GRIL: Knowledge Graph Retrieval-Integrated Learning with Large Language Models

Jialin Chen[1], **Houyu Zhang**[2], **Seongjun Yun**[2], **Alejandro Mottini**[2], **Rex Ying**[1], **Xiang Song**[2], **Vassilis N. Ioannidis**[2], **Zheng Li**[2], **Qingjun Cui**[2]
Yale University[1] Amazon[2]

Abstract

Retrieval-Augmented Generation (RAG) has significantly mitigated the hallucinations of Large Language Models (LLMs) by grounding the generation with external knowledge. Recent extensions of RAG to graph-based retrieval offer a promising direction, leveraging the structural knowledge for multi-hop reasoning. However, existing graph RAG typically decouples retrieval and reasoning processes, which prevents the retriever from adapting to the reasoning needs of the LLM. They also struggle with scalability when performing multi-hop expansion over large-scale graphs, or depend heavily on annotated ground-truth entities, which are often unavailable in open-domain settings. To address these challenges, we propose a novel graph retriever trained end-to-end with LLM, which features an attention-based growing and pruning mechanism, adaptively navigating multi-hop relevant entities while filtering out noise. Within the extracted subgraph, structural knowledge and semantic features are encoded via soft tokens and the verbalized graph, respectively, which are infused into the LLM together, thereby enhancing its reasoning capability and facilitating interactive joint training of the graph retriever and the LLM reasoner. Experimental results across three QA benchmarks show that our approach consistently achieves state-of-the-art performance, validating the strength of joint graph–LLM optimization for complex reasoning tasks. Notably, our framework eliminates the need for predefined ground-truth entities by directly optimizing the retriever using LLM logits as implicit feedback, making it especially effective in open-domain settings.

1 Introduction

Large Language Models (LLMs) have shown remarkable abilities in natural language processing tasks (Brown, 2020; Dubey et al., 2024; Achiam et al., 2023). Despite their success, LLMs often suffer from hallucinations, generating outputs that may be factually incorrect, particularly in scenarios requiring domain-intensive knowledge. To mitigate hallucinations and improve domain-specific performance, recent approaches have explored the Retrieval-Augmented Generation (RAG) framework, which enhances LLMs by retrieving external knowledge (Gao et al., 2023; Lewis et al., 2020; Wu et al., 2023; Fan et al., 2024) and has proven especially beneficial in tasks such as Knowledge Graph Question Answering (KGQA) (Bao et al., 2016; Huang et al., 2019; Zheng et al., 2017).

Unlike traditional knowledge bases such as documents and textbooks, knowledge graphs (KGs) provide cleaner and well-structured relational knowledge, offering a more precise and efficient base for navigating complex reasoning paths and reducing hallucinations compared to unstructured data. Consequently, recent efforts have extended RAG by incorporating graph retrieval, where knowledge graphs—structured and domain-specific knowledge bases—are used to guide retrieval and reasoning. Existing approaches use LLMs as a retriever for KGQA tasks (abbreviated as *LLM-as-Retriever*) (Luo et al., 2023; Sun et al., 2023; Wang et al., 2023), extracting relevant facts or relation paths from the KGs based on LLMs' internal knowledge. However, such *LLM-as-retriever* approaches are typically ill-equipped to fully exploit the intricate structures within KGs, such as multi-hop logical relations, and require multiple calls to process different parts of the graph, which introduces scalability issues when dealing with large-scale KGs. There have also been attempts using a graph neural network (GNN) during retrieval (abbreviated as *GNN-as-Retriever*) (He et al., 2024; Hu et al., 2024; Mavromatis and Karypis, 2024). However, one limitation is that most existing approaches train the retrieval and reasoning components separately, resulting in a disjoint optimization process where the retriever focuses solely on

relevant facts without aligning with the reasoning needs of the LLM. Moreover, many rely on ground truth retrieval paths for retriever training, limiting their applicability to open-domain scenarios where the ground truth entities are not available.

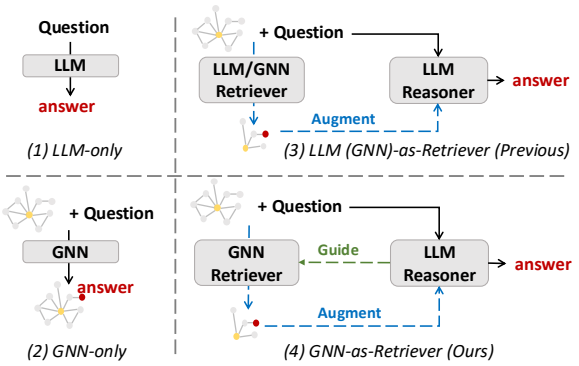

Figure 1: The landscape of existing methods. *(1) LLM-only* and *(2) GNN-only* approaches use a single LLM or GNN to predict the answer. *(3) LLM (GNN)-as-Retriever* approaches rely on the RAG framework to reduce the hallucination and improve the accuracy of LLMs' output. Different from previous approaches, we utilize the LLM reasoner to supervise the GNN retriever, improving the retrieval quality and reasoning accuracy.

To address these limitations, we propose **GRIL**, a novel framework that enables **G**raph **R**etrieval-**I**ntegrated **L**earning with LLM in an end-to-end manner. As illustrated in Figure 1, our approach extends beyond conventional retriever-augmented reasoning by introducing a reverse feedback loop from the LLM to guide the retriever. This LLM feedback is essential for the model's generalizability in open-domain scenarios, by providing a complementary supervision signal, thus eliminating its reliance on answer entities during retriever training. Intuitively, the reverse feedback shifts retrieval from mere relevance to actual usefulness for LLM reasoning. Moreover, we introduce a novel graph retriever that iteratively grows and prunes the extracted knowledge subgraph, which allows the retrieval process to focus on the most relevant multi-hop entities while filtering out irrelevant information, improving retrieval efficiency and accuracy. The proposed integrated training ensures the graph retriever and LLM reasoner are tightly coupled and jointly optimized, fostering better synergy between retrieval and reasoning and improving overall performance.

Our contributions are threefold. (1) We introduce a novel framework that combines an adaptive attention-based graph retriever with joint training alongside the LLM reasoner, improving the accuracy and relevance of knowledge retrieval. (2) Experiments demonstrate performance improvement over competitive models, achieving state-of-the-art results on KGQA benchmarks, and showing strong open-domain generalizability, where most baselines fail. (3) We address the inefficiency of multiple LLM calls during the retrieval process and enable small BERT-level language models to match or even outperform 7B LLMs when paired with our graph retriever, improving inference efficiency and reducing deployment costs significantly.

2 Related Work

Knowledge Graph Question Answering aims to answer natural language queries based on a structured knowledge graph (KG), which consists of entities and their relationships. (Sun et al., 2019; Li et al., 2023; Pan et al., 2024; Yani and Krisnadhi, 2021; Yasunaga et al., 2021; Reinanda et al., 2020). The main challenge in KGQA is handling complex reasoning and mapping it accurately to relevant subgraphs in the KG. Traditional approaches to KGQA often utilize Graph Neural Networks (GNNs) to learn embeddings for entities by aggregating information from their neighbors, supervised by labels that indicate whether a node is an answer for a given question (Yasunaga et al., 2021; Mavromatis and Karypis, 2022; He et al., 2021a). These methods typically lack the ability to effectively traverse long, complex paths and multi-hop reasoning, leading to limited expressivity in capturing deeper structural information. Moreover, they cannot generalize well to open-domain settings where the ground-truth answers are not covered by the KGs. Our method addresses this issue through a dynamic, attention-based graph retriever that iteratively grows and prunes the subgraph, enabling better exploration of multi-hop relationships while eliminating the dependency on ground truth entities during training of the graph retriever.

