Retrieval-Augmented Generation with Hierarchical Knowledge

Anonymous ACL submission

Abstract

Graph-based Retrieval-Augmented Generation (RAG) methods have significantly enhanced the performance of large language models (LLMs) in domain-specific tasks. However, existing RAG methods do not adequately utilize the naturally inherent hierarchical knowledge in human cognition, which limits the capabilities of RAG systems. In this paper, we introduce a new RAG approach, called HiRAG, which utilizes hierarchical knowledge to enhance the semantic understanding and structure capturing capabilities of RAG systems in the indexing and retrieval processes. Our extensive experiments demonstrate that HiRAG achieves significant performance improvements over the state-of-the-art baseline methods.

1 Introduction

Retrieval Augmented Generation (RAG) (Gao et al., 2023) (Lewis et al., 2020) (Fan et al., 2024) has been introduced to enhance the capabilities of LLMs in domain-specific or knowledge-intensive tasks. Naive RAG methods retrieve text chunks that are relevant to a query, which serve as references for LLMs to generate responses, thus helping address the problem of "Hallucination" (Zhang et al., 2023) (Tang and Yang, 2024). However, naive RAG methods usually overlook the relationships among entities in the retrieved text chunks. To address this issue, RAG systems with graph structures were proposed (Edge et al., 2024) (Liang et al., 2024) (Zhang et al., 2025) (Peng et al., 2024a), which construct knowledge graphs (KGs) to model relationships between entities in the input documents. Although existing RAG systems integrating graph structures have demonstrated outstanding performance on various tasks, they still have some serious limitations. GraphRAG (Edge et al., 2024) introduces communities in indexing using the Leiden algorithm (Traag et al., 2019), but the communities only capture the structural proximity of the entities in the KG. KAG (Liang et al., 2024) indexes with a hierarchical representation of information and knowledge, but their hierarchical structure relies too much on manual annotation and requires a lot of human domain knowledge, which renders their method not scalable to general tasks. LightRAG (Guo et al., 2024) utilizes a dual-level retrieval approach to obtain local and global knowledge as the contexts for a query, but it ignores the **knowledge gap** between local and global knowledge, that is, local knowledge represented by the retrieved individual entities (i.e., entity-specific details) may not be semantically related to the global knowledge represented in the retrieved community summaries (i.e., broader, aggregated summaries), as these individual entities may not be a part of the retrieved communities for a query.

We highlight two critical challenges in existing RAG systems that integrate graph structures: **(1) distant structural relationship between semantically similar entities** and **(2) knowledge gap between local and global knowledge**. We illustrate them using a real example from a public dataset, as shown in Figure 1.

Challenge (1) occurs because existing methods over-rely on source documents, often resulting in constructing a knowledge graph (KG) with many entities that are not structurally proximate in the KG even though they share semantically similar attributes. For example, in Figure 1, although the entities "BIG DATA" and "RECOMMENDATION SYSTEM" share semantic relevance under the concept of "DATA MINING", their distant structural relationship in the KG reflects a corpus-driven disconnect. These inconsistencies between semantic relevance and structural proximity are systemic in KGs, undermining their utility in RAG systems where contextual coherence is critical.

Challenge (2) occurs as existing methods (Guo et al., 2024) (Edge et al., 2024) typically retrieve

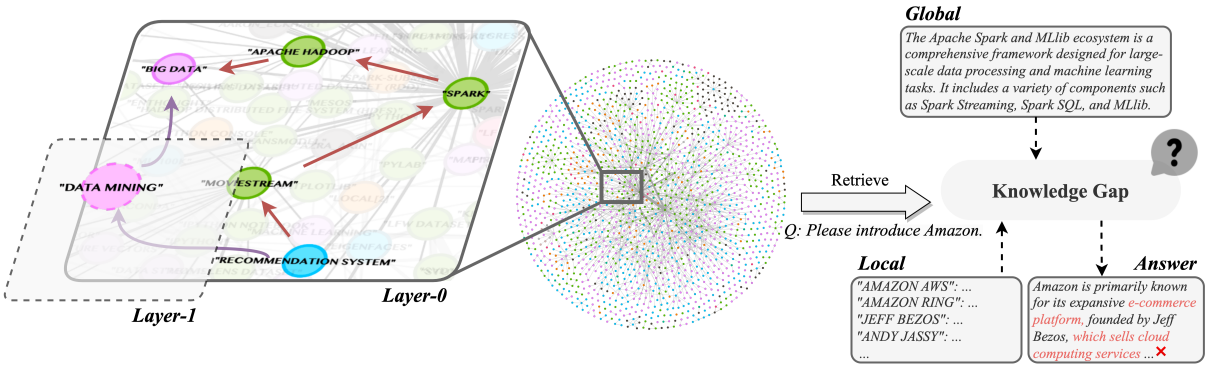

Figure 1: The challenges faced by existing RAG systems: (1) Distant structural relationship between semantically similar entities. (2) Knowledge gap between local and global knowledge.

