLR*, Addis Ababa, Ethiopia, 2020.
- [22] Y. Rong, Y. Bian, T. Xu, W. Xie, Y. Wei, W. Huang, and J. Huang, “Self-supervised graph transformer on large-scale molecular data,” in *NeurIPS*, Virtual, 2020.
- [23] F. Sun, J. Hoffmann, V. Verma, and J. Tang, “Infograph: Unsupervised and semi-supervised graph-level representation learning via mutual information maximization,” in *ICLR*, Addis Ababa, Ethiopia, 2020.
- [24] Z. Hou, X. Liu, Y. Cen, Y. Dong, H. Yang, C. Wang, and J. Tang, “GraphMAE: Self-supervised masked graph autoencoders,” *CoRR*, vol. abs/2205.10803, pp. 1–11, 2022.
- [25] M. Sun, K. Zhou, X. He, Y. Wang, and X. Wang, “GPPT: Graph pre-training and prompt tuning to generalize graph neural networks,” in *KDD*, Washington, DC, USA, 2022, pp. 1717–1727.
- [26] X. Sun, H. Cheng, J. Li, B. Liu, and J. Guan, “All in One: Multi-task prompting for graph neural networks,” in *KDD*, Long Beach, CA, USA, 2023, pp. 2120–2131.

- [27] Z. Liu, X. Yu, Y. Fang, and X. Zhang, “GraphPrompt: Unifying pre-training and downstream tasks for graph neural networks,” in *WWW*, Austin, TX, USA, 2023, pp. 417–428.
- [28] C. Gong, X. Li, J. Yu, C. Yao, J. Tan, C. Yu, and D. Yin, “Prompt tuning for multi-view graph contrastive learning,” *CoRR*, vol. abs/2310.10362, 2023.
- [29] Y. Zhu, J. Guo, and S. Tang, “SGL-PT: A strong graph learner with graph prompt tuning,” *CoRR*, vol. abs/2302.12449, 2023.
- [30] R. Shirikavand and H. Huang, “Deep prompt tuning for graph transformers,” *CoRR*, vol. abs/2309.10131, 2023.
- [31] Y. Ma, N. Yan, J. Li, M. S. Mortazavi, and N. V. Chawla, “HetGPT: Harnessing the power of prompt tuning in pre-trained heterogeneous graph neural networks,” *CoRR*, vol. abs/2310.15318, 2023.
- [32] X. Sun, J. Zhang, X. Wu, H. Cheng, Y. Xiong, and J. Li, “Graph prompt learning: A comprehensive survey and beyond,” *CoRR*, vol. abs/2311.16534, 2023.
- [33] Z. Sun, J. Huang, J. Lin, X. Xu, Q. Chen, and W. Hu, “Joint pre-training and local re-training: Transferable representation learning on multi-source knowledge graphs,” in *KDD*, 2023, pp. 2132–2144.
- [34] M. Galkin, X. Yuan, H. Mostafa, J. Tang, and Z. Zhu, “Towards foundation models for knowledge graph reasoning,” in *ICLR*, Vienna, Austria, 2024.
- [35] Y. Cui, Z. Sun, and W. Hu, “A prompt-based knowledge graph foundation model for universal in-context reasoning,” in *NeurIPS*, 2024.
- [36] Y. Zhang, B. Bevilacqua, M. Galkin, and B. Ribeiro, “TRIX: A more expressive model for zero-shot domain transfer in knowledge graphs,” *CoRR*, vol. abs/2502.19512, 2025.
- [37] A. Bordes, N. Usunier, A. García-Durán, J. Weston, and O. Yakhnenko, “Translating embeddings for modeling multi-relational data,” in *NeurIPS*, 2013, pp. 2787–2795.
- [38] A. Saxena, A. Tripathi, and P. P. Talukdar, “Improving multi-hop question answering over knowledge graphs using knowledge base embeddings,” in *ACL*, 2020, pp. 4498–4507.