Retrieval-augmented LLM Reasoning. To address the hallucination issue within the LLMs' output, recent works have explored a retrieval-augmented generation (RAG) framework, where a retriever extracts relevant information from an external knowledge base (*e.g.,* text corpus, a KG, or other structured resources) and converts it into textual prompts for LLMs (Gao et al., 2023; Li et al., 2024). Compared with traditional RAG systems applied to document-based knowledge bases (Robertson et al., 2009; Karpukhin et al., 2020), RAG

on knowledge graphs (KGs) provide cleaner and more well-structured relations with less ambiguity for LLM reasoning. Some works focus on leveraging LLMs as retrievers to extract relevant facts or relational paths from a graph (Wu et al., 2023; Sun et al., 2023; Luo et al., 2023; Yu et al., 2022), which are then used for reasoning. However, these approaches often require multiple LLM calls, making them computationally expensive and inefficient. Recent works also attempt to amalgamate graph retrievers and LLMs (Peng et al., 2024; He et al., 2024; Hu et al., 2024; Mavromatis and Karypis, 2024), incorporating structural information during the retrieval process to improve performance. However, the graph retriever and LLMs are typically optimized separately, which limits their ability to fully exploit the synergy between the graph structure and LLMs' reasoning capabilities. Our proposed GRIL addresses the above issues by enabling end-to-end training between the GNN retriever and the LLM reasoner, optimizing both components in an interactive way.

3 Preliminaries and Background

Knowledge Graph Question Answering (KGQA) approaches aim to predict the correct answer a, given a question q and a KG $\mathcal{G}$ that provides relevant reasoning information. To reduce hallucination, a graph retriever extracts a subgraph $\mathcal{G}_s \subseteq \mathcal{G}$ that is the most relevant and useful for answering the question. The target is to learn a model that optimizes the conditional probability:

$$p(a|\mathcal{G}, q) = \sum_{\mathcal{G}_s} p_\phi(a|\mathcal{G}_s, q) p_\theta(\mathcal{G}_s|q, G), \quad (1)$$

where p_θ estimates the prior distribution on an extracted subgraph $\mathcal{G}_s$ conditioned on the given query q, and p_ϕ indicates the likelihood of the answer a given the query q and the subgraph $\mathcal{G}_s$, predicted by a reasoner (*e.g.,* LLMs). Maximizing the log-likelihood decouples the retriever from the reasoner as follows,

$$\mathcal{L} = \max_{\phi,\theta} \sum_{(q,a,\mathcal{G}_s)} \log p_\phi(a|\mathcal{G}_s, q) + \log p_\theta(\mathcal{G}_s|q). \quad (2)$$

4 Methodology

The overall framework of GRIL is shown in Figure 2. In KGQA tasks, a given question q is typically associated with query entities (*aka* seed entities) in the knowledge graph $\mathcal{G}$. These seed entities

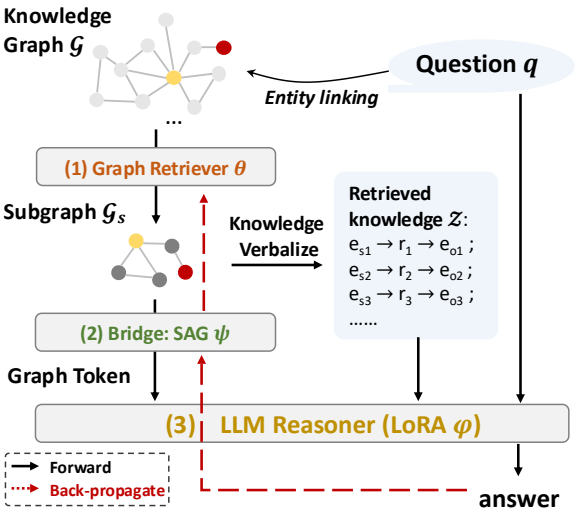

Figure 2: The framework of GRIL.

are either provided as part of the dataset annotations or could be identified by standard linking procedures (Neumann et al., 2019) when not explicitly available. A graph retriever based on the attention-based growing and pruning mechanism is used to extend the seed entities to a succinct multi-hop subgraph $\mathcal{G}_s \subseteq \mathcal{G}$ (Sec. 4.1.) A bridge module encodes the retrieved subgraph through (i) a soft graph token for necessary structural supervision and (ii) verbalized triples for semantic alignment with the LLM (Sec. 4.2). These components enable seamless integration with the LLM reasoner, where the LLM's output logits serve as implicit feedback to train the retriever, enabling end-to-end optimization that encourages the retriever to select the most useful subgraphs for answering the question (Sec. 4.3).

4.1 Attention-based Graph Retriever

Growing and Pruning Steps. The knowledge graph can be represented as a set of N fact triplets $\mathcal{G} = \{(e_s^{(i)}, r_{st}^{(i)}, e_t^{(i)})\}_{i=1}^N$, where $e_s^{(i)}$ and $e_t^{(i)}$ denote the source and target entities, and $r_{st}^{(i)}$ represents the relation, respectively. The attention-based graph retriever dynamically constructs the most relevant knowledge subgraph by employing a growing and pruning mechanism guided by attention scores between entities, as shown in Figure 3. Starting with an initial set of seed entities E_0 derived from the question q, it computes attention scores α_{ij} between each entity $e_i \in E_0$ and its neighbors $e_j \in \mathcal{N}(E_0)$, where the attention score is calcu-

lated by

$$\alpha_{ij} = \frac{\exp(\text{score}(c_{ij}))}{\sum_{k \in \mathcal{N}(E_0)} \exp(\text{score}(c_{ik}))}, \quad (3)$$

with $\text{score}(c_{ij}) = \text{Linear}([h_{e_i}, h_{e_j}, h_{r_{ij}}, h_q])$, and h_x indicates the initial representation of x (including entities, relations and the given question) generated by language models, such as Sentence-BERT (Reimers, 2019). In the growing step, each relation r_{ik} between $v_k \in \mathcal{N}(E_0)$ and $v_i \in E_0$ is associated with an attention score α_{ik}, indicating the probability that this relation would be grown from the seed entity v_i. Entities in the neighbor set $\mathcal{N}(E_0)$ are added to the entity set $E_1 = E_0 \cup \{v_k | \alpha_{ik} > 0; v_k \in \mathcal{N}(E_0); v_i \in E_0\}$ for the next growing step. In the pruning step, entities with probability scores higher than a certain threshold σ are retained in the subgraph, while others are pruned out. The retained nodes will grow to their neighbors in the next growing step. The growing and pruning steps are iteratively performed, ensuring a balance between approaching multi-hop relations and filtering out irrelevant noise.