context either from global or local perspectives but fail to address the inherent disparity between these knowledge layers. Consider the query "Please introduce Amazon" in Figure 1, where global context emphasizes Amazon's involvement in technological domains like big data and cloud computing, but local context retrieves entities directly linked to Amazon (e.g., subsidiaries, leadership). When these two knowledge layers are fed into LLMs as the contexts of a query without contextual alignment, LLMs may struggle to reconcile their distinct scopes, leading to disjointed reasoning, incomplete answers, or even contradictory outputs. For instance, an LLM might conflate Amazon's role as a cloud provider (global) with its e-commerce operations (local), resulting in incoherent or factually inconsistent responses as the red words shown in the case. This underscores the need for new methods that bridge hierarchical knowledge layers to ensure cohesive reasoning in RAG systems.

To address these challenges, we propose **Retrieval-Augmented Generation with Hierarchical Knowledge (HiRAG)**, which integrates hierarchical knowledge into the indexing and retrieval processes. Hierarchical knowledge (Sarrafzadeh and Lank, 2017) is a natural concept in both graph structure and human cognition, yet it has been overlooked in existing approaches. Specifically, to address Challenge (1), we introduce **Indexing with Hierarchical Knowledge (HiIndex)**. Rather than simply constructing a flat KG, we index a KG hierarchically layer by layer. Each entity (or node) in a higher layer summarizes a cluster of entities in the lower layer, which can enhance the connectivity between semantically similar entities. For example, in Figure 1, the inclusion of the summary entity "DATA MINING" strengthens the connection between "BIG DATA" and "RECOMMENDATION SYSTEM". To address Challenge (2), we propose **Retrieval with Hierarchical Knowledge (HiRetrieval)**. HiRetrieval effectively bridges local knowledge of entity descriptions to global knowledge of communities, thus resolving knowledge layer disparities. It provides a three-level context comprising the global level, the bridge level, and the local level knowledge to an LLM, enabling the LLM to generate more comprehensive and precise responses.

In summary, we make the following main contributions:

- We identify and address two critical challenges in graph-based RAG systems: distant structural relationships between semantically similar entities and the knowledge gap between local and global information.
- We propose HiRAG, which introduces unsupervised hierarchical indexing and a novel bridging mechanism for effective knowledge integration, significantly advancing the state-of-the-art in RAG systems.
- Extensive experiments demonstrate both the effectiveness and efficiency of our approach, with comprehensive ablation studies validating the contribution of each component.

2 Related Work

In this section, we discuss recent research concerning graph-augmented LLMs, specifically RAG methods with graph structures. GNN-RAG (Mavromatis and Karypis, 2024) employs GNN-based reasoning to retrieve query-related entities. Then they find the shortest path between the retrieved entities

and candidate answer entities to construct reasoning paths. LightRAG (Guo et al., 2024) integrates a dual-level retrieval method with graph-enhanced text indexing. They also decrease the computational costs and speed up the adjustment process. GRAG (Hu et al., 2024) leverages a soft pruning approach to minimize the influence of irrelevant entities in retrieved subgraphs. It also implements prompt tuning to help LLMs comprehend textual and topological information in subgraphs by incorporating graph soft prompts. StructRAG (Li et al., 2024) identifies the most suitable structure for each task, transforms the initial documents into this organized structure, and subsequently generates responses according to the established structure. Microsoft GraphRAG (Edge et al., 2024) first retrieves related communities and then let the LLM generate the response with the retrieved communities. They also answer a query with global search and local search. KAG (Liang et al., 2024) introduces a professional domain knowledge service framework and employs knowledge alignment using conceptual semantic reasoning to mitigate the noise issue in OpenIE. KAG also constructs domain expert knowledge using human-annotated schemas.

3 Preliminary and Definitions

In this section, we give a general formulation of an RAG system with graph structure referring to the definitions in (Guo et al., 2024) and (Peng et al., 2024b).