- [39] Z. Zhu, Z. Zhang, L. A. C. Xhonneux, and J. Tang, “Neural bellman-ford networks: A general graph neural network framework for link prediction,” in *NeurIPS*, 2021, pp. 29476–29490.
- [40] Y. Zhang and Q. Yao, “Knowledge graph reasoning with relational digraph,” in *WWW*, 2022, pp. 912–924.
- [41] J. Devlin, M. Chang, K. Lee, and K. Toutanova, “BERT: Pre-training of deep bidirectional transformers for language understanding,” in *NAACL*, Stroudsburg, PA, USA, 2019, pp. 4171–4186.
- [42] T. Lacroix, N. Usunier, and G. Obozinski, “Canonical tensor decomposition for knowledge base completion,” in *ICML*, ser. Proceedings of Machine Learning Research, vol. 80, 2018, pp. 2869–2878.
- [43] A. Elgohary, D. Peskov, and J. L. Boyd-Graber, “Can you unpack that? Learning to rewrite questions-in-context,” in *EMNLP*, 2019, pp. 5917–5923.
- [44] S. Min, J. Michael, H. Hajishirzi, and L. Zettlemoyer, “AmbigQA: Answering ambiguous open-domain questions,” in *EMNLP*, 2020, pp. 5783–5797.
- [45] W. Yih, M. Chang, X. He, and J. Gao, “Semantic parsing via staged query graph generation: Question answering with knowledge base,” in *ACL*, 2015, pp. 1321–1331.
- [46] A. Talmor and J. Berant, “The web as a knowledge-base for answering complex questions,” in *NAACL*, 2018, pp. 641–651.
- [47] J. Jiang, K. Zhou, X. Zhao, and J. Wen, “UniKGQA: Unified retrieval and reasoning for solving multi-hop question answering over knowledge graph,” in *ICLR*, 2023.
- [48] H. Sun, T. Bedrax-Weiss, and W. W. Cohen, “PullNet: Open domain question answering with iterative retrieval on knowledge bases and text,” in *EMNLP*, 2019, pp. 2380–2390.
- [49] G. He, Y. Lan, J. Jiang, W. X. Zhao, and J. Wen, “Improving multi-hop knowledge base question answering by learning intermediate supervision signals,” in *WSDM*, 2021, pp. 553–561.
- [50] H. Sun, B. Dhingra, M. Zaheer, K. Mazaitis, R. Salakhutdinov, and W. W. Cohen, “Open domain question answering using early fusion of knowledge bases and text,” in *EMNLP*, 2018, pp. 4231–4242.
- [51] C. Ge, X. Liu, L. Chen, B. Zheng, and Y. Gao, “LargeEA: Aligning entities for large-scale knowledge graphs,” *VLDB*, vol. 15, no. 2, pp. 237–245, 2021.
- [52] R. Wang, Y. Yan, J. Wang, Y. Jia, Y. Zhang, W. Zhang, and X. Wang, “AceKG: A large-scale knowledge graph for academic data mining,” in *CIKM*, 2018, pp. 1487–1490.
- [53] D. Dessì, F. Osborne, D. R. Recupero, D. Buscaldi, and E. Motta, “CS-KG: A large-scale knowledge graph of research entities and claims in computer science,” in *ISWC*, ser. Lecture Notes in Computer Science, vol. 13489, 2022, pp. 678–696.
- [54] Y. Gu, S. Kase, M. Vanni, B. M. Sadler, P. Liang, X. Yan, and Y. Su, “Beyond I.I.D.: three levels of generalization for question answering on knowledge bases,” in *WWW*, 2021, pp. 3477–3488.
- [55] L. Chen, P. Tong, Z. Jin, Y. Sun, J. Ye, and H. Xiong, “Plan-on-Graph: Self-correcting adaptive planning of large language model on knowledge graphs,” in *NeurIPS*, 2024.
