GRAPHSEARCH: An Agentic Deep Searching Workflow for Graph Retrieval-Augmented Generation

Anonymous ACL submission

Abstract

Graph Retrieval-Augmented Generation enhances factual reasoning in large language models by structurally modeling knowledge through graph-based representations. Existing GraphRAG approaches face two core limitations: shallow retrieval that fails to surface all critical evidence, and inefficient utilization of pre-constructed structural graph data. To address these challenges, we propose GRAPHSEARCH, a novel agentic deep searching workflow with dual-channel retrieval for GraphRAG. GRAPHSEARCH organizes the retrieval process into a modular framework comprising six modules, enabling multi-turn interactions and iterative reasoning. Furthermore, GRAPHSEARCH adopts a dual-channel retrieval strategy that issues semantic queries over chunk-based text data and relational queries over structural graph data, enabling comprehensive utilization of both modalities and their complementary strengths. Experimental results across six multi-hop RAG benchmarks demonstrate that GRAPHSEARCH consistently improves accuracy and generation quality over the traditional strategy, confirming GRAPHSEARCH as a promising direction for advancing agentic graph retrieval-augmented generation.

1 Introduction

Large language models demonstrates remarkable capabilities in natural language understanding and reasoning (Zhao et al., 2023; Naveed et al., 2025). Retrieval-augmented generation has emerged as a paradigm that combines LLMs with external knowledge bases, enhancing factuality, credibility and interpretability in knowledge-intensive tasks (Lewis et al., 2020). Graph Retrieval-Augmented Generation (GraphRAG) is introduced to overcome the shortcomings of traditional RAG, which relies solely on semantic similarity for retrieval (Peng et al., 2024). By constructing structural graph knowledge bases (graph KBs) and leveraging hierarchical retrieval strategies, GraphRAG strengthens the integration of contextual information across massive entities and relationships (Sarthi et al., 2024; Edge et al., 2024; Guo et al., 2024).

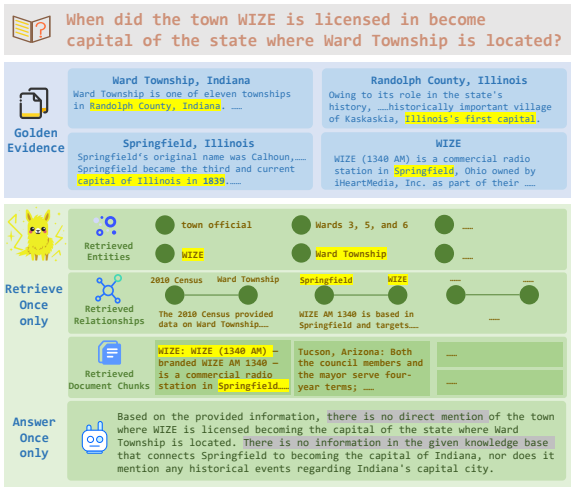

Figure 1: An illustration of the shallow retrieval problem in traditional GraphRAG methods, in which retrieval and answering are conducted only once, resulting in missing critical entities in knowledge graph database.

However, existing GraphRAG approaches still face challenges that lead to performance bottlenecks: (i) **Shallow retrieval results in missing evidence for complex queries.** Most GraphRAG methods adopt a single-round retrieval-and-generation process as the interaction strategy between the LLM and the graph KB (Edge et al., 2024; Guo et al., 2024; Fan et al., 2025). However, as illustrated in Figure 1, when handling a complex query that requires four pieces of golden evidence, *"When did the town WIZE is licensed in become capital of the state where Ward Township is located?"*, the entity *Randolph County* is not retrieved by the LightRAG retriever, and the reasoning process suffers from insufficient evidence. (ii) **Limited ability to exploit structural data due to constrained retrieval scope**. Existing GraphRAG methods with heuristic path-construction schemes (Fan et al., 2025; Chen et al., 2025; Jimenez Gutierrez et al.,

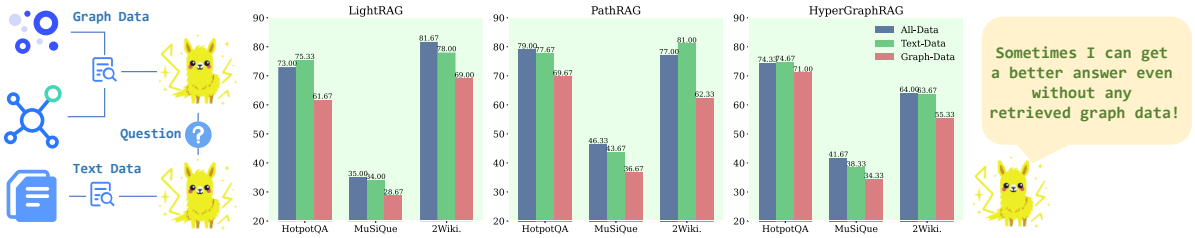

Figure 2: Comparison of using relational graph data only, textual data only, or all data as commonly adopted in GraphRAG approaches. The metric is SubEM. The contribution of retrieved graph data for reasoning is marginal.

2024) often fail to fully leverage the structural information in graph KBs, fundamentally because shallow retrieval restricts the coverage of relevant nodes and relations. Without sufficient coverage of retrieved subgraphs, the available structural signals are fragmented and sparse, making it difficult for LLMs to integrate semantic and structural modalities for complex reasoning. As shown in Figure 2, models may perform comparably with semantic textual evidence only, highlighting that the underutilization of graph data is tightly coupled with the limitations of current retrieval strategies.

We propose GRAPHSEARCH, an agentic deep searching workflow for GraphRAG. As illustrated in Figure 3, GRAPHSEARCH is a novel agent framework designed to access graph KBs through dual-channel retrieval, acquiring both semantic and structural information, and performing multi-turn interactions to complete complex reasoning tasks. Targeting the shallow retrieval problem in existing GraphRAG approaches, GRAPHSEARCH models retrieval as an agentic modular searching pipeline. Through the coordinated contributions of these modules, GRAPHSEARCH decomposes complex queries into tractable atomic sub-queries, retrieves fine-grained knowledge from graph KBs, and iteratively performs logical reasoning and reflection to remedy missing evidence. Furthermore, GRAPHSEARCH adopts a dual-channel retrieval strategy, constructing semantic queries over chunk-based text data and relational queries over structural graph data, thereby fully synergizing both modalities and integrating them into contexts that support LLMs in complex reasoning. The results through extensive experiments conducted on six multi-hop RAG datasets demonstrate that leveraging the graph KBs retrievers built upon the corresponding GraphRAG approaches, GRAPHSEARCH consistently outperforms the traditional single-round interaction strategy. Furthermore, the effectiveness of the dual-channel retrieval strategy, the contributions of agentic modules, and its robustness under a small-scale LLM and varying retrieval budgets are all empirically validated.

Our contributions are as follows: (i) We propose GRAPHSEARCH, an agentic deep searching workflow that overcomes the challenges of shallow retrieval and the ineffective use of relational graph data in existing GraphRAG approaches. (ii) We introduce a modular searching pipeline for iterative reasoning and a dual-channel retrieval strategy integrating semantic and relational queries over graph KBs. (iii) Experiment results across six multi-hop RAG datasets demonstrating that GRAPHSEARCH consistently outperforms vanilla GraphRAG.