In an RAG framework $\mathcal{M}$ as shown in Equation 1, LLM is the generation module, $\mathcal{R}$ represents the retrieval module, φ denotes the graph indexer, and ψ refers to the graph retriever:

$$\mathcal{M} = (LLM, \mathcal{R}(\varphi, \psi)). \quad (1)$$

When we answer a query, the answer we get from an RAG system is represented by a^*, which can be formulated as

$$a^* = arg\max_{a \in A} \mathcal{M}(a|q, \mathcal{G}), \quad (2)$$

$$\mathcal{G} = \varphi(\mathcal{D}) = \{(h, r, t)|h, t \in \mathcal{V}, r \in \mathcal{E}\}, \quad (3)$$

where $\mathcal{M}(a|q, \mathcal{G})$ is the target distribution with a graph retriever $\psi(G|q, \mathcal{G})$ and a generator $LLM(a|q, G)$, and A is the set of possible responses. The graph database $\mathcal{G}$ is constructed from the original external database $\mathcal{D}$. We utilize the total probability formula to decompose $M(a|q, \mathcal{G})$, which can be expressed as

$$\mathcal{M}(a|q, \mathcal{G}) = \sum_{G \in \mathcal{G}} LLM(a|q, G) \cdot \psi(G|q, \mathcal{G}). \quad (4)$$

Most of the time, we only need to retrieve the most relevant subgraph G from the external graph database $\mathcal{G}$. Therefore, here we can approximate $\mathcal{M}(a|q, \mathcal{G})$ as follows:

$$\mathcal{M}(a|q, \mathcal{G}) \approx LLM(a|q, G^*) \cdot \psi(G^*|q, \mathcal{G}), \quad (5)$$

where G^* denotes the optimal subgraph we retrieve from the external graph database $\mathcal{G}$. What we finally want is to get a better generated answer a^*.

4 The HiRAG Framework

HiRAG consists of the two modules, HiIndex and HiRetrieval, as shown in Figure 2. In the HiIndex module, we construct a hierarchical KG with different knowledge granularity in different layers. The summary entities in a higher layer represent more coarse-grained, high-level knowledge but they can enhance the connectivity between semantically similar entities in a lower layer. In the HiRetrieval module, we select the most relevant entities from each retrieved community and find the shortest path to connect them, which serve as the bridge-level knowledge to connect the knowledge at both local and global levels. Then an LLM will generate responses with these three-level knowledge as the context.

4.1 Indexing with Hierarchical Knowledge

In the HiIndex module, we index the input documents as a hierarchical KG. First, we employ the entity-centric triple extraction to construct a basic KG $\mathcal{G}_0$ following (Carta et al., 2023). Specifically, we split the input documents into text chunks with some overlaps. These chunks will be fed into the LLM with well-designed prompts to extract entities $\mathcal{V}_0$ first. Then the LLM will generate relations (or edges) $\mathcal{E}_0$ between pairs of the extracted entities based on the information of the corresponding text chunks. The basic KG can be represented as

$$\mathcal{G}_0 = \{(h, r, t)|h, t \in \mathcal{V}_0, r \in \mathcal{E}_0\}. \quad (6)$$

The basic KG is also the 0-th layer of our hierarchical KG. We denote the set of entities (nodes) in the i-th layer as $\mathcal{L}_i$ where $\mathcal{L}_0 = \mathcal{V}_0$. To construct