Updating Entity Embedding. After each growing and pruning iteration, the graph retriever updates the entity embedding through a message-passing mechanism to aggregate neighbor information.

$$h'_{e_i} = W_1 h_{e_i} + W_2 \sum_{j \in \mathcal{N}(v_i)} \alpha_{ji} h_{e_j}, \quad (4)$$

where W_1 and W_2 are learnable weights and h_{e_i} denotes the embeddings of entity e_i. α_{ji} is the attention score calculated in the growing step (Eq. 3). The updated embeddings h'_{e_i} are subsequently used for recalculating attention scores in the next iteration. The Entity Embedding Updating step ensures that more relevant neighbors contribute more significantly to the updated entity embeddings, and refines the entity embeddings by incorporating multi-hop contextual information from its neighborhood, which is crucial for accurately modeling complex relationships in the graph.

Subgraph Output. Let $\mathbf{P} \in [0,1]^{|\mathcal{G}|}$ indicate the output probability scores across all relations (*i.e.*, edges) in $\mathcal{G}$. The final subgraph $\mathcal{G}_s$ is generated by $\mathcal{G}_s = \mathcal{G} \odot \mathbf{M}$ where the mask matrix $\mathbf{M}$ is sampled conditioned on the probability scores through a differentiable reparameterization trick (Jang et al., 2016) as follows,

$$\mathbf{M}_i = \sigma\left((\log \frac{\epsilon}{1-\epsilon} + \log \frac{\mathbf{P}_i}{1-\mathbf{P}_i})/\tau\right) \quad (5)$$

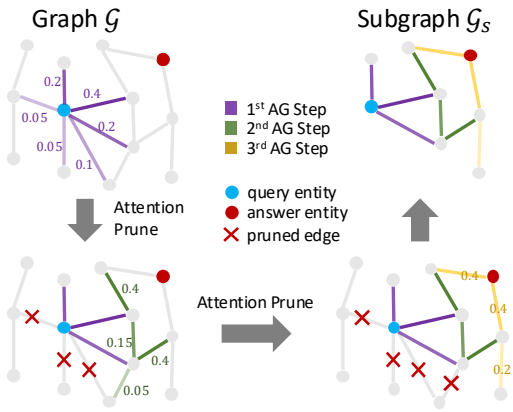

Figure 3: Illustration of Attention-based Graph Retriever. Different colors indicate different steps of Attention Growing (AG Step). Numbers represent the attention scores across neighboring edges. Edges with low attention scores are pruned for model efficiency and better retrieval quality.

for the i-th relation, where $\epsilon \sim \text{Uniform}(0,1)$, τ is the temperature and σ is the sigmoid function. Note that we use the approximately binary matrix $\mathbf{M}$ to achieve the growing and pruning operations during training to make it differentiable, enhancing both model efficiency and training stability. The detailed algorithm is given in Alg. 1 in Appendix C.1.

Complexity Assessment Module. Since questions across datasets vary in difficulty, they require different amounts of knowledge triplets for reasoning. We thereby propose a Complexity Assessment Module (CAM) that leverages an MLP to predict question complexity, measured by the number of reasoning hops required to reach answer entities. This module takes query embeddings as input and dynamically determines the number of knowledge triplets to provide to the LLM based on the predicted complexity level. The module could be pre-trained and treated as a preprocessing step, offering advantages over using a fixed hyperparameter to specify triplet count, as it automatically adjusts to varying question complexities. We refer to Appendix C.2 for more details.

4.2 Semantic and Structural Graph Encoding

The bridge module connects the retriever and reasoner by encoding both semantic and structural information, each addressing a distinct and necessary challenge in retrieval-augmented reasoning (Samel et al., 2023). To encode structural bias, we employ a self-attention graph pooling (SAG) (Lee et al., 2019) to generate a dense graph-level embedding from the extracted knowledge sub-

graph $\mathcal{G}_s$. The SAG layer computes self-attention scores $A^s \in \mathbb{R}^{|\mathcal{G}_s| \times 1}$, with A^s_i indicating the importance of entity $e_i \in \mathcal{G}_s$, and generates the graph token $h_{\text{GT}} = \text{MLP}(\sum_{e_i \in \mathcal{G}_s} A^s_i h'_{e_i})$ that aggregates global information, where h'_{e_i} denotes the retriever's contextualized embedding of e_i, and an MLP module projects the contextual embeddings into the same embedding space as the LLM, ensuring dimensional consistency. To incorporate semantic information and facilitate alignment with the LLM's language-based reasoning capabilities (Mavromatis and Karypis, 2024), we construct a verbalized subgraph by converting each retrieved triple $(e_s^{(i)}, r_{st}^{(i)}, e_t^{(i)})$ into a natural language format, expressed as $< e_s^{(i)} \rightarrow r_{st}^{(i)} \rightarrow e_t^{(i)} >$. These verbalized triples are concatenated with the original question as follows.

`[Graph Token] Based on the following reasoning paths, please answer the given question. \n Reasoning Paths:` $e_s^{(1)} \rightarrow r_{st}^{(1)} \rightarrow e_t^{(1)}; \cdots; e_s^{(n)} \rightarrow r_{st}^{(n)} \rightarrow e_t^{(n)}$ `\n Question: {Question} \n Answer: {Answer}`

`[Graph Token]` indicates the soft token derived by the SAG bridge from the global structure within the extract knowledge subgraph. {Question} and {Answer} are replaced by the question and answer in a certain sample, respectively. In implementation, let h_{IS} represent the token embeddings of the verbalized triplets and the question. We prepend the soft graph token to the embeddings of the input sequence to form $[h_{\text{GT}}||h_{\text{IS}})]$ as the final LLM input. The soft token is essential for end-to-end training, as it enables the reasoner to influence subgraph selection by optimizing the retriever through task-driven feedback. Together, the soft token and verbalized subgraph allow the reasoner to leverage both structural and semantic information, facilitating coherent reasoning in complex question-answering tasks.

4.3 Joint Training of Retriever and Reasoner

Extracting a sub-graph from a massive KG is a discrete, non-differentiable operation. Previous *GNN-as-Retriever* approaches (Mavromatis and Karypis, 2022, 2024; Jiang et al., 2022) circumvented this by ranking candidate sub-graphs with a frozen GNN and later fine-tuning an LLM on question-answer pairs. These methods typically rely on answer entities as the training labels, which requires additional cost for the entity labeling and focus merely on relevance regardless of whether the retrieved entities are truly useful for LLM reasoning. Moreover, it leads to a limitation in open-domain settings where answers are often free-form text rather than explicit KG entities. Instead, we propose to involve implicit feedback from LLMs as a supervision signal that not only optimizes the relevance of the retrieved knowledge but also ensures its maximal utility for the reasoning needs of the LLM. The training loss for this joint system is defined as

$$\mathcal{L}_{\text{joint}} = \max_{\phi,\psi} \log P_{\phi,\psi}(a|\mathcal{G}_s, q) \quad (6)$$

$$+ \max_{\theta} \log(P_{\phi,\psi}(a|\mathcal{G}_s, q) P_\theta(\mathcal{G}_s|q)) \quad (7)$$

ϕ and ψ are associated with the LLM reasoner and the SAG bridge, while θ pertains to the graph retriever. By Bayes' rule, the second part (Eq. 7) is equivalent to maximizing the posterior $p_{\phi,\psi,\theta}(\mathcal{G}_s|q,a)$, which involves the additional information from the ground truth answer a compared with the traditional objective Eq. 2. $\log(P_{\phi,\psi}(a|\mathcal{G}_s, q))$ is estimated by the LLM reasoner, reflecting its feedback on the quality of the retrieved subgraph $\mathcal{G}_s$. We apply a stop-gradient operator to stop updating the LLM reasoner and the SAG bridge when computing $\log(P_{\phi,\psi}(a|\mathcal{G}_s, q))$, ensuring that the gradient flows correctly during back-propagation. Therefore, the retriever and reasoner are enriched with the inductive bias from the retrieved knowledge subgraph. The LLM (ϕ) is finetuned with LoRA (Hu et al., 2021) conditioned on the retrieved subgraph $\mathcal{G}_s$ and question q.