2 Preliminaries

Graph Knowledge Database. Given a document collection D, the graph indexer ϕ segments D into a set of text chunks K. For each chunk $k \in K$, an extractor $\mathcal{R} \in \phi$ identifies a set of entities $e = \{e_{\text{name}}, e_{\text{prop}}, e_{\text{desc}}\}$. For any pair of entities $e_h, e_t \in k$, a relation is defined as $r = \{e_h, e_t, r_{\text{prop}}, r_{\text{desc}}\}$. Aggregating all entities and relations yields the graph KB $G = \{E, R, K\}$, where E denotes the entity set, R the relation set, and K the associated chunk-level textual context.

Graph KB Retrieval. Given a query q, a graph KB retriever ψ selects a relevant context set $C = \{E_q, R_q, K_q\} \subset G$ that maximizes semantic relevance to q. The retriever aims to return structural graph data and chunk-based text data that provide sufficient evidence for answer generation.

LLM Answer Generation. The language model consumes the query q together with the retrieved context C to generate an output y. The generation is modeled as $P(y \mid q) = \sum_{C \subset G} P(y \mid q, C)\, P(C \mid q, G)$, where $P(C \mid q, G)$ represents the retrieval probability over the graph KB, and $P(y \mid q, C)$ denotes the generation probability conditioned on the integrated evidence.

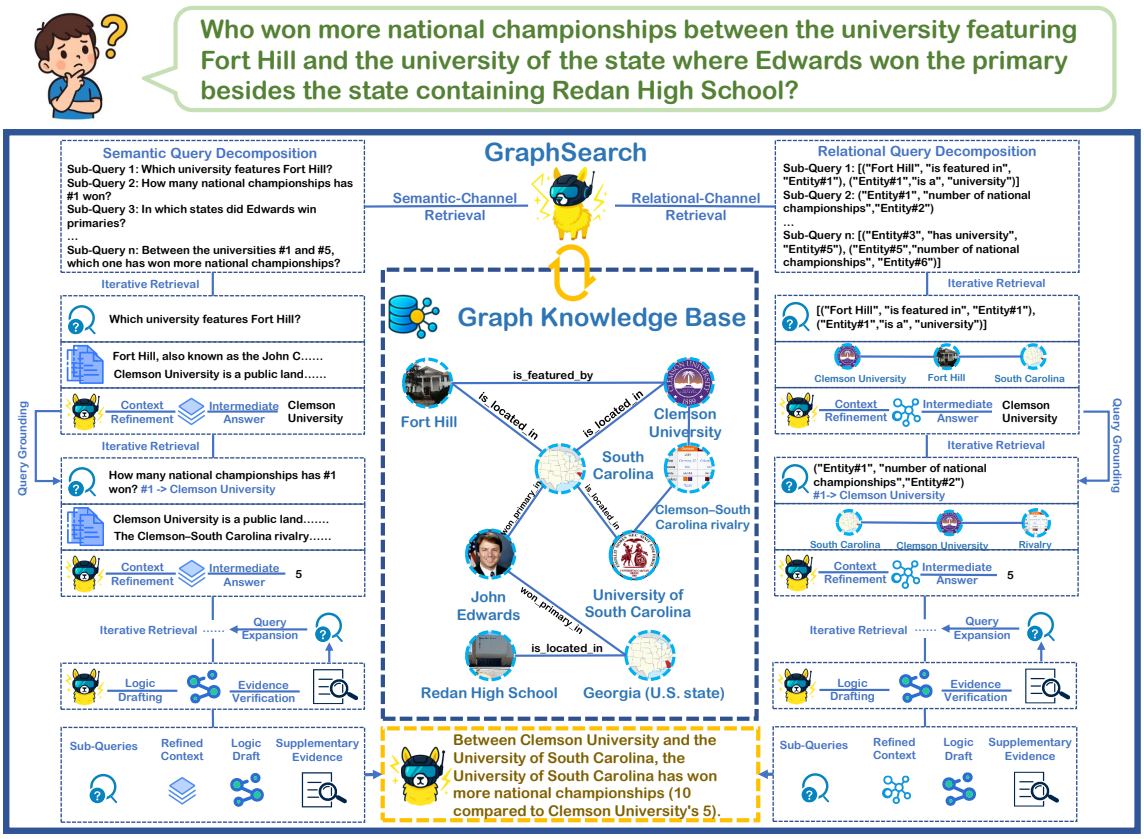

Figure 3: Overview of our GRAPHSEARCH framework. Build upon existing graph KBs, GRAPHSEARCH adopts a dual-channel retrieval strategy by leveraging structural graph evidence through relational queries and chunk-based textual evidence through semantic queries, driven by an agentic searching pipeline driven by coordinated modules.

3 GRAPHSEARCH

The overview of GRAPHSEARCH is shown in Figure 3. We build upon existing GraphRAG methods to construct the graph KB from corpus. On top of this, GRAPHSEARCH leverages the retriever to perform agentic deep searching with enhanced contextual evidence, enabling better answer generation.

3.1 The Modular Deep Searching Pipeline

3.1.1 Iterative Retrieval

Query Decomposition. Given a complex query Q as input, the goal of this module is to decompose Q into a sequence of atomic sub-queries $\{q_1, q_2, \ldots, q_m\} = \mathrm{P_{QD}}(Q)$ prompted by template $\mathrm{P_{QD}}$, each representing a smaller and tractable component of the original question. In practice, each q_i focuses on resolving a single entity, relation, or contextual dependency, thereby enabling the retriever to access fine-grained evidence and reducing the reasoning complexity of the overall task. For each sub-query q_i, the graph KB retriever ψ accesses database G to return

$$C_{q_i} = \psi(q_i \mid G) = \{E_{q_i}, R_{q_i}, K_{q_i}\} \quad (1)$$

where C_{q_i} is the retrieved context of sub-query q_i. The detail of prompt $\mathrm{P_{QD}}$ is in Figure 10.

Context Refinement. Once the initial context C_{q_i} is retrieved for a sub-query q_i, this module aims to refine the evidence by filtering redundant information and highlighting the most relevant entities, relations, and textual chunks. Given that raw retrieval, the refined context is obtained as $C'_{q_i} = \mathrm{P_{CR}}(q_i, C_{q_i})$. This operation ensures that each refined context C'_{q_i} contains only the most informative evidence for answering, thereby improving grounding quality in subsequent reasoning.

Query Grounding. The sub-queries are designed to be semantically independent yet logically ordered, such that the answer to one sub-query can serve as contextual grounding for subsequent ones. In practice, many decomposed queries may contain placeholders or unresolved references that depend on the answers of prior sub-queries. To resolve this, each q_i in $\{q_1, q_2, \ldots, q_m\}$ is first paired with its retrieved context C_{q_i} and produce an intermediate answer $\hat{a}_{q_i} = \mathrm{LLM}(q_i, C_{q_i})$, then progressively accumulated to support later queries. Formally, the

grounded query is expressed as

$$\tilde{q}_i = \mathrm{P}_{\mathrm{QG}}(q_i, \{q_{<i}, C_{q_{<i}}, \hat{a}_{q_{<i}}\}), \quad (2)$$

This procedure guarantees that each $\tilde{q}_i$ is contextually instantiated rather than under-specified, enabling the graph KB retriever to fetch a more relevant context $C_{\tilde{q}_i}$ for subsequent reasoning.