- [56] S. Saha, P. Yadav, L. Bauer, and M. Bansal, “ExplaGraphs: An explanation graph generation task for structured commonsense reasoning,” in *EMNLP*, 2021, pp. 7716–7740.
- [57] A. Talmor, J. Herzig, N. Lourie, and J. Berant, “CommonsenseQA: A question answering challenge targeting commonsense knowledge,” in *NAACL*, 2019, pp. 4149–4158.
- [58] T. Mihaylov, P. Clark, T. Khot, and A. Sabharwal, “Can a suit of armor conduct electricity? A new dataset for open book question answering,” in *EMNLP*, 2018, pp. 2381–2391.
- [59] R. Speer, J. Chin, and C. Havasi, “ConceptNet 5.5: An open multilingual graph of general knowledge,” in *AAAI*, 2017, pp. 4444–4451.
- [60] B. Y. Lin, X. Chen, J. Chen, and X. Ren, “KagNet: Knowledge-aware graph networks for commonsense reasoning,” in *EMNLP*, 2019, pp. 2829–2839.
- [61] M. Yasunaga, H. Ren, A. Bosselut, P. Liang, and J. Leskovec, “QA-GNN: Reasoning with language models and knowledge graphs for question answering,” in *NAACL*, 2021, pp. 535–546.
- [62] D. Jin, E. Pan, N. Oufatolle, W. Weng, H. Fang, and P. Szolovits, “What disease does this patient have? A large-scale open domain question answering dataset from medical exams,” *CoRR*, vol. abs/2009.13081, 2020.
- [63] O. Bodenreider, “The unified medical language system (UMLS): Integrating biomedical terminology,” *Nucleic Acids Res.*, vol. 32, no. Database-Issue, pp. 267–270, 2004.
- [64] D. S. Wishart, Y. D. Feunang, A. C. Guo, E. J. Lo, A. Marcu, J. R. Grant, T. Sajed, D. Johnson, C. Li, Z. Sayeeda, N. Assempour, I. Iynkkaran, Y. Liu, A. Maciejewski, N. Gale, A. Wilson, L. Chin, R. Cummings, D. Le, A. Pon, C. Knox, and M. Wilson, “DrugBank 5.0: A major update to the DrugBank database for 2018,” *Nucleic Acids Res.*, vol. 46, no. Database-Issue, pp. D1074–D1082, 2018.
- [65] A. H. Miller, A. Fisch, J. Dodge, A. Karimi, A. Bordes, and J. Weston, “Key-value memory networks for directly reading documents,” in *EMNLP*, 2016, pp. 1400–1409.
- [66] J. Shi, S. Cao, L. Hou, J. Li, and H. Zhang, “TransferNet: An effective and transparent framework for multi-hop question answering over relation graph,” in *EMNLP*, 2021, pp. 4149–4158.
- [67] P. Sen, A. Oliya, and A. Saffari, “Expanding end-to-end question answering on differentiable knowledge graphs with intersection,” in *EMNLP*, 2021, pp. 8805–8812.
- [68] J. Zhang, X. Zhang, J. Yu, J. Tang, J. Tang, C. Li, and H. Chen, “Subgraph retrieval enhanced model for multi-hop knowledge base question answering,” in *ACL*, 2022, pp. 5773–5784.
- [69] G. He, Y. Lan, J. Jiang, W. X. Zhao, and J. Wen, “Improving multi-hop knowledge base question answering by learning intermediate supervision signals,” in *WSDM*, 2021, pp. 553–561.
- [70] M. Atzeni, J. Bogojeska, and A. Loukas, “SQALER: Scaling question answering by decoupling multi-hop and logical reasoning,” in *NeurIPS*, 2021, pp. 12587–12599.
- [71] X. Ye, L. Xiao, C. Zhang, and T. Yamasaki, “E-ReaRev: Adaptive reasoning for question answering over incomplete knowledge graphs by edge and meaning extensions,” in *NLDB*, ser. Lecture Notes in Computer Science, vol. 14763, 2024, pp. 85–95.