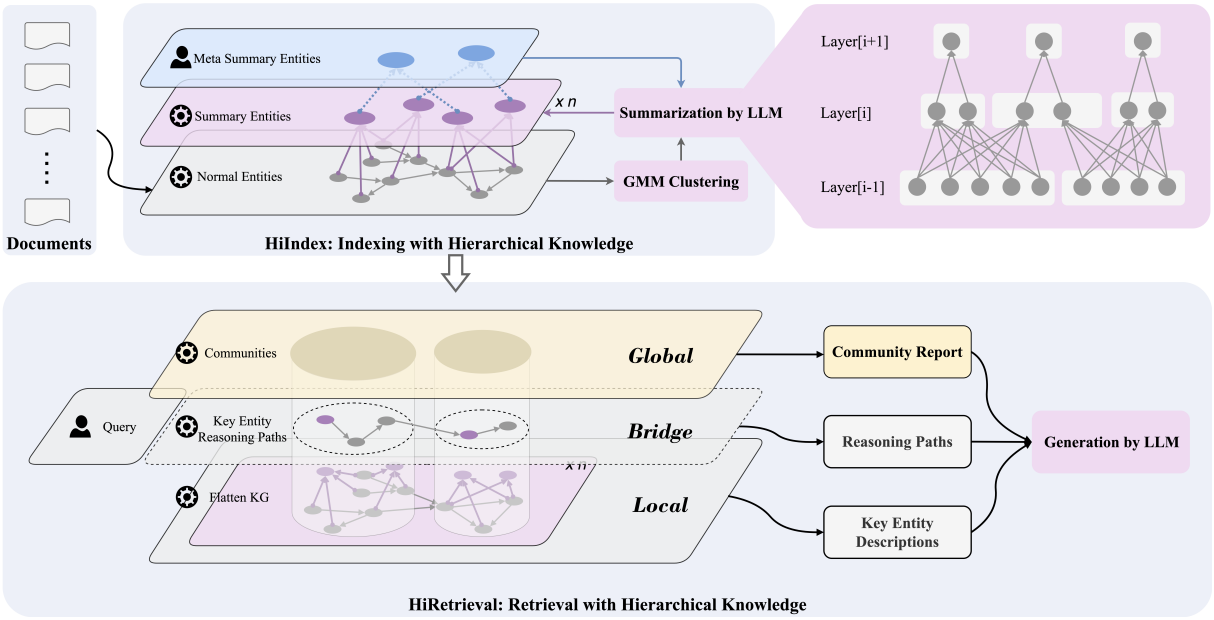

Figure 2: The overall architecture of the HiRAG framework.

the i-th layer of the hierarchical KG, for $i \geq 1$, we first fetch the embeddings of the entities in the $(i-1)$-th layer of the hierarchical KG, which is denoted as

$$\mathcal{Z}_{i-1} = \{Embedding(v) | v \in \mathcal{L}_{i-1}\}, \quad (7)$$

where $Embedding(v)$ is the embedding of an entity v. Then we employ Gaussian Mixture Models (GMMs) to conduct semantical clustering on $\mathcal{L}_{i-1}$ based on $\mathcal{Z}_{i-1}$, following the method described in RAPTOR (Sarthi et al., 2024). We obtain a set of clusters as

$$\mathcal{C}_{i-1} = GMM(\mathcal{L}_{i-1}, \mathcal{Z}_{i-1}) = \{\mathcal{S}_1, \ldots, \mathcal{S}_c\}, \quad (8)$$

where $\forall x, y \in [1, c]$, $\mathcal{S}_x \cap \mathcal{S}_y = \emptyset$ and $\bigcup_{1 \leq x \leq c} \mathcal{S}_x = \mathcal{L}_{i-1}$. After clustering with GMMs, the descriptions of the entities in each cluster in $\mathcal{C}_{i-1}$ are fed into the LLM to generate a set of summary entities for the i-th layer. Thus, the set of summary entities in the i-th layer, i.e., $\mathcal{L}_i$, is the union of the sets of summary entities generated from all clusters in $\mathcal{C}_{i-1}$. Then, we create the relations between entities in $\mathcal{L}_{i-1}$ and entities in $\mathcal{L}_i$, denoted as $\mathcal{E}_{\{i-1,i\}}$, by connecting the entities in each cluster $\mathcal{S} \in \mathcal{C}_{i-1}$ to the corresponding summary entities in $\mathcal{L}_i$ that are generated from the entities in $\mathcal{S}$.

To generate summary entities in $\mathcal{L}_i$, we use a set of meta summary entities $\mathcal{X}$ to guide the LLM to generate the summary entities. Here, $\mathcal{X}$ is a small set of general concepts such as "organization", "person", "location", "event", "technology", etc., that are generated by LLM. For example, the meta summary "technology" could guide the LLM to generate summary entities such as "big data" and "AI". Note that conceptually $\mathcal{X}$ is added as the top layer in Figure 2, but $\mathcal{X}$ is actually not part of the hierarchical KG.

After generating the summary entities and relations in the i-th layer, we update the KG as follows:

$$\mathcal{E}_i = \mathcal{E}_{i-1} \cup \mathcal{E}_{\{i-1,i\}}, \quad (9)$$

$$\mathcal{V}_i = \mathcal{V}_{i-1} \cup \mathcal{L}_i, \quad (10)$$

$$\mathcal{G}_i = \{(h, r, t) | h, t \in \mathcal{V}_i, r \in \mathcal{E}_i\}. \quad (11)$$

We repeat the above process for each layer from the 1st layer to the k-th layer. We will discuss how to choose the parameter k in Section 5. Also note that there is no relation between the summary entities in each layer except the 0-th layer (i.e., the basic KG).