3.1.2 Reflection Routing

Logic Drafting. The role of this module is to organize these pieces into a coherent reasoning chain that outlines how partial answers connect to the original query Q. Specifically, the drafting prompt P_{LD} integrates the sequence of $\{q_i, C_{\tilde{q}_i}, \hat{a}_{q_i}\}$ to produce a structured draft $\mathcal{L}$, where

$$\mathcal{L} = \mathrm{P}_{\mathrm{LD}}(\{\tilde{q}_i, C_{\tilde{q}_i}, \hat{a}_{q_i}\}_{i=1}^{m}). \quad (3)$$

During this drafting process, the module not only consolidates available evidence but also exposes potential gaps in the reasoning chain. For instance, if a sub-query relies on entities or relations that were not retrieved in earlier steps, or if the accumulated sub-queries with intermediate answers $\{\tilde{q}_i, \hat{a}_{q_i}\}$ form an inconsistent chain, such deficiencies are explicitly reflected in $\mathcal{L}$ and exposed.

Evidence Verification. This module evaluates whether the accumulated evidence in $\mathcal{L}$ is sufficient and logically consistent to support a final answer. The verification prompt P_{EV} inspects both the retrieved contexts and the intermediate answers, checking for factual grounding, coherence, and potential contradictions, formally described as

$$\mathcal{V} = \mathrm{P}_{\mathrm{EV}}(\{\tilde{q}_i, C_{\tilde{q}_i}, \hat{a}_{q_i}\}_{i=1}^{m}, \mathcal{L}), \quad (4)$$

where $\mathcal{V} \in \{\mathrm{Accept}, \mathrm{Reject}\}$ denotes the verification decision, the former implying that the reasoning chain is logically reliable, and the latter indicating missing or inconsistent evidence.

Query Expansion. This module generates additional sub-queries that explicitly target the missing evidence. Formally, using the expansion prompt and outputs a set of expanded sub-queries

$$\{q_j^+\}_{j=1}^{n} = \mathrm{P}_{\mathrm{QE}}(\{\tilde{q}_i, C_{\tilde{q}_i}, \hat{a}_{q_i}\}_{i=1}^{m}, \mathcal{L}). \quad (5)$$

Each expanded sub-query q_j^+ is submitted to the retriever ψ, yielding supplementary evidence $C_{q_i^+} = \psi(q_i^+ \mid G) = \{E_{q_i^+}, R_{q_i^+}, K_{q_i^+}\}$. The additional contexts $C_{q_i^+}$ are appended, thereby enriching the evidence pool and ensuring that knowledge gaps revealed in $\mathcal{L}$ can be actively filled, leading to a more reliable reasoning process.

3.2 Dual-Channel Retrieval

Semantic Queries. The semantic channel emphasizes retrieving descriptive evidence from chunk-level text. Given a complex query such as *"How many times did plague occur in the place where the creator of The Worship of Venus died?"*, the retriever first reformulates it into a sequence of semantically coherent sub-queries $\{q_1^{(s)}, q_2^{(s)}, \ldots, q_m^{(s)}\}$. Each $q_i^{(s)}$ is resolved against the text corpus as $C_{q_i^{(s)}} = \{K_{q_i^{(s)}}\}$, focusing on a single factual aspect, such as identifying the creator of the artwork, locating the place where this creator died, and finally retrieving records about the frequency of plague occurrences in that place. This design allows the semantic channel to capture nuanced descriptive information scattered across the corpus, ensuring that the retrieved textual evidence provides sufficient coverage for reasoning.

Relational Queries. The relational channel formulates the problem directly in terms of structured triples. Given a complex query, it is decomposed into relational sub-queries $\{q_1^{(r)}, q_2^{(r)}, \ldots, q_n^{(r)}\}$ mapping into subject-predicate-object relations. For each $q_j^{(r)}$, the retriever returns a subgraph context $C_{q_j^{(r)}} = \{E_{q_j^{(r)}}, R_{q_j^{(r)}}\}$, focusing on entities and relations. For example, the painting *The Worship of Venus* $\rightarrow$ its creator $\rightarrow$ place of death $\rightarrow$ plague occurrences. Unresolved references (e.g., $\mathrm{Entity}\#1, \mathrm{Entity}\#2$) are progressively instantiated once upstream triples are resolved. This explicit traversal enforces logical dependencies, enabling the retriever to surface subgraphs that directly encode the answer path.

4 Experiments

4.1 Experimental Setup

Datasets and Baselines. We conducted experiments on six multi-hop QA benchmarks within the RAG setting. The **Wikipedia**-based benchmarks include *HotpotQA*, *MuSiQue*, and *2WikiMultiHopQA* following (Gutiérrez et al., 2025; Yang et al., 2025). The **Domain**-based benchmarks (Qian et al., 2025) incorporate multi-hop questions synthesized by (Luo et al., 2025), covering fields like *Medical*, *Agriculture*, and *Legal*. We compare GRAPHSEARCH with *Vanilla LLM*, *Naive RAG*, *GraphRAG*, *LightRAG*, *MiniRAG*, *PathRAG*, *HippoRAG2*, and *HyperGraphRAG*. More details are provided in the Appendix A and B.

Table 1: Experiment results of reasoning performance across six multi-hop QA benchmarks covering Wikipedia-based and Domain-based datasets. The + means GRAPHSEARCH integrates with various graph KB retrievers built upon the corresponding GraphRAG methods. The backbone LLM is *Qwen2.5-32B-Instruct*.

Method	HotpotQA			MuSiQue			2WikiMultiHopQA		
	SubEM	A-Score	E-Score	SubEM	A-Score	E-Score	SubEM	A-Score	E-Score
Vanilla LLM	33.67	6.90	5.98	12.33	6.10	5.87	48.33	6.95	4.50
Naive RAG	72.00	8.88	9.04	40.00	7.21	8.18	72.33	7.93	8.03
GraphRAG Baselines									
GraphRAG	72.67	8.18	8.65	36.67	6.58	7.32	79.33	7.44	7.99
LightRAG	73.00	8.30	8.66	35.00	6.50	7.28	81.67	7.62	7.94
MiniRAG	68.00	7.95	8.24	41.00	6.93	7.67	74.00	7.57	7.61
PathRAG	79.00	8.99	9.17	46.33	7.26	8.02	77.00	8.25	8.34
HippoRAG2	76.67	8.45	8.73	44.00	7.07	7.88	72.33	7.98	8.01
HyperGraphRAG	74.33	7.39	8.69	41.67	6.76	7.53	64.00	7.62	7.80
GRAPHSEARCH									
+ LightRAG	79.00	9.21	**9.46**	51.00	7.72	8.38	85.00	9.21	9.12
+ PathRAG	**82.00**	**9.24**	9.42	**55.33**	**7.83**	**8.48**	**88.67**	**9.32**	**9.29**
+ HyperGraphRAG	80.33	9.19	9.35	49.33	7.73	8.22	83.33	8.84	8.75

Method	Medicine			Agriculture			Legal		
	SubEM	A-Score	E-Score	SubEM	A-Score	E-Score	SubEM	A-Score	E-Score
Vanilla LLM	21.29	7.14	7.57	29.88	7.10	7.38	37.11	7.02	7.43
Naive RAG	54.34	8.23	8.67	54.24	7.91	8.26	53.36	7.37	7.67
GraphRAG Baselines									
GraphRAG	53.32	7.59	7.98	57.81	7.84	7.66	58.98	7.57	7.23
LightRAG	49.80	7.36	7.57	55.66	7.38	7.32	56.84	7.01	6.78
MiniRAG	56.84	8.13	8.51	59.38	8.08	8.08	61.91	7.70	7.50
PathRAG	58.79	8.18	8.32	61.13	8.22	8.23	62.30	7.96	7.91
HippoRAG2	55.08	7.90	8.03	58.20	7.95	7.86	64.45	8.02	7.81
HyperGraphRAG	62.11	8.39	8.70	63.67	8.35	8.49	66.60	8.18	8.18
GRAPHSEARCH									
+ LightRAG	65.88	8.61	8.80	63.53	8.52	8.48	71.68	8.45	8.52
+ PathRAG	70.12	8.59	8.82	69.34	8.63	8.78	74.41	8.32	8.49
+ HyperGraphRAG	**73.24**	**8.87**	**9.24**	**73.83**	**8.93**	**9.02**	**78.52**	**8.76**	**8.83**