- [72] T. Li, X. Ma, A. Zhuang, Y. Gu, Y. Su, and W. Chen, “Few-shot in-context learning on knowledge base question answering,” in *ACL*, 2023, pp. 6966–6980.
- [73] Y. Xu, C. Zhu, R. Xu, Y. Liu, M. Zeng, and X. Huang, “Fusing context into knowledge graph for commonsense question answering,” in *ACL*, ser. Findings of ACL, vol. ACL/IJCNLP 2021, 2021, pp. 1201–1207.
- [74] M. Yasunaga, A. Bosselut, H. Ren, X. Zhang, C. D. Manning, P. Liang, and J. Leskovec, “Deep bidirectional language-knowledge graph pre-training,” in *NeurIPS*, 2022.
- [75] K. Wang, Y. Zhang, D. Yang, L. Song, and T. Qin, “GNN is A Counter? Revisiting GNN for question answering,” in *ICLR*, 2022.
- [76] D. Taunk, L. Khanna, S. V. P. K. Kandru, V. Varma, C. Sharma, and M. Tapaswi, “GrapeQA: Graph augmentation and pruning to enhance question-answering,” in *WWW*, 2023, pp. 1138–1144.
- [77] Y. Tian, H. Song, Z. Wang, H. Wang, Z. Hu, F. Wang, N. V. Chawla, and P. Xu, “Graph neural prompting with large language models,” in *AAAI*, 2024, pp. 19080–19088.
- [78] T. Kojima, S. S. Gu, M. Reid, Y. Matsuo, and Y. Iwasawa, “Large language models are zero-shot reasoners,” in *NeurIPS*, 2022.

- [79] H. Wang, S. Feng, T. He, Z. Tan, X. Han, and Y. Tsvetkov, “Can language models solve graph problems in natural language?” in *NeurIPS*, 2023.
- [80] J. Baek, A. F. Aji, and A. Saffari, “Knowledge-augmented language model prompting for zero-shot knowledge graph question answering,” *CoRR*, vol. abs/2306.04136, 2023.
- [81] B. Perozzi, B. Fatemi, D. Zelle, A. Tsitsulin, S. M. Kazemi, R. Al-Rfou, and J. Halcrow, “Let your graph do the talking: Encoding structured data for LLMs,” *CoRR*, vol. abs/2402.05862, 2024.



**Yuanning Cui** received the BS and MS degrees in Computer Science and Technology from China University of Mining and Technology in 2018 and Nanjing University of Aeronautics and Astronautics in 2021, respectively, and the PhD degree from Nanjing University in 2025. He is currently a Lecturer with the School of Computer, Nanjing University of Information Science and Technology, China. His research interests include knowledge graphs, foundation models, and question answering.



**Zequn Sun** is currently a postdoctoral researcher at Nanjing University, China. He received his BS degree in Computer Science and Technology from Hohai University, China, in 2016, and PhD degree in Computer Science and Technology from Nanjing University, China, in 2023. His research interests include knowledge graph, representation learning, and entity alignment.



**Wei Hu** is a full professor at the State Key Laboratory for Novel Software Technology and the National Institute of Healthcare Data Science, Nanjing University, China. He received his PhD degree in Computer Software and Theory in 2009, and BS degree in Computer Science and Technology in 2005, both from Southeast University, China. His main research interests include knowledge graph, database, and intelligent software.



**Zhangjie Fu** (Member, IEEE) received the Ph.D. degree in computer science from the College of Computer, Hunan University, China, in 2012. He is currently a Professor with the School of Computer, Nanjing University of Information Science and Technology, China. His research interests include cloud and outsourcing security, digital forensics, networks, and information security. His research has been supported by NSFC, PAPD, and GYHY. He is a member of ACM.