We also employ the Leiden algorithm (Traag et al., 2019) to compute a set of communities $\mathcal{P}$ from the hierarchical KG. Each community may contain entities from multiple layers and an entity may appear in multiple communities. For each community $p \in \mathcal{P}$, we generate an interpretable semantic report using LLMs. Unlike existing methods such as GraphRAG (Edge et al., 2024) and LightRAG (Guo et al., 2024), which identify communities based solely on direct structural proximity in a basic KG, our hierarchical KG introduces multi-resolution semantic aggregation. Higher-layer entities in our KG act as semantic hubs that

abstract clusters of semantically related entities regardless of their distance from each other in a lower layer. For example, while a flat KG might separate "cardiologist" and "neurologist" nodes due to limited direct connections, their hierarchical abstraction as "medical specialists" in upper layers enables joint community membership. The hierarchical structure thus provides dual connectivity enhancement: structural cohesion through localized lower-layer connections and semantic bridging via higher-layer abstractions. This dual mechanism ensures our communities reflect both explicit relational patterns and implicit conceptual relationships, yielding more comprehensive knowledge groupings than structure-only approaches.

4.2 Retrieval with Hierarchical Knowledge

We now discuss how we retrieve hierarchical knowledge to address the knowledge gap issue. Based on the hierarchical KG $\mathcal{G}_k$ constructed in Section 4.1, we retrieve three-level knowledge at both local and global levels, as well as the bridging knowledge that connects them.

To retrieve local-level knowledge, we extract the top-n most relevant entities $\hat{\mathcal{V}}$ as shown in Equation 12, where $Sim(q, v)$ is a function that measures the semantic similarity between a user query q and an entity v in the hierarchical KG $\mathcal{G}_k$. We set n to 20 as default.

$$\hat{\mathcal{V}} = TopN(\{v \in \mathcal{V}_k | Sim(q, v)\}, n). \quad (12)$$

To access global-level knowledge related to a query, we find the communities $\hat{\mathcal{P}} \subset \mathcal{P}$ that are connected to the retrieved entities as described in Equation 13, where $\mathcal{P}$ is computed during indexing in Section 4.1. Then the community reports of these communities are retrieved, which represent coarse-grained knowledge relevant to the user's query.

$$\hat{\mathcal{P}} = \bigcup_{p \in \mathcal{P}} \{p | p \cap \hat{\mathcal{V}} \neq \emptyset\}. \quad (13)$$

To bridge the knowledge gap between the retrieved local-level and global-level knowledge, we also find a set of reasoning paths $\mathcal{R}$ connecting the retrieved communities. Specifically, from each community, we select the top-m query-related key entities and collect them into $\hat{\mathcal{V}}_{\hat{\mathcal{P}}}$, as shown in Equation 14. The set of reasoning paths $\mathcal{R}$ is defined as the set of shortest paths between each pair of key entities according to their order in $\hat{\mathcal{V}}_{\hat{\mathcal{P}}}$, as shown in Equation 15. Based on $\mathcal{R}$, we construct a subgraph $\hat{\mathcal{R}}$ as described in Equation 16. Here, $\hat{\mathcal{R}}$ collects a set of triples from the KG that connect the knowledge in the local entities and the knowledge in the global communities.

$$\hat{\mathcal{V}}_{\hat{\mathcal{P}}} = \bigcup_{p \in \hat{\mathcal{P}}} TopN(\{v \in p | Sim(q, v)\}, m), \quad (14)$$

$$\mathcal{R} = \bigcup_{i \in [1, |\hat{\mathcal{V}}_{\hat{\mathcal{P}}}| - 1]} ShortestPath_{\mathcal{G}_k}(\hat{\mathcal{V}}_{\hat{\mathcal{P}}}[i], \hat{\mathcal{V}}_{\hat{\mathcal{P}}}[i{+}1]), \quad (15)$$

$$\hat{\mathcal{R}} = \{(h, r, t) \in \mathcal{G}_k | h, t \in \mathcal{R}\}. \quad (16)$$

After retrieving the three-level hierarchical knowledge, i.e., local-level descriptions of the individual entities in $\hat{\mathcal{V}}$, global-level community reports of the communities in $\hat{\mathcal{P}}$, and bridge-level descriptions of the triples in $\hat{\mathcal{R}}$, we feed them as the context to the LLM to generate a comprehensive answer to the query. We also provide the detailed procedures of HiRAG with pseudocodes in Appendix B.

4.3 Why is HiRAG effective?

HiRAG's efficacy stems from its hierarchical architecture, HiIndex (i.e., hierarchical KG) and HiRetrieval (i.e., three-level knowledge retrieval), which directly mitigates the limitations outlined in Challenges (1) and (2) as described in Section 1.