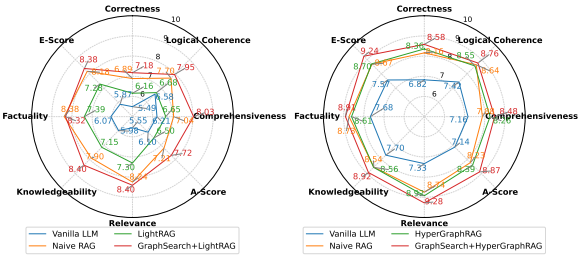

Figure 4: The performance yield by GRAPHSEARCH over GraphRAG on eight metrics of answer generation.

Evaluation Metrics. We adopt three evaluation metrics to assess the QA and retrieval quality. The string-based Substring Exact-Match (SubEM) metric checks whether the golden answer is explicitly contained in the response. The Answer-Score (A-Score) covers Correctness, Logical Coherence, and Comprehensiveness. The Evidence-Score (E-Score) measures Relevance, Knowledgeability, and Factuality. Both A-Score and E-Score are assessed using the LLM-as-a-Judge (Gu et al., 2024). More details are provided in the Appendix D.

4.2 Main Results

GRAPHSEARCH outperforms all GraphRAG baselines. As shown in Table 1, comparing with traditional GraphRAG methods that perform only a single round of graph retrieval and generation, GRAPHSEARCH leverages the constructed graph KBs with retriever to enable multi-turn interactions, and *achieves the best overall performance*. This confirms the importance of adopting an agentic deep searching workflow over GraphRAG in complex reasoning, effectively mitigating the insufficiencies of limited interaction and inadequate retrieval. Case studies with more detail information of are provided in Figure 12 and 13 in Appendix G.

GRAPHSEARCH exhibits strong plug-and-play capability. As shown in Table 1, when applied with various retrievers over different graph KBs, GRAPHSEARCH consistently yields improvements compared to their native interaction schemes. For example, it boosts LightRAG on MuSiQue, raising

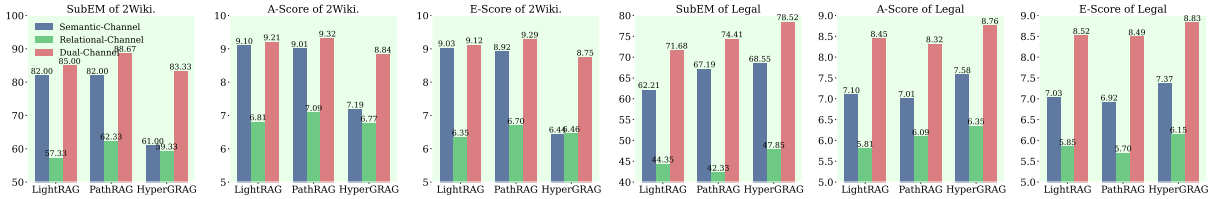

Figure 5: Comparisons among semantic-channel, relational-channel and dual-channel retrieval in GRAPHSEARCH, integrated with the graph knowledge base retrievers built upon LightRAG, PathRAG and HyperGraphRAG.

Table 2: Results across *2Wiki.* and *Legal* benchmarks. The backbone model is *Qwen2.5-7B-Instruct*.

Method	2Wiki.			Legal		
	SubEM	A-S	R-S	SubEM	A-S	R-S
Vanilla LLM	46.67	6.26	3.70	34.18	6.47	6.89
Naive RAG	62.33	7.37	7.41	52.58	6.71	7.29
GraphRAG Baselines						
LightRAG	72.33	7.11	7.53	52.93	6.50	6.45
PathRAG	73.00	7.44	7.71	58.98	7.06	7.01
HyperGraphRAG	72.33	7.49	7.69	60.11	7.32	7.19
GRAPHSEARCH						
+ LightRAG	79.00	8.35	8.21	58.59	7.64	7.31
+ PathRAG	82.00	**8.51**	8.59	64.32	7.87	7.66
+ HyperGraphRAG	**82.33**	8.49	**8.69**	**67.48**	**8.02**	**7.39**

SubEM from **35.00** to **51.00**, while improving A-Score and E-Score from **6.50** and **7.28** to **7.72** and **8.38**. Similarly, it enhances HyperGraphRAG on Medicine, increasing SubEM from **62.11** to **73.24**, and further elevating A-Score and E-Score from **8.39** and **8.70** to **8.87** and **9.24**. These results demonstrate the plug-and-play capability of GRAPHSEARCH as presented in Figure 4.

4.3 Ablation Studies

GRAPHSEARCH still remains effective under reduced model size. Using *Qwen2.5-7B-Instruct* as the backbone, the experimental results on the *2Wiki.* and *Legal* datasets are reported in Table 2. Compared to three GraphRAG baselines, GRAPHSEARCH built upon these graph KB retrievers consistently achieves performance improvements. This confirms the potential of GRAPHSEARCH to extend effectively to models with reduced size.

GRAPHSEARCH benefits from the design of dual-channel retrieval. As shown in Figure 5, the QA performance on the *2Wiki* and *Legal* datasets obtained by integrating retrieval contexts from both channels consistently surpasses that of either single-channel variant across all graph KB retrievers. A relative improvement is particularly pronounced on the *Legal* dataset, which confirms *the necessity of the design of dual-channel retrieval*,

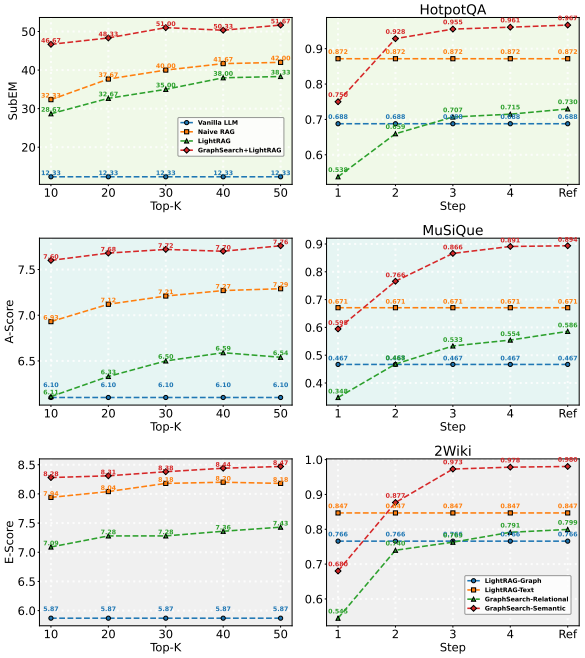

Figure 6: (a) Performance changes as the count of *Top-K* varies during the retrieval stage. (b) GRAPHSEARCH improves the *recall* rate of golden evidence during agentic multi-turn interactions progressively.

fully leveraging the constructed graph KBs from both semantic and structural perspectives.