Graph Supervision. In scenarios where answer entities are present in the given KG, prior works (Mavromatis and Karypis, 2022; Yasunaga et al., 2021) utilize answer entities as positive labels to supervise the graph retriever. In contrast, our approach expands the set of positive labels to include entities along the shortest paths between the query and answer entities. Specifically, for each question-answer pair (q, a), we extract $\mathcal{P}(q, a)$ as the set of entities that lie on any shortest path between the query entities (q) and gold answer entities (a) in the KG and supervise the retrieved subgraph $\mathcal{G}_s$ to cover $\mathcal{P}(q, a)$ with a binary cross-entropy loss. This additional graph supervision loss is crucial for guiding the retrieval process, as it helps establish the logical connections for effective reasoning. GRIL is trained with both graph supervision loss and the joint loss $\mathcal{L}_{\text{joint}}$. This dual supervision strategy ensures that retrieval prioritizes not just structural relevance but also logical coherence. In

open-domain scenarios where traditional methods fail due to the absence of explicit answer entities, GRIL is trained with a single $\mathcal{L}_{\text{joint}}$ containing LLM feedback, which serves as an alternative supervision signal and enables generalizable retrieval in weakly supervised scenarios.

5 Experiments

5.1 Datasets

We evaluate GRIL on KGQA tasks. Given a question q, the task is to extract relevant subgraphs from the given KG and leverage them for reasoning to get the answer a. We conduct experiments on WebQuestionsSP (WebQSP) (Yih et al., 2015) and Complex WebQuestions (CWQ) (Talmor and Berant, 2018). WebQSP contains 4,737 natural language questions and CWQ contains 34,699 total complex questions. Both are answerable with a subset Freebase KG within up to 2-hop for WebQSP and up to 4-hop for CWQ. Moreover, we test in the open-domain scenario, where answers might not be explicitly present in the KG. For this more challenging setting, we use the MedQA dataset to evaluate the proposed framework. MedQA contains 12,723 medical questions about disease diagnosis. MedQA is accompanied by the USMLE database and 18 medical textbooks. We manually curate a knowledge graph from the medical textbooks. More details are given in Appendix A.

5.2 Implementation

All the experiments are implemented with PyTorch on NVIDIA RTX A100 40GB GPUs. Standard dataset splits are applied to each dataset. Detailed hyperparameter setting is given in Appendix B. Following previous studies, we use Hits@1 and F1 as the evaluation metrics. Hits@1 measures the proportion of questions whose top-1 predicted answer is correct. F1 instead balances the precision and recall of the predicted answers. Experimental results in this paper are averaged from three runs with different random seeds.

5.3 Experimental Results

Baselines. The baselines can be categorized into four types: (1) *GNN-only* methods, including GraftNet (Sun et al., 2018), NSM (He et al., 2021a), UniKGQA (Jiang et al., 2022) and ReaRev (Mavromatis and Karypis, 2022) that solely rely on graph neural networks for reasoning; (2) *LLM-only* approaches including Llama (Dubey et al., 2024) and ChatGPT (Achiam et al., 2023) that utilize LLMs without graph structure; where Llama is fine-tuned on training set.(3) *LLM-as-Retriever* methods including KD-CoT (Wang et al., 2023), ToG (Sun et al., 2023), EffiQA (Dong et al., 2024), RoG (Luo et al., 2023) that leverage LLMs to generate relevant relation paths; and (4) *GNN-as-Retriever* approaches (Hu et al., 2024; He et al., 2024; Mavromatis and Karypis, 2024) that employ GNNs for retrieval and LLMs for reasoning. We select Llama3-8B as the LLM reasoner, while the proposed graph retriever is agnostic to any LLM reasoners.

Table 1: Performance comparison of different methods on the two KGQA benchmarks. We compare with *LLM-only*, *GNN-only*, LLM-as-Retriever (*L-as-R*) and GNN-as-Retriever (*G-as-R*) baselines.

		WebQSP		CWQ	
		Hits@1	F1	Hits@1	F1
GNN-only	GraftNet	66.4	60.4	36.8	32.7
	SR+NSM+E2E	69.5	64.1	49.3	46.3
	UniKGQA	77.2	72.2	51.2	49.1
	ReaRev	76.4	70.9	52.9	47.8
LLM-only	Llama2-7B	64.4	-	34.6	-
	Llama3-8B	65.2	-	35.8	-
	ChatGPT	66.8	-	39.9	-
	ChatGPT+CoT	75.6	-	48.9	-
L-as-R	KD-CoT	68.6	52.5	55.7	-
	ToG+Llama2-70B	68.9	-	57.6	-
	ToG+ChatGPT	76.2	-	58.9	-
	RoG	85.7	70.8	62.6	56.2
	ToG+GPT4	82.6	-	**69.5**	-
	EffiQA+GPT4	82.9	-	**69.5**	-
G-as-R	G-Retriever	70.1	-	-	-
	GRAG	72.7	-	-	-
	GNN-RAG	85.7	71.3	66.8	59.4
	GRIL (8B)	**86.8**	**73.0**	68.3	**60.5**

Results. The results on WebQSP and CWQ are shown in Table 1. Compared to prior *LLM-as-Retriever* methods, GRIL demonstrates significant improvements in both retrieval accuracy and reasoning quality. Importantly, GRIL achieves comparable or even better performance than leading pipelines (*e.g.,* EffiQA and ToG) that rely on proprietary LLMs like GPT-4, which raise potential concerns around accessibility and limited adaptability for domain-specific customization. In contrast, GRIL is built entirely on a smaller, open-source 8B LLM, yet still achieves superior results, showcasing the power of integrating structured retrieval with LLM reasoning in an end-to-end framework. Moreover, GRIL consistently outperforms *GNN-as-Retriever* methods by incorporating LLM feedback directly into retriever training, with a 1.35% average improvement over previous baseline (Mavromatis and Karypis, 2024). The joint optimization al-

lows GRIL to identify more relevant subgraphs and retrieve information better aligned with the LLM's reasoning trajectory, resulting in more precise and robust QA performance across both datasets.

Table 2: Hits@1 performance of different retrievers with two LLM reasoners on WebQSP dataset

Retriever	Mistral-7B	Llama-8B
None	61.3	65.2
ES	76.8	75.7
RoG	83.6	86.3
GNN-RAG	83.4	85.4
GRIL$_{\text{separate}}$	84.3	86.4
GRIL$_{\text{end-to-end}}$	**85.2**	**86.8**

Table 2 presents Hits@1 performance comparing different retriever methods paired with different LLM reasoners on the WebQSP dataset. We implement Embedding Similarity (ES), which employs dot-product similarity on contextualized representations from RoBERTa-large (Liu, 2019) as a dense retrieval approach. All LLM reasoners are fine-tuned with the respective retrieval to ensure a fair comparison. Across both Mistral-7B and Llama3-8B, GRIL demonstrates superior performance, outperforming other retrievers including ES, RoG (*LLM-as-retriever*), and GNN-RAG (*GNN-as-retriever*), which highlights the robustness and generalizability of the proposed method.