Addressing Challenge (1): The hierarchical knowledge graph $\mathcal{G}_k$ introduces summary entities in its higher layers, creating shortcuts between entities that are distantly located in lower layers. This design bridges semantically related concepts efficiently, bypassing the need for exhaustive traversal of fine-grained relationships in the KG.

Resolving Challenge (2): HiRetrieval constructs reasoning paths by linking the top-n entities most semantically relevant to a query with their associated communities. These paths represent the shortest connections between localized entity descriptions and global community-level insights, ensuring that both granular details and broader contextual knowledge inform the reasoning process.

Synthesis: By integrating (i) semantically similar entities via hierarchical shortcuts, (ii) global community contexts, and (iii) optimized pathways connecting local and global knowledge, HiRAG generates comprehensive, context-aware answers to user queries.

5 Experimental Evaluation

We report the performance evaluation results of HiRAG in this section.

Baseline Methods. We compared HiRAG with state-of-the-art and popular baseline RAG methods. **NaiveRAG** (Gao et al., 2022) (Gao et al., 2023) splits original documents into chunks and retrieves relevant text chunks through vector search. **GraphRAG** (Edge et al., 2024) utilizes communities and we use the local search mode in our experiments as it retrieves community reports as global knowledge, while their global search mode is known to be too costly and does not use local entity descriptions. **LightRAG** (Guo et al., 2024) uses both global and local knowledge to answer a query. **FastGraphRAG** (Circlemind, 2024) integrates KG and personalized PageRank as proposed in HippoRAG (Gutiérrez et al., 2025). **KAG** (Liang et al., 2024) integrates structured reasoning of KG with LLMs and employs mutual indexing and logical-form-guided reasoning to enhance professional domain knowledge services.

Datasets and Queries. We used four datasets from the UltraDomain benchmark (Qian et al., 2024), which is designed to evaluate RAG systems across diverse applications, focusing on long-context tasks and high-level queries in specialized domains. We used Mix, CS, Legal, and Agriculture datasets like in LightRAG (Guo et al., 2024). We also used the benchmark queries provided in UltraDomain for each of the four datasets. The statistics of these datasets are given in Appendix A.

LLM. We employed DeepSeek-V3 (DeepSeek-AI et al., 2024) as the LLM for information extraction, entity summarization, and answer generation in HiRAG and other baseline methods. We utilized GLM-4-Plus (GLM et al., 2024) as the embedding model for vector search and semantic clustering because DeepSeek-V3 does not provide an accessible embedding model.

5.1 Overall Performance Comparison

Evaluation Details. Our experiments followed the evaluation methods of recent work (Edge et al., 2024)(Guo et al., 2024) by employing a powerful LLM to conduct multi-dimensional comparison. We used the **win rate** to compare different methods, which indicates the percentage of instances that a method generates higher-quality answers compared to another method as judged by the LLM. We utilized GPT-4o (Achiam et al., 2023) as the evaluation model to judge which method generates a superior answer for each query for the following four dimensions: (1) **Comprehensiveness**: how thoroughly does the answer address the question, covering all relevant aspects and details? (2) **Empowerment**: how effectively does the answer provide actionable insights or solutions that empower the user to take meaningful steps? (3) **Diversity**: how well does the answer incorporate a variety of perspectives, approaches, or solutions to the problem? (4) **Overall**: how does the answer perform overall, considering comprehensiveness, empowerment, diversity, and any other relevant factors? For a fair comparison, we also alternated the order of the answers generated by each pair of methods in the prompts and calculated the overall win rates of each method.

Evaluation Results. We report the win rates of HiRAG and the five baseline methods in Table 1. HiRAG outperforms the baselines accross the four datasets and the four dimensions in most of the cases. Here are the conclusions we can draw from the results:

Evaluation Results. We present the win rates of HiRAG and five baseline methods in Table 1. HiRAG consistently outperforms existing approaches across all four datasets and four evaluation dimensions in the majority of cases. Key insights derived from the results are summarized below.

Graph structure enhances RAG systems: NaiveRAG exhibits inferior performance compared to methods integrating graph structures, primarily due to its inability to model relationships between entities in retrieved components. Furthermore, its context processing is constrained by the token limitations of LLMs, highlighting the importance of structured knowledge representation for robust retrieval and reasoning.

Global knowledge improves answer quality: Approaches incorporating global knowledge (GraphRAG, LightRAG, KAG, HiRAG) significantly surpass FastGraphRAG, which relies on local knowledge via personalized PageRank. Answers generated without global context lack depth and diversity, underscoring the necessity of holistic knowledge integration for comprehensive responses.