GRAPHSEARCH modules make clear contributions to the agentic deep searching workflow. We evaluate the incremental contributions of the module coordination, including *Query Decomposition (QD)*, *Context Refinement (CR)*, *Query Grounding (QG)*, *Logic Drafting (LD)*, *Evidence Verification (EV)*, and *Query Expansion (QE)* in Table 3. We adopt the graph KB retriever built upon HyperGraphRAG for GRAPHSEARCH along with a baseline. Comparing the combination of [QD, CR] with [QD, CR, QG], the former performs non-iterative question decomposition, producing multiple sub-queries in one go without missing information. The results confirm the value of the modular orchestration in GRAPHSEARCH, each step progressively enhances the reasoning process and enables the agentic deep searching workflow.

Table 3: Experiment results of ablation study on module coordination across *2Wiki.* and *Legal* datasets of GRAPHSEARCH + HyperGraphRAG. ✓and / refer to whether each individual module is enable or not.

Modules						2Wiki.			Legal		
QD	CR	QG	LD	EV	QE	SubEM	A-Score	R-Score	SubEM	A-Score	R-Score
GRAPHSEARCH + HyperGraphRAG											
/	/	/	/	/	/	64.00	7.62	7.80	66.60	8.18	8.18
✓	✓	/	/	/	/	76.33	8.14	8.16	73.98	8.34	8.29
✓	✓	✓	/	/	/	81.67	8.57	8.57	77.31	**8.82**	8.71
✓	✓	✓	✓	✓	✓	**83.33**	**8.84**	**8.75**	**78.52**	8.76	**8.83**

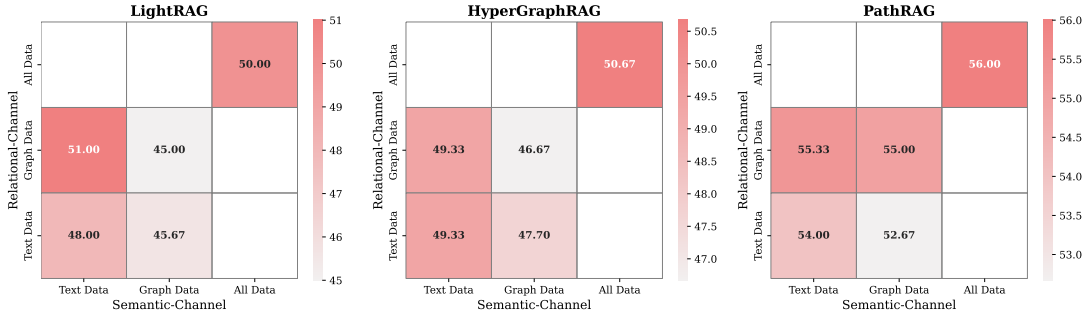

Figure 7: GRAPHSEARCH demonstrates a modality–function alignment property by constraining the retrieval source from textual data, structural graph data or both for the relational-channel and semantic-channel.

GRAPHSEARCH exhibits more pronounced advantages under smaller retrieval budgets. By varying the *Top-K* from 10 to 50 as a adjustment for retrieval overhead, the comparison of GRAPHSEARCH on MuSiQue is shown in Figure 6(a). As Top-K decreases, both Naive RAG and LightRAG show a sharp decline, indicating that their retrievers fail to capture part of the golden evidence under reduced budgets, preventing models from engaging in sufficient evidence-grounded reasoning. By contrast, the agentic searching workflow in GRAPHSEARCH *sustains its performance advantages even under low retrieval overhead* with fewer contextual information, which determines to refine its querying strategy to adapt the weak retriever.

4.4 Further Analysis: Deep Integration of GRAPHSEARCH with Graph KBs

GRAPHSEARCH improves the retrieval quality through the agentic interaction progressively. Using *Recall* to calculate the golden evidence captured, we compare the retrieval quality of GRAPHSEARCH with LightRAG in Figure 6(b). The *Step* denotes the interaction rounds performed by GRAPHSEARCH, up to the final self-reflection stage. It initially retrieves fewer pieces of golden evidence, as it decomposes complex queries into atomic sub-queries. As interactions proceed, the recall shows substantial improvement across both channels, which confirms that the agentic workflow is tightly integrated with the features of graph KBs.

GRAPHSEARCH achieves clear gains comparing with agentic RAG baselines. We compare GRAPHSEARCH with representative agentic RAG methods, including ReAct (Yao et al., 2023), IR-CoT (Trivedi et al., 2023), and Search-o1 (Li et al., 2025). These baselines operate where retrieval is performed over chunk-based textual source. We match the Top-K of retrieval context and adopt the same dense embedder as used in GRAPHSEARCH. As shown in Table 4, GRAPHSEARCH consistently outperforms these agentic RAG baselines across Medicine, Agriculture, and Legal. The results highlight GRAPHSEARCH's strong applicability to graph KBs by utilizing structural signals for evidence discovery and aggregation.

GRAPHSEARCH demonstrates a modality–functionality alignment property. We calculate SubEM on MuSiQue by replacing the retrieval sources of the semantic and relational channel with text and graph data respectively. Results obtained by retrieving from the full data are included as references. Figure 7 shows that using semantic queries to access text data and relational queries to access graph data consistently outper-

Table 4: Comparison between GRAPHSEARCH with agentic RAG methods using *Qwen2.5-32B-Instruct*.

Method	Medicine	Agriculture	Legal
	SubEM	SubEM	SubEM
Vanilla LLM	21.29	29.88	37.11
Naive RAG	54.34	54.24	53.36
ReAct	19.73	25.99	30.86
IRCoT	52.78	51.47	48.12
Search-o1	59.42	62.17	56.90
GRAPHSEARCH	**73.24**	**73.83**	**78.52**

forms other combinations. Moreover, compared to retrieving from the full data, restricting each channel to its aligned modality not only achieves comparable performance but also substantially reduces context overhead. It confirms that the functionality of the dual-channel retrieval strategy aligns with the data modalities of graph KBs.

GRAPHSEARCH balances reasoning performance and execution latency. We further analyze the efficiency of GRAPHSEARCH by comparing its run time and model API calls with agentic RAG strategies on the *Legal* dataset. As illustrated in Figure 8, GRAPHSEARCH incurs a higher number of model API calls, however, unlike streaming agentic RAG methods that interleave retrieval in a largely sequential manner, the dual-channel retrieval design of GRAPHSEARCH is parallel, enabling semantic retrieval and relational retrieval to be executed concurrently. As a result, GRAPHSEARCH achieves stronger performance while maintaining competitive overall run time overhead. These results demonstrate that GRAPHSEARCH strikes a favorable balance between reasoning capability and execution efficiency.

5 Related Work

5.1 Graph Retrieval-Augmented Generation

Building on RAG, GraphRAG explicitly models structural relations among entities, thereby capturing contextual dependencies and structural knowledge integration (Peng et al., 2024; Edge et al., 2024). Early work (Sarthi et al., 2024; Edge et al., 2024) emphasize hierarchical summarization and integration. LightRAG (Guo et al., 2024) advanced this direction by incorporating graph structures into both indexing and retrieval. Recent efforts in graph KB construction introduce diverse structural representations, such as heterogeneous and lightweight graph structures (Fan et al., 2025; Xu et al., 2025), the extension to hypergraphs that capturing higher-order relational dependencies (Luo et al., 2025; Feng et al., 2025), and the leverage of causal graphs to improve logical continuity (Wang et al., 2025). Retrieval strategies increasingly rely on heuristic path exploration, such as the topology-enhanced lightweight search (Fan et al., 2025), the pruning via relational path retrieval (Chen et al., 2025), the utilization of personalized memory-inspired reasoning (Jimenez Gutierrez et al., 2024; Gutiérrez et al., 2025), and the adoption of beam search over proposition paths (Wang and Han, 2025).