Why End-to-End Wins. GRIL$_{\text{separate}}$, which removes the graph soft token and thereby disables LLM feedback (Eq. 7), shows a noticeable performance drop. This suggests that retrieval quality is significantly enhanced through joint optimization with the LLM. Overall, the end-to-end training within GRIL offers two key advantages: (1) it eliminates the need for separately training the retriever, reducing compute costs; and (2) it generalizes better to settings where ground-truth entities are unavailable, as the retriever can be learned directly through supervision from LLMs' signals. These benefits make GRIL particularly useful for open-domain and weakly supervised scenarios.

5.4 Open-domain Scenario: MedQA

Baselines. We compare GRIL against a diverse set of strong baselines spanning classical information retrieval, dense retrieval, and hybrid methods. We compare with Embedding Similarity (ES) based on dot-product similarity. BM25 (Robertson et al., 2009) leverages exact lexical matching through TF-IDF statistics, which remains competitive in many retrieval scenarios. Contriever (Izacard et al., 2021) employs contrastive learning with large-scale pre-training on web documents, while SPECTER (Cohan et al., 2020) leverages scientific documents for more accurate representation. BMRetriever (Xu et al., 2024) is specifically designed for biomedical information retrieval and pre-trained on massive medical data. All baselines are implemented using the official codebases to ensure fair comparison.

Table 3: Accuracy (%) on MedQA. † indicates that the retriever requires pre-training on massive data.

Model	Llama2-7b	Llama3-8b
Without Retriever		
Zero-shot	42.3±1.8	60.8±1.6
Fine-tuned	44.8±1.0	61.7±1.2
With Retriever		
ES	47.6±1.7	63.0±1.5
BM25	48.6±1.6	62.5±1.3
Contriever†	49.3±1.4	63.1±1.0
SPECTER†	51.8±0.9	66.2±1.1
BMRetriever†	57.4±1.7	68.9±1.1
GRIL	**58.9**±1.3	**70.4**±1.6

Results. The results in Table 3 highlight GRIL's outstanding performance on MedQA dataset, where most baselines from Table 1 fail due to their reliance on answer entities for retriever training. Our GRIL achieves the highest accuracy across both Llama2-7B and Llama3-8B, significantly outperforming all the baselines, including retrievers that have been massively pre-trained on large-scale data. These results emphasize how GRIL operates effectively in open-domain scenarios, where no predefined answer entities are available, and the retrieval process must dynamically adapt to new and unseen queries. The improvement can be attributed to GRIL's structural awareness achieved by the graph retriever and reverse feedback loop from the LLM, key features that distinguish it from conventional RAG methods.

6 More Evaluation and Analysis

6.1 Ablation Study

GNN Depth Sensitivity and Complexity Assessment Module (CAM) Impact. We conduct a sensitivity analysis on the number of GNN layers to assess the importance of the Complexity Assessment Module (CAM), as shown in Figure 4. Specifically, we evaluate model performance across varying GNN depths, ensuring the CAM remains consistently integrated (yellow line). Additionally, we

compare this to an alternative approach where a fixed number of triplets is retrieved (*e.g.,* 16, 32, 64), bypassing the CAM (green line). As shown in the figure, when using a fixed number of triplets, performance initially improves with more triplets but peaks and declines beyond 16 triplets. Instead, CAM successfully addresses the sensitivity with a more stable performance (yellow line). CAM dynamically adjusts the number of retrieved triplets based on the question complexity level, achieving improved performance while avoiding the computational overhead introduced by excessive retrieval.

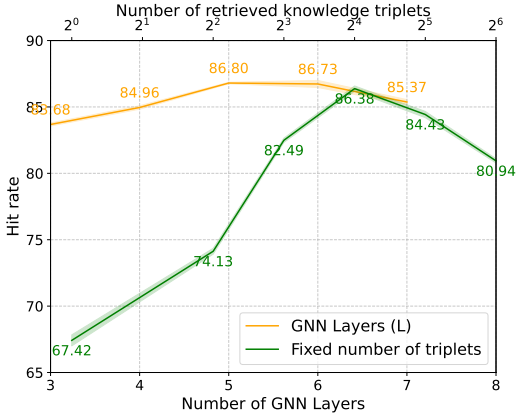

Figure 4: Importance of GNN depth and Complexity Assessment Module on WebQSP dataset

Ablation on Graph Retriever. Table 4 presents the effects of different graph pruning mechanisms (*e.g.,* threshold-based or top-K strategy) and specific operators on the WebQSP dataset. When using threshold-based pruning, increasing the threshold tends to slightly improve inference efficiency while reducing the F1 scores. $\sigma = 0.1$ yields the highest F1 and a balance between performance and efficiency, which is set as default in GRIL. Removing key mechanisms, such as the pruning operation and the entity embedding updating step (Eq. 4) in the graph retriever, results in significant performance drops. The absence of pruning also substantially increases inference time and destroys the performance, underscoring its importance in maintaining retrieval efficiency and precision.

6.2 Combination with Small LMs

Table 5 compares the performance of small language models (LMs) and LLMs as the reasoner on the WebQSP dataset. GRIL, paired with a small LM, *e.g.,* RoBERTa or BERT, demonstrates comparable or even better performance to LLMs, while

Table 4: Ablation study on the graph retriever

	F1	inference time (s)
Pruning Mechanism Alternatives		
Threshold ($\sigma = 0.1$)	**72.68**	0.476
Threshold ($\sigma = 0.2$)	$71.82_{(\downarrow 1.18\%)}$	$0.463_{(\downarrow 2.73\%)}$
Threshold ($\sigma = 0.5$)	$71.61_{(\downarrow 1.47\%)}$	$0.423_{(\downarrow 11.13\%)}$
Top 5	$72.54_{(\downarrow 0.19\%)}$	$0.459_{(\downarrow 3.57\%)}$
Top 10	$72.60_{(\downarrow 0.11\%)}$	$0.463_{(\downarrow 2.73\%)}$
Top 20	$72.37_{(\downarrow 0.43\%)}$	$0.479_{(\uparrow 0.63\%)}$
Removing Key Steps in Graph Retriever		
w/o pruning	$70.28_{(\downarrow 3.30\%)}$	$0.687_{(\uparrow 44.33\%)}$
w/o Entity Update	$70.32_{(\downarrow 3.25\%)}$	$0.437_{(\downarrow 8.19\%)}$

Table 5: Performance comparison between small language models and LLMs as the reasoner.