HiRAG's superior performance: Among graph-enhanced RAG systems, HiRAG achieves the highest performance across all datasets (spanning diverse domains) and evaluation dimensions. This superiority stems primarily from two innovations: (1) HiIndex which enhances connections between remote but semantically similar entities in the hierarchical KG, and (2) HiRetrieval which effectively bridges global knowledge with localized context to

Table 1: Win rates (%) of HiRAG, its two variants (for ablation study), and baseline methods.

	Mix		**CS**		**Legal**		**Agriculture**	
	NaiveRAG	**HiRAG**	NaiveRAG	**HiRAG**	NaiveRAG	**HiRAG**	NaiveRAG	**HiRAG**
Comprehensiveness	16.6%	83.4%	30.0%	70.0%	32.5%	67.5%	34.0%	66.0%
Empowerment	11.6%	88.4%	29.0%	71.0%	25.0%	75.0%	31.0%	69.0%
Diversity	12.7%	87.3%	14.5%	85.5%	22.0%	78.0%	21.0%	79.0%
Overall	12.4%	87.6%	26.5%	73.5%	25.5%	74.5%	28.5%	71.5%
	GraphRAG	**HiRAG**	GraphRAG	**HiRAG**	GraphRAG	**HiRAG**	GraphRAG	**HiRAG**
Comprehensiveness	42.1%	57.9%	40.5%	59.5%	48.5%	51.5%	49.0%	51.0%
Empowerment	35.1%	64.9%	38.5%	61.5%	43.5%	56.5%	48.5%	51.5%
Diversity	40.5%	59.5%	30.5%	69.5%	47.0%	53.0%	45.5%	54.5%
Overall	35.9%	64.1%	36.0%	64.0%	45.5%	54.5%	46.0%	54.0%
	LightRAG	**HiRAG**	LightRAG	**HiRAG**	LightRAG	**HiRAG**	LightRAG	**HiRAG**
Comprehensiveness	36.8%	63.2%	44.5%	55.5%	49.0%	51.0%	38.5%	61.5%
Empowerment	34.9%	65.1%	41.5%	58.5%	43.5%	56.5%	36.5%	63.5%
Diversity	34.1%	65.9%	33.0%	67.0%	63.0%	37.0%	37.5%	62.5%
Overall	34.1%	65.9%	41.0%	59.0%	48.0%	52.0%	38.5%	61.5%
	FastGraphRAG	**HiRAG**	FastGraphRAG	**HiRAG**	FastGraphRAG	**HiRAG**	FastGraphRAG	**HiRAG**
Comprehensiveness	0.8%	99.2%	0.0%	100.0%	1.0%	99.0%	0.0%	100.0%
Empowerment	0.8%	99.2%	0.0%	100.0%	0.0%	100.0%	0.0%	100.0%
Diversity	0.8%	99.2%	0.5%	99.5%	1.5%	98.5%	0.0%	100.0%
Overall	0.8%	99.2%	0.0%	100.0%	0.0%	100.0%	0.0%	100.0%
	KAG	**HiRAG**	KAG	**HiRAG**	KAG	**HiRAG**	KAG	**HiRAG**
Comprehensiveness	2.3%	97.7%	1.0%	99.0%	16.5%	83.5%	5.0%	99.5%
Empowerment	3.5%	96.5%	4.5%	95.5%	9.0%	91.0%	5.0%	99.5%
Diversity	3.8%	96.2%	5.0%	95.0%	11.0%	89.0%	3.5%	96.5%
Overall	2.3%	97.7%	1.5%	98.5%	8.5%	91.5%	0.0%	100.0%
	w/o HiIndex	**HiRAG**	w/o HiIndex	**HiRAG**	w/o HiIndex	**HiRAG**	w/o HiIndex	**HiRAG**
Comprehensiveness	46.7%	53.3%	44.2%	55.8%	49.0%	51.0%	50.5%	49.5%
Empowerment	43.2%	56.8%	38.8%	61.2%	47.5%	52.5%	50.5%	49.5%
Diversity	40.5%	59.5%	40.0%	60.0%	48.0%	52.0%	48.5%	51.5%
Overall	42.4%	57.6%	40.0%	60.0%	46.5%	53.5%	48.0%	52.0%
	w/o Bridge	**HiRAG**	w/o Bridge	**HiRAG**	w/o Bridge	**HiRAG**	w/o Bridge	**HiRAG**
Comprehensiveness	49.2%	50.8%	46.5%	53.5%	49.5%	50.5%	47.0%	53.0%
Empowerment	44.2%	55.8%	43.0%	57.0%	38.5%	61.5%	41.0%	59.0%
Diversity	44.6%	55.4%	44.0%	56.0%	43.5%	56.5%	46.0%	54.0%
Overall	47.3%	52.7%	42.5%	57.5%	44.0%	56.0%	42.0%	58.0%

optimize relevance and coherence.