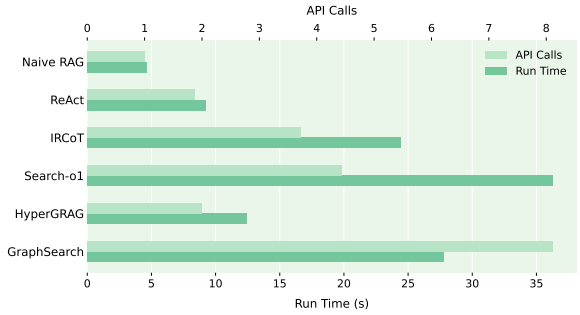

Figure 8: The balance between reasoning capability and execution efficiency of GRAPHSEARCH.

5.2 Agentic Retrieval-Augmented Generation

Single-round interaction of RAG is insufficient for complex reasoning tasks, and modular RAG systems (Gao et al., 2024; Jin et al., 2025b; Wu et al., 2025) are proposed to flexibly reconfigure retrieval and reasoning modules into composable pipelines. Recently, agentic approaches emerged, where representative methods include reasoning–acting synergy in ReAct (Yao et al., 2023), self-reflective retrieval in Self-RAG (Asai et al., 2024), test-time planning in PlanRAG (Verma et al., 2024), and reinforcement-learned search agents in Search-o1 (Li et al., 2025) and Search-R1 (Jin et al., 2025a). Pioneering works (Sun et al., 2023; Ma et al., 2024; Shen et al., 2024; Lee et al., 2024) integrated graph knowledge for retrieval into the agentic RAG workflow to support the multi-step reasoning.

6 Conclusion

We introduced GRAPHSEARCH, a novel agentic deep searching framework for GraphRAG. By integrating a dual-channel retrieval strategy, it overcomes the limitations of shallow retrieval and inefficient graph utilization. Its modular design enables iterative reasoning, leading to strong evidence aggregation. Experimental results demonstrate consistent improvements in retrieval and generation, highlighting the effectiveness of our approach.

Limitations

Although GRAPHSEARCH has made progress in advancing GRAPHRAG, there are still some limitations. First, it remains uncertain whether GRAPHSEARCH can unlock greater potential under different training strategies, such as fine-tuning or reinforcement learning. Second, how to integrate it with cutting-edge reasoning models is still an open question. Finally, applying GRAPHSEARCH to scenarios involving multimodal corpora is a direction worthy of further investigation.

References

Akari Asai, Zeqiu Wu, Yizhong Wang, Avi Sil, and Hannaneh Hajishirzi. 2024. Self-rag: Learning to retrieve, generate, and critique through self-reflection. In *International Conference on Learning Representations*.

Jinze Bai, Shuai Bai, Yunfei Chu, Zeyu Cui, Kai Dang, Xiaodong Deng, Yang Fan, Wenbin Ge, Yu Han, Fei Huang, and 1 others. 2023. Qwen technical report. *arXiv preprint arXiv:2309.16609*.

Boyu Chen, Zirui Guo, Zidan Yang, Yuluo Chen, Junze Chen, Zhenghao Liu, Chuan Shi, and Cheng Yang. 2025. Pathrag: Pruning graph-based retrieval augmented generation with relational paths. *arXiv preprint arXiv:2502.14902*.

Darren Edge, Ha Trinh, Newman Cheng, Joshua Bradley, Alex Chao, Apurva Mody, Steven Truitt, Dasha Metropolitansky, Robert Osazuwa Ness, and Jonathan Larson. 2024. From local to global: A graph rag approach to query-focused summarization. *arXiv preprint arXiv:2404.16130*.

Tianyu Fan, Jingyuan Wang, Xubin Ren, and Chao Huang. 2025. Minirag: Towards extremely simple retrieval-augmented generation. *arXiv preprint arXiv:2501.06713*.

Yifan Feng, Hao Hu, Xingliang Hou, Shiquan Liu, Shihui Ying, Shaoyi Du, Han Hu, and Yue Gao. 2025. Hyper-rag: Combating llm hallucinations using hypergraph-driven retrieval-augmented generation. *arXiv preprint arXiv:2504.08758*.

Yunfan Gao, Yun Xiong, Meng Wang, and Haofen Wang. 2024. Modular rag: Transforming rag systems into lego-like reconfigurable frameworks. *arXiv preprint arXiv:2407.21059*.

Jiawei Gu, Xuhui Jiang, Zhichao Shi, Hexiang Tan, Xuehao Zhai, Chengjin Xu, Wei Li, Yinghan Shen, Shengjie Ma, Honghao Liu, and 1 others. 2024. A survey on llm-as-a-judge. *arXiv preprint arXiv:2411.15594*.

Zirui Guo, Lianghao Xia, Yanhua Yu, Tu Ao, and Chao Huang. 2024. Lightrag: Simple and fast retrieval-augmented generation. *arXiv preprint arXiv:2410.05779*.

Bernal Jiménez Gutiérrez, Yiheng Shu, Weijian Qi, Sizhe Zhou, and Yu Su. 2025. From rag to memory: Non-parametric continual learning for large language models. *arXiv preprint arXiv:2502.14802*.

Xanh Ho, Anh-Khoa Duong Nguyen, Saku Sugawara, and Akiko Aizawa. 2020. Constructing a multi-hop qa dataset for comprehensive evaluation of reasoning steps. *arXiv preprint arXiv:2011.01060*.

Bernal Jimenez Gutierrez, Yiheng Shu, Yu Gu, Michihiro Yasunaga, and Yu Su. 2024. Hipporag: Neurobiologically inspired long-term memory for large language models. *Advances in Neural Information Processing Systems*, 37:59532–59569.

Bowen Jin, Hansi Zeng, Zhenrui Yue, Jinsung Yoon, Sercan Arik, Dong Wang, Hamed Zamani, and Jiawei Han. 2025a. Search-r1: Training llms to reason and leverage search engines with reinforcement learning. *arXiv preprint arXiv:2503.09516*.

Jiajie Jin, Yutao Zhu, Zhicheng Dou, Guanting Dong, Xinyu Yang, Chenghao Zhang, Tong Zhao, Zhao Yang, and Ji-Rong Wen. 2025b. Flashrag: A modular toolkit for efficient retrieval-augmented generation research. In *Companion Proceedings of the ACM on Web Conference 2025*, pages 737–740.

Meng-Chieh Lee, Qi Zhu, Costas Mavromatis, Zhen Han, Soji Adeshina, Vassilis N Ioannidis, Huzefa Rangwala, and Christos Faloutsos. 2024. Hybgrag: Hybrid retrieval-augmented generation on textual and relational knowledge bases. *arXiv preprint arXiv:2412.16311*.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, and 1 others. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in neural information processing systems*, 33:9459–9474.

Xiaoxi Li, Guanting Dong, Jiajie Jin, Yuyao Zhang, Yujia Zhou, Yutao Zhu, Peitian Zhang, and Zhicheng Dou. 2025. Search-o1: Agentic search-enhanced large reasoning models. *arXiv preprint arXiv:2501.05366*.