	WebQSP			
	Hits@1	F1	Time (s)	Size
BERT-large	40.8	-	0.97	$\sim$ 336M
RoBERTa-large	41.3	-	0.87	$\sim$ 355M
Llama2-7B	64.4	-	4.02	$\sim$ 7B
Llama3-8B	65.2	-	3.87	$\sim$ 8B
GRIL *w/* BERT	64.8	70.2	1.54	$\sim$ 768M
GRIL *w/* RoBERTa	67.7	71.4	1.32	$\sim$ 806M

incurring significantly lower inference costs. The observations justify that graph-based reasoning capability compensates for the reduced model size without sacrificing accuracy. This result underscores the parameter efficiency of our approach, as it effectively harnesses the graph retriever to enhance reasoning and retrieval quality, making it highly practical for resource-constrained scenarios. Importantly, GRIL's retrieval cost grows with the size of the extracted subgraph, not with the full knowledge graph (when the entity embedding updating step is disabled). In addition, the Complexity Assessment Module (CAM) predicts the minimal hop depth required for each query, effectively bounding the retrieval scope. This design keeps inference latency stable even as the full KG size increases, ensuring that GRIL scales efficiently to large graphs.

6.3 Case Studies

We conduct a case study on CWQ dataset to illustrate the effectiveness of GRIL on retrieval and complex reasoning, as shown in Figure 5. While ChatGPT provides partially correct answers, it fails to capture crucial aspects of the ground truth (*i.e.,* "vocals" in the case). Instead, GRIL successfully identifies the necessary logical connections between the question entity ("Randy Jackson") and the answer entity ("Vocals"). We visualize how

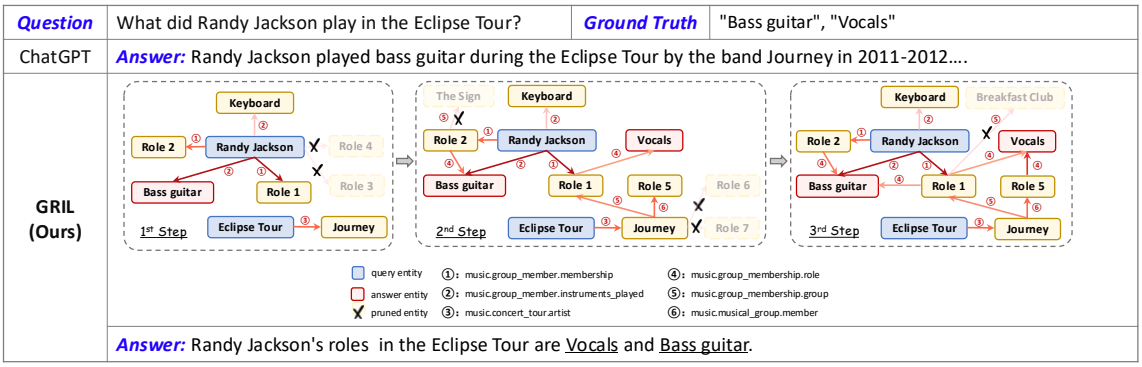

Figure 5: A case study of GRIL retrieval and reasoning on CWQ dataset. In the Retrieved Subgraph, the edge color intensity indicates the importance score of the certain knowledge triplet.

GRIL navigates the knowledge graph to establish connections between relevant entities, through iterative growing and pruning. It clearly highlights the predicate relationships that lead to a comprehensive and accurate answer generated by the LLM reasoner. Moreover, the edge color intensity represents the importance score of each edge at its corresponding growing step, further enhancing the GRIL's self-interpretability and providing insights into the underlying reasoning logic of GRIL.

7 Conclusion

In this work, we present GRIL, a novel framework that integrates graph retrieval and reasoning through attention-based growing and pruning mechanisms and joint training with LLMs. Our approach enhances reasoning over knowledge graphs and eliminates the need for predefined answer entities, making it highly effective in open-domain scenarios. Experimental results show significant performance improvement on KGQA tasks and superior scalability, while also improving inference efficiency. GRIL provides a cost-effective, state-of-the-art solution for knowledge-intensive tasks and offers potential for real-world applications.

Limitations and Future Work. While the proposed GRIL shows promise, several aspects could be further explored to extend its applicability. First, GRIL assumes that the graph structure is inherently necessary for question answering, as it relies on structured knowledge graphs for multi-hop reasoning. However, this assumption may limit its ability to handle scenarios where the underlying knowledge is either unstructured or where the graph structure does not fully capture the complexity of natural language semantics. Future work could explore ways to automatically and organically integrate *GNN-as-Retriever* and *LLM-as-Retriever* approaches, enabling the model to dynamically determine when to leverage graph-based reasoning and when to rely on unstructured, text-based retrieval. Moreover, extending GRIL to other applications that require graph reasoning, such as recommendation systems and biomedical knowledge extraction, may reveal additional challenges and opportunities for improvement.

References

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

Junwei Bao, Nan Duan, Zhao Yan, Ming Zhou, and Tiejun Zhao. 2016. Constraint-based question answering with knowledge graph. In *Proceedings of COLING 2016, the 26th international conference on computational linguistics: technical papers*, pages 2503–2514.

Olivier Bodenreider. 2004. The unified medical language system (umls): integrating biomedical terminology. *Nucleic acids research*, 32(suppl_1):D267–D270.

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

Tom B Brown. 2020. Language models are few-shot learners. *arXiv preprint arXiv:2005.14165*.

Arman Cohan, Sergey Feldman, Iz Beltagy, Doug Downey, and Daniel S Weld. 2020. Specter: Document-level representation learning using citation-informed transformers. *arXiv preprint arXiv:2004.07180*.

Zixuan Dong, Baoyun Peng, Yufei Wang, Jia Fu, Xiaodong Wang, Yongxue Shan, and Xin Zhou. 2024. Effiqa: Efficient question-answering with strategic multi-model collaboration on knowledge graphs. *arXiv preprint arXiv:2406.01238*.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Wenqi Fan, Yujuan Ding, Liangbo Ning, Shijie Wang, Hengyun Li, Dawei Yin, Tat-Seng Chua, and Qing Li. 2024. A survey on rag meeting llms: Towards retrieval-augmented large language models. In *Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 6491–6501.

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

Yu Gu, Robert Tinn, Hao Cheng, Michael Lucas, Naoto Usuyama, Xiaodong Liu, Tristan Naumann, Jianfeng Gao, and Hoifung Poon. 2020. Domain-specific language model pretraining for biomedical natural language processing.

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

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

Xiaoxin He, Yijun Tian, Yifei Sun, Nitesh V Chawla, Thomas Laurent, Yann LeCun, Xavier Bresson, and Bryan Hooi. 2024. G-retriever: Retrieval-augmented generation for textual graph understanding and question answering. *arXiv preprint arXiv:2402.07630*.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2021. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*.

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

Xiao Huang, Jingyuan Zhang, Dingcheng Li, and Ping Li. 2019. Knowledge graph embedding based question answering. In *Proceedings of the twelfth ACM international conference on web search and data mining*, pages 105–113.

Gautier Izacard, Mathilde Caron, Lucas Hosseini, Sebastian Riedel, Piotr Bojanowski, Armand Joulin, and Edouard Grave. 2021. Unsupervised dense information retrieval with contrastive learning. *arXiv preprint arXiv:2112.09118*.

Eric Jang, Shixiang Gu, and Ben Poole. 2016. Categorical reparameterization with gumbel-softmax. *arXiv preprint arXiv:1611.01144*.

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

Di Jin, Eileen Pan, Nassim Oufattole, Wei-Hung Weng, Hanyi Fang, and Peter Szolovits. 2021. What disease does this patient have? a large-scale open domain question answering dataset from medical exams. *Applied Sciences*, 11(14):6421.

Vladimir Karpukhin, Barlas Oğuz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. *arXiv preprint arXiv:2004.04906*.