5.2 Hierarchical KG vs. Flat KG

To evaluate the significance of the hierarchical KG, we replace the hierarchical KG with a flat KG (or a basic KG), denoted by **w/o HiIndex** as reported in Table 1. Compared with HiRAG, the win rates of **w/o HiIndex** drop in almost all cases and quite significantly in at least half of the cases. This ablation study thus shows that the hierarchical indexing plays an important role in the quality of answer generation, since the connectivity among semantically similar entities is enhanced with the hierarchical KG, with which related entities can be grouped together both from structural and semantical perspectives.

From Table 1, we also observe that the win rates of **w/o HiIndex** are better or comparable to those of GraphRAG and LightRAG when compared with HiRAG. This suggests that our three-level knowledge retrieval method, i.e., HiRetrieval, is effective even applied on a flat KG, because GraphRAG and LightRAG also index on a flat KG but they only use the local entity descriptions and global community reports, while **w/o HiIndex** uses an additional bridge-level knowledge.

5.3 HiRetrieval vs. Gapped Knowledge

To show the effectiveness of HiRetrieval, we also created another variant of HiRAG without using the bridge-level knowledge, denoted by **w/o Bridge** in Table 1. The result shows that without the bridge-layer knowledge, the win rates drop significantly across all datasets and evaluation dimensions, be-

cause there is knowledge gap between the local-level and global-level knowledge as discussed in Section 1.

Case Study. Figure 3 shows the three-level knowledge used as the context to an LLM to answer the query in Figure 1. The bridge-level knowledge contains entity descriptions from different communities, as shown by the different colors in Figure 3, which helps the LLM correctly answer the question about Amazon's role as an e-commerce and cloud provider.

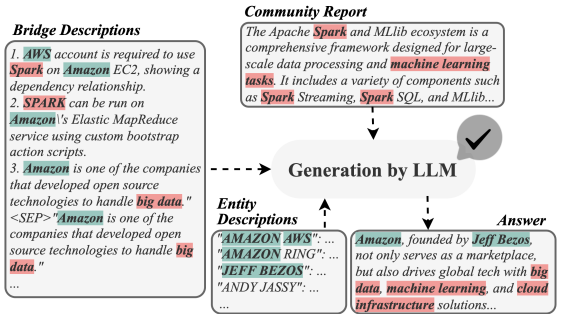

Figure 3: Answer to the query in Figure 1 with additional bridge-level knowledge.

5.4 Determining the Number of Layers

One important thing in HiIndex is to determine the number of layers, k, for the hierarchical KG, which should be determined dynamically according to the quality of clusters in each layer. We stop building another layer when the majority of the clusters consist of only a small number of entities, meaning that the entities can no longer be effectively grouped together. To measure that, we introduce the notion of **cluster sparsity** CS_i, as inspired by graph sparsity, to measure the quality of clusters in the i-th layer as described in Equation 17.

$$CS_i = 1 - \frac{\sum_{\mathcal{S} \in \mathcal{C}_i} |\mathcal{S}|(|\mathcal{S}| - 1)}{|\mathcal{L}_i|(|\mathcal{L}_i| - 1)}. \quad (17)$$

The more the clusters in $\mathcal{C}_i$ have a small number of entities, the larger is CS_i, where the worst case is when each cluster contains only one entity (i.e., $CS_i = 1$). Figure 4 shows that as we have more layers, the cluster sparsity increases and then stabilizes. We also plot the change rate from CS_i to CS_{i+1}, which shows that there is little or no more change after constructing a certain number of layers. We set a threshold $\epsilon = 5\%$ and stop constructing another layer when the change rate of cluster sparsity is lower than ϵ because the cluster quality has little or no improvement after that.

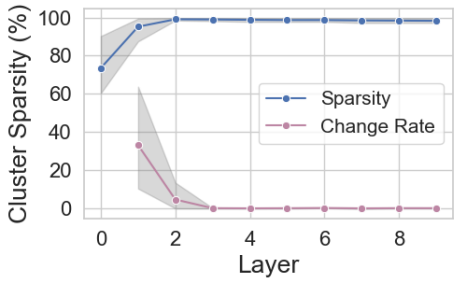

Figure 4: Cluster sparsity CS_i and change rate from CS_i to CS_{i+1}, where the shadow areas represent the value ranges of the four datasets and