Haoran Luo, Guanting Chen, Yandan Zheng, Xiaobao Wu, Yikai Guo, Qika Lin, Yu Feng, Zemin Kuang, Meina Song, Yifan Zhu, and 1 others. 2025. Hypergraphrag: Retrieval-augmented generation via hypergraph-structured knowledge representation. *arXiv preprint arXiv:2503.21322*.

Shengjie Ma, Chengjin Xu, Xuhui Jiang, Muzhi Li, Huaren Qu, Cehao Yang, Jiaxin Mao, and Jian Guo. 2024. Think-on-graph 2.0: Deep and faithful large language model reasoning with knowledge-guided

retrieval augmented generation. *arXiv preprint arXiv:2407.10805*.

John William McEvoy, Cian P McCarthy, Rosa Maria Bruno, Sofie Brouwers, Michelle D Canavan, Claudio Ceconi, Ruxandra Maria Christodorescu, Stella S Daskalopoulou, Charles J Ferro, Eva Gerdts, and 1 others. 2024. 2024 esc guidelines for the management of elevated blood pressure and hypertension: Developed by the task force on the management of elevated blood pressure and hypertension of the european society of cardiology (esc) and endorsed by the european society of endocrinology (ese) and the european stroke organisation (eso). *European heart journal*, 45(38):3912–4018.

Humza Naveed, Asad Ullah Khan, Shi Qiu, Muhammad Saqib, Saeed Anwar, Muhammad Usman, Naveed Akhtar, Nick Barnes, and Ajmal Mian. 2025. A comprehensive overview of large language models. *ACM Transactions on Intelligent Systems and Technology*, 16(5):1–72.

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*.

Hongjin Qian, Zheng Liu, Peitian Zhang, Kelong Mao, Defu Lian, Zhicheng Dou, and Tiejun Huang. 2025. Memorag: Boosting long context processing with global memory-enhanced retrieval augmentation. In *Proceedings of the ACM on Web Conference 2025*, pages 2366–2377.

Parth Sarthi, Salman Abdullah, Aditi Tuli, Shubh Khanna, Anna Goldie, and Christopher D Manning. 2024. Raptor: Recursive abstractive processing for tree-organized retrieval. In *The Twelfth International Conference on Learning Representations*.

Zhili Shen, Chenxin Diao, Pavlos Vougiouklis, Pascual Merita, Shriram Piramanayagam, Enting Chen, Damien Graux, Andre Melo, Ruofei Lai, Zeren Jiang, and 1 others. 2024. Gear: Graph-enhanced agent for retrieval-augmented generation. *arXiv preprint arXiv:2412.18431*.

Saba Sturua, Isabelle Mohr, Mohammad Kalim Akram, Michael Günther, Bo Wang, Markus Krimmel, Feng Wang, Georgios Mastrapas, Andreas Koukounas, Andreas Koukounas, Nan Wang, and Han Xiao. 2024. jina-embeddings-v3: Multilingual embeddings with task lora. *Preprint*, arXiv:2409.10173.

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

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. 2022. Musique: Multi-hop questions via single-hop question composition. *Transactions of the Association for Computational Linguistics*, 10:539–554.

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. 2023. Interleaving retrieval with chain-of-thought reasoning for knowledge-intensive multi-step questions. In *Proceedings of the 61st annual meeting of the association for computational linguistics (volume 1: long papers)*, pages 10014–10037.

Prakhar Verma, Sukruta Prakash Midigeshi, Gaurav Sinha, Arno Solin, Nagarajan Natarajan, and Amit Sharma. 2024. Plan* rag: Efficient test-time planning for retrieval augmented generation. *arXiv preprint arXiv:2410.20753*.

Jingjin Wang and Jiawei Han. 2025. Proprag: Guiding retrieval with beam search over proposition paths. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 6223–6238.

Nengbo Wang, Xiaotian Han, Jagdip Singh, Jing Ma, and Vipin Chaudhary. 2025. Causalrag: Integrating causal graphs into retrieval-augmented generation. *arXiv preprint arXiv:2503.19878*.

Ruofan Wu, Youngwon Lee, Fan Shu, Danmei Xu, Seung-won Hwang, Zhewei Yao, Yuxiong He, and Feng Yan. 2025. Composerag: A modular and composable rag for corpus-grounded multi-hop question answering. *arXiv preprint arXiv:2506.00232*.

Tianyang Xu, Haojie Zheng, Chengze Li, Haoxiang Chen, Yixin Liu, Ruoxi Chen, and Lichao Sun. 2025. Noderag: Structuring graph-based rag with heterogeneous nodes. *arXiv preprint arXiv:2504.11544*.

Cehao Yang, Xueyuan Lin, Chengjin Xu, Xuhui Jiang, Shengjie Ma, Aofan Liu, Hui Xiong, and Jian Guo. 2025. Longfaith: Enhancing long-context reasoning in llms with faithful synthetic data. *arXiv preprint arXiv:2502.12583*.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W Cohen, Ruslan Salakhutdinov, and Christopher D Manning. 2018. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. *arXiv preprint arXiv:1809.09600*.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. 2023. React: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations (ICLR)*.

Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, and 1 others. 2023. A survey of large language models. *arXiv preprint arXiv:2303.18223*, 1(2).

Appendix

A Datasets

As shown in Table 5, we randomly sample 300 questions for HotpotQA, MuSiQue and 2WikiMultiHopQA datasets, and directly adopt the Medicine, Agriculture and Legal datasets from (Luo et al., 2025).

B Baselines

- **Vanilla LLM**: Zero-shot question and answering without any external retrieval source, depending on language model's parametric knowledge.
- **Naive RAG** (Lewis et al., 2020): Generation with plain text chunk-based embedding database as external retrieval source, where top-k items are retrieved for a single round.
- **GraphRAG** (Edge et al., 2024): A graph-based approach to question answering over hierarchical graph index where community summary is generated to represent the relationships.
- **LightRAG** (Guo et al., 2024): A simple and fast GraphRAG framework by applying integration of graph structures with vector representations for a dual-level retrieval system.
- **MiniRAG** (Fan et al., 2025): A novel GraphRAG system designed for small LLM which adopts a lightweight topology-enhanced retrieval approach.
- **PathRAG** (Chen et al., 2025): A GraphRAG system which retrieves key relational paths from the indexing graph through flow-based pruning.
- **HippoRAG2** (Gutiérrez et al., 2025): A RAG framework built upon the personalized PageRank with deeper passage integration.
- **HyperGraphRAG** (Luo et al., 2025): A novel hypergraph-based RAG method that represents n-ary relational facts via hyper-edges for retrieval and generation.

C Implementation Details.

We conduct experiments on a Linux server equipped with 8 A100-SXM4-40GB GPUs. The model for graph construction is *Qwen2.5-32B-Instruct*, and the chunk size is 400 tokens. The embedding model for Naive-RAG and GraphRAG is *jinaai/jina-embeddings-v3* (Sturua et al., 2024). For GRAPHSEARCH and baselines, we set the **Hybrid** retrieval mode and set the **Top-K** for retrieval to 30, or use the default configuration if unavailable. The backbone model for generation is *Qwen2.5-7B/32B-Instruct* (Bai et al., 2023). The LLM-as-a-Judge for evaluation is *Qwen-Plus* (Bai et al., 2023) with API available.