Junhyun Lee, Inyeop Lee, and Jaewoo Kang. 2019. Self-attention graph pooling. In *International conference on machine learning*, pages 3734–3743. pmlr.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in Neural Information Processing Systems*, 33:9459–9474.

Shiyang Li, Yifan Gao, Haoming Jiang, Qingyu Yin, Zheng Li, Xifeng Yan, Chao Zhang, and Bing Yin. 2023. Graph reasoning for question answering with triplet retrieval. *arXiv preprint arXiv:2305.18742*.

Xiaoxi Li, Jiajie Jin, Yujia Zhou, Yuyao Zhang, Peitian Zhang, Yutao Zhu, and Zhicheng Dou. 2024. From matching to generation: A survey on generative information retrieval. *arXiv preprint arXiv:2404.14851*.

Yinhan Liu. 2019. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 364.

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

Costas Mavromatis and George Karypis. 2022. Rearev: Adaptive reasoning for question answering over knowledge graphs. *arXiv preprint arXiv:2210.13650*.

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

Mark Neumann, Daniel King, Iz Beltagy, and Waleed Ammar. 2019. Scispacy: fast and robust models for biomedical natural language processing. *arXiv preprint arXiv:1902.07669*.

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*.

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

N Reimers. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*.

Ridho Reinanda, Edgar Meij, Maarten de Rijke, et al. 2020. Knowledge graphs: An information retrieval perspective. *Foundations and Trends® in Information Retrieval*, 14(4):289–444.

Stephen Robertson, Hugo Zaragoza, et al. 2009. The probabilistic relevance framework: Bm25 and beyond. *Foundations and Trends® in Information Retrieval*, 3(4):333–389.

Karan Samel, Houyu Zhang, Jun Ma, Haoming Jiang, Qing Ping, Sheng Wang, Yi Xu, Belinda Zeng, and Trishul Chilimbi. 2023. Sst: Semantic and structural transformers for hierarchy-aware language models in e-commerce. In *2023 IEEE International Conference on Big Data (BigData)*, pages 838–846.

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

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

Jiashuo Sun, Chengjin Xu, Lumingyuan Tang, Saizhuo Wang, Chen Lin, Yeyun Gong, Heung-Yeung Shum, and Jian Guo. 2023. Think-on-graph: Deep and responsible reasoning of large language model with knowledge graph. *arXiv preprint arXiv:2307.07697*.

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

Keheng Wang, Feiyu Duan, Sirui Wang, Peiguang Li, Yunsen Xian, Chuantao Yin, Wenge Rong, and Zhang Xiong. 2023. Knowledge-driven cot: Exploring faithful reasoning in llms for knowledge-intensive question answering. *arXiv preprint arXiv:2308.13259*.

David S Wishart, Yannick D Feunang, An C Guo, Elvis J Lo, Ana Marcu, Jason R Grant, Tanvir Sajed, Daniel Johnson, Carin Li, Zinat Sayeeda, et al. 2018. Drugbank 5.0: a major update to the drugbank database for 2018. *Nucleic acids research*, 46(D1):D1074–D1082.

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

Ran Xu, Wenqi Shi, Yue Yu, Yuchen Zhuang, Yanqiao Zhu, May D Wang, Joyce C Ho, Chao Zhang, and Carl Yang. 2024. Bmretriever: Tuning large language models as better biomedical text retrievers. *arXiv preprint arXiv:2404.18443*.

Mohammad Yani and Adila Alfa Krisnadhi. 2021. Challenges, techniques, and trends of simple knowledge graph question answering: a survey. *Information*, 12(7):271.

Michihiro Yasunaga, Hongyu Ren, Antoine Bosselut, Percy Liang, and Jure Leskovec. 2021. Qa-gnn: Reasoning with language models and knowledge graphs for question answering. *arXiv preprint arXiv:2104.06378*.

Scott 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 Joint Conference of the 53rd Annual Meeting of the ACL and the 7th International Joint Conference on Natural Language Processing of the AFNLP*.

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

Weiguo Zheng, Hong Cheng, Lei Zou, Jeffrey Xu Yu, and Kangfei Zhao. 2017. Natural language question/answering: Let users talk with the knowledge graph. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, pages 217–226.

A Dataset

We evaluate GRIL on three question answering datasets: WebQSP (Yih et al., 2015), CWQ (Talmor and Berant, 2018) and MedQA (Jin et al., 2021). Detailed dataset statistics are shown in Table 6.

WebQSP and CWQ. Both datasets are designed for question answering tasks that leverage the Freebase knowledge graph (Bollacker et al., 2008), which comprises over 164.6 million facts and 24.9 million entities. WebQSP primarily requires up to 2-hop reasoning to answer questions, whereas CWQ presents a more complex challenge, necessitating up to 4-hop reasoning over the provided knowledge graph. We follow the previous setting (Luo et al., 2023; He et al., 2021b; Jiang et al., 2022)[1][2] for knowledge graph extraction. Specifically, the input knowledge graph for each question is constructed by a subset of Freebase KG that contains all triples within the max reasoning hops of question entities. We follow the previous studies (Luo et al., 2023) for dataset split. The initial seed entities are derived from the question and provided in the dataset. Both benchmarks are inherently tied to a specific knowledge graph (KG) and our experiments are conducted using the same KG provided by the respective benchmarks.

MedQA. is a 4-way multiple-choice medical question-answering task, originating from practice tests for the United States Medical License Exams (USMLE), which generally require a deep understanding of related medical concepts from associated medical textbooks. We utilize the original dataset split setting (Jin et al., 2021), with 80% for training, 10% for development, and 10% for test. For MedQA, we use a self-constructed knowledge graph based on the 18 given medical textbooks. Specifically, we use ScispaCy (Neumann et al., 2019) to identify biomedical concepts as entities and the *RecursiveCharacterTextSplitter* from LangChain[3] to split the medical textbooks into snippets. Edges are created between two entities that are mentioned within one snippet. The embeddings of entities and relations are initialized using Sentence PubmedBert (Gu et al., 2020). Note that we do not consider the effect of different entity recognition tools and splitters, as they are orthogonal to the focus of this work. Experimental comparisons with baselines are based on the same curated knowledge graph.

Compared with the knowledge graph in previous studies (Yasunaga et al., 2021) built on the Disease Database of the Unified Medical Language System (UMLS) (Bodenreider, 2004) and DrugBank (Wishart et al., 2018), our curated knowledge graph demonstrates a significantly improved answer coverage, increasing from 24.6% to 88.4%.

Table 6: Dataset statistics. Coverage indicates the

Dataset	Train	Dev	Test	Coverage(%)
WebQSP	2,848	250	1,639	94.9
CWQ	27,639	3,519	3,531	79.3
MedQA	10,178	1,272	1,273	88.4

B Training Details

We set the hidden size as 512 in the graph retriever and the GNN encoder. The batch size is set to 2 during training and 4 during evaluation. The learning rate is 1e-5. The maximum number of epochs is 100. An early stopping strategy is used to mitigate overfitting. We utilize the LoRA (Hu et al., 2021) technique to finetune the LLM reasoner with rank 8 by default. All experiments are conducted with PyTorch on NVIDIA RTX A100 GPUs for three runs with different random seeds.