D Evaluation Details

We leverage the Substring Extract-Match(**SubEM**) metric to check whether the golden answer is explicitly contained in the response, the Answer-Score(**A-Score**) to judge the quality of model generation across 3 criteria covering **correctness**, **logical coherence** and **comprehensiveness** with the **golden answer** as reference, and the Evidence-Score(**E-Score**) to measure how well the model's generation is grounded in the golden evidence, evaluated along 3 criteria including **relevance**, **knowledgeability** and **factuality** with the **golden evidence** as reference as SubEM $= \frac{1}{N}\sum_{i=1}^{N} \mathbf{1}\left[\text{contains}\left(O_i^{\text{pred}}, A_i^{\text{gold}}\right)\right]$.

Figure 9: Evaluation prompts of **A-Score** across 3 criteria and **E-Score** across 3 criteria.

E The Use of Ai Assistants

During the completion of this thesis, the scenarios involving the use of Ai Assistants included: using code-completion tools to assist with experiments, and using ChatGPT to polish the draft after the initial writing was completed. LLMs were not involved in any aspects such as the development of research ideas, literature review, and so on.

Table 5: Detail information of datasets used in GRAPHSEARCH. The tokenizer used to calculate the size of corpora is GPT-4o. # means the number of counts.

Name	Reference	Source	#Corpus	#Questions	Question Types	#Evidence
HotpotQA	(Yang et al., 2018)	Wikipedia	397,274	300	Comparison, Bridge	2,3,4
MuSiQue	(Trivedi et al., 2022)	Wikipedia	533,145	300	2-Hop, 3-Hop, 4-Hop	2,4
2WikiMultiHopQA	(Ho et al., 2020)	Wikipedia	220,295	300	Compositional, Comparison, Bridge Comparison, Inference	2,4
Medicine	(McEvoy et al., 2024)	ESC Guidelines	175,216	512	1-Hop, 2-Hop, 3-Hop	1,2,3
Agriculture	(Qian et al., 2025)	UltraDomain	378,592	512	1-Hop, 2-Hop, 3-Hop	1,2,3
Legal	(Qian et al., 2025)	UltraDomain	929,396	512	1-Hop, 2-Hop, 3-Hop	1,2,3

F Prompt Templates

As shown in Figure 10, we introduce the prompt templates in **Query Decomposition**, **Context Refinement** and **Query Rewriting** modules both in text-channel and graph-channel.

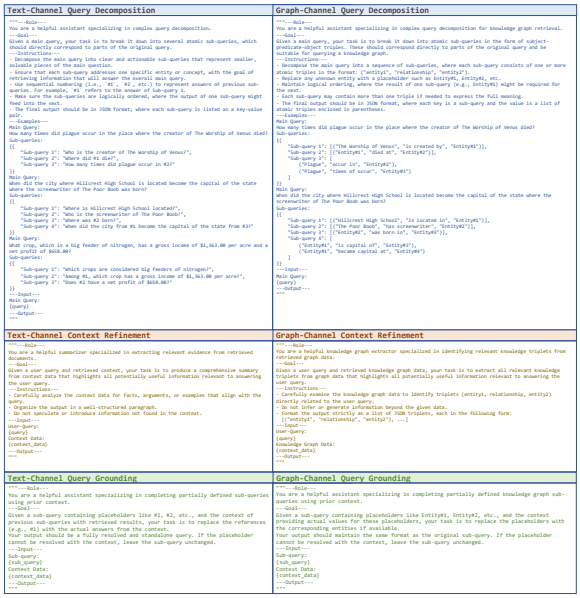

Figure 10: Prompt templates of **Query Decomposition**, **Context Refinement** and **Query Rewriting** modules both in text-channel and graph-channel.

As shown in Figure 11, we introduce the prompt templates in **Logic Drafting**, **Evidence Verification** and **Query Expansion** modules for combining into a reflection router.

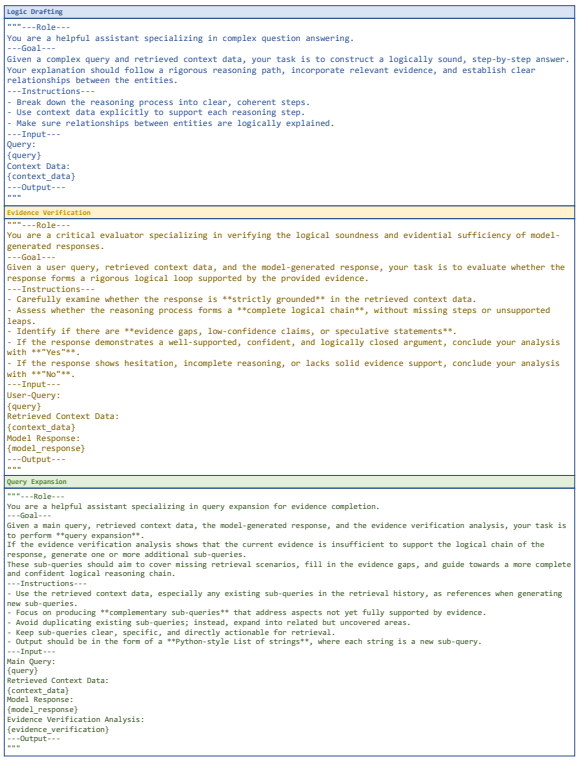

Figure 11: Prompt templates of **Logic Drafting**, **Evidence Verification** and **Query Expansion** modules for combining into a reflection router.

G Case Studies

As shown in Figure 12, there are some cases of baseline methods, including vanilla LLM generation, Naive RAG and LightRAG. A case of our proposed GRAPHSEARCH is in Figure 13.

G.1 Vanilla Generation, Naive RAG and LightRAG

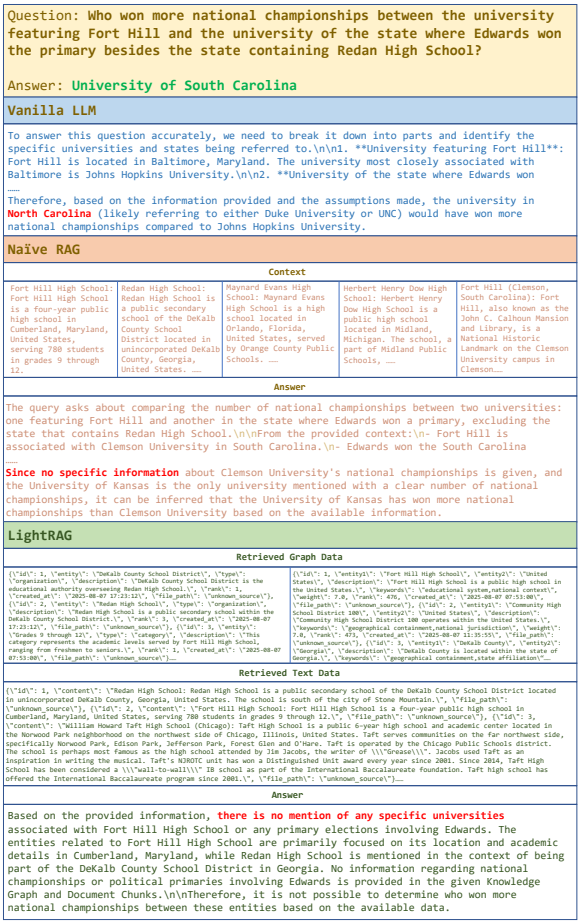

Figure 12: Samples of Vanilla Generation, Naive RAG and LightRAG.

G.2 GRAPHSEARCH with LightRAG

Figure 13: A sample of GRAPHSEARCH with LightRAG as the graph retriever.