C Additional Module Details

C.1 Attention-based Graph Retriever

The detailed algorithm of our attention-based graph retriever is shown in Alg. 1. The attention scores are calculated for each edge (*i.e.,* knowledge triplet). To avoid useless attention growing for irrelevant entities and keep the focus on important entities, the graph retriever iteratively performs growing and pruning steps. After the attention calculation, if the number of triplets with attention scores lower than $\sigma = 0.1$ is larger than a certain budget (*e.g.,* 16), then the retriever automatically performs the pruning step. The hyperparameter sensitivity of σ is shown in Table 4. Moreover, the entity embeddings are updated by message-passing and aggregating mechanisms with the previously calculated attention scores. The output of the algorithm contains G with updated entity embeddings and probabilities on each triplet $\mathbf{P}$. The final subgraph $\mathcal{G}_s$ is generated by $\mathcal{G}_s = \mathcal{G} \odot \mathbf{M}$ where the mask matrix $\mathbf{M}$ is sampled conditioned on the probability

[1] https://huggingface.co/datasets/rmanluo/RoG-webqsp

[2] https://huggingface.co/datasets/rmanluo/RoG-cwq

[3] https://www.langchain.com

scores through a differentiable reparameterization trick (Jang et al., 2016).

C.2 Complexity assessment module

We approach the complexity assessment as a classification task, utilizing a multilayer perceptron (MLP) to predict the question complexity (i.e., the number of reasoning hops) based on the query embedding generated by language models. The ground truth is defined as the shortest path distance between the query entities and the answer entities. The MLP is trained using cross-entropy loss, comparing the predicted number of hops with the ground truth. Table 7 presents the prediction accuracy (%) when the model is trained on individual datasets or a combined dataset of WebQSP and CWQ. Notably, BERT outperforms RoBERTa in all settings, achieving higher accuracy on both individual datasets (WebQSP and CWQ) as well as the joint dataset (WebQSP+CWQ). The joint dataset consistently yields the best results, with BERT achieving the highest accuracy of 74.28%, showcasing the benefits of combining diverse reasoning tasks to improve generalization. We select BERT as the language model and train the MLP on the combined dataset as the default Complexity Assessment Module. Given the predicted number of hops, c, for a specific question, we allocate $5 \times c$ as the number of final retrieved triplets to be provided for downstream reasoning. Notably, this module can be pre-trained and treated as a preprocessing step, enhancing its efficiency. Alternatively, a fixed hyperparameter can be employed to specify the number of retrieved triplets, offering a trade-off for reduced computational overhead.

Table 7: Accuracy (%) of number of hops prediction

Dataset	WebQSP	CWQ	WebQSP+CWQ
RoBERTa	63.33	70.28	**73.46**
BERT	64.98	72.36	**74.28**

C.3 Retrieval Augmentation Ensemble

Retrieval augmentation (RA) (Mavromatis and Karypis, 2024) enhances the performance of LLM reasoners by aggregating knowledge retrieved through different mechanisms. Building on the previous work (Mavromatis and Karypis, 2024), we extend the *GNN-as-Retriever* paradigm by incorporating an *LLM-as-Retriever* approach to further enrich the retrieval process. Specifically, we integrate reasoning paths retrieved from RoG (Luo et al., 2023), complementing them with those retrieved by our graph-based method. This union of reasoning paths combines the strengths of both graph-based and language-based retrieval, thereby expanding the diversity of knowledge incorporated into the reasoning process. As a result, this approach not only broadens the scope of relevant information but also enhances the robustness and accuracy of the overall LLM reasoning mechanism. Beam-search decoding is used in *LLM-as-Retriever* approaches to generate diverse reasoning paths for better answer coverage. We set the number of beams as 3 in RoG and report the performance of Retrieval augmentation (RA) in Table 8.

Table 8: KGQA Performance with and without RA on WebQSP and CWQ dataset

	WebQSP		CWQ	
	Hits@1	F1	Hits@1	F1
GNN-RAG	85.7	71.3	66.8	59.4
GNN-RAG+RA	90.7	**73.5**	68.7	60.4
GRIL	86.8	73.0	68.3	60.5
GRIL+RA	**91.4**	73.1	**69.2**	**61.8**

We observe that Retrieval augmentation (RA) consistently improves the KGQA performance, with an average improvement rate of 3.58% on WebQSP and 1.99% on CWQ. GRIL demonstrates superiority when paired with RA. For example, GRIL+RA achieves the highest Hits@1 on both WebQSP and CWQ, outperforming GNN-RAG+RA by significant margins of 0.7% and 0.5%, respectively. While GRIL without RA already outperforms baselines on both datasets, RA further enhances its performance. This demonstrates GRIL's ability to better exploit the additional reasoning paths provided by RA, particularly in the more complex CWQ dataset, which features longer and more intricate question-answering dependencies.

C.4 Ablation on Textual Subgraph

We further study the role of textualizing the retrieved subgraph. In standard retrieval-augmented frameworks, converting the retrieved knowledge into natural language is a widely adopted practice, since LLMs are inherently trained to process text rather than raw graph embeddings. To quantify its effect, we compare GRIL with and without the textual representation of subgraphs on MedQA dataset.

Algorithm 1: Attention-based Graph Retriever

Input: Knowledge Graph $G = \{(e_s^{(i)}, e_t^{(i)}, r_{st}^{(i)})\}_{i=1}^{N}$, Query q, Seed Entity e_0, Number of Layers L
Output: Probability on triplets, Updated G

Initialize zero $\mathbf{E} \in \mathbb{R}^N$; `// Initialize probability on triplets`
$P_1 \leftarrow$ get-neighbor$(\{e_0\})$; `// Retrieve initial list of neighbor triplets`
for $i = 1, 2, \ldots, L$ **do**
  $\mathbf{A}_i \leftarrow \text{AttnScore}(P_i, q_0, G) \in [0,1]^{|P_i|}$; `// Compute Attention Scores (Eq. 3) on` P_i
  if *Pruning step* **then**
    $\mathbf{A}_i \leftarrow [\mathbf{A}_i > \sigma]$; `// Keep attention scores greater than threshold` σ
  end
  $P_{i+1} \leftarrow$ get-neighbor$(P_i, \mathbf{A}_i)$; `// Update neighbors based on attention weights`
  $\mathbf{E} \leftarrow \text{update}(\mathbf{E}, \mathbf{A}_i) \in \mathbb{R}^N$; `// Update probability on triplets`
  for $v \in P_i$ **do**
    $\mathcal{N}(v) \leftarrow \{\text{neighbors of } v \text{ in } P_i \text{ with non-zero attention}\}$
    $\mathbf{m}_v \leftarrow \sum_{u \in \mathcal{N}(v)} \mathbf{A}_i[u] \cdot \text{Message}(e_u, r_{uv})$; `// Aggregate messages`
    $e_v \leftarrow \text{Update}(e_v, \mathbf{m}_v)$; `// Update entity embedding`
  end
end
return $\mathbf{E}$, G with updated embedding

From Table 9, we observe that removing the textual representation and the graph soft token both lead to a significant performance drop, highlighting their importance and necessity in transferring semantic and structural information.

Table 9: Ablation study on the textualization of the retrieved subgraph.

Model	Llama2-7B	Llama3-8B
GRIL	**58.9**	**70.4**
w/o soft token	53.6	66.1
w/o textual subgraph	50.7	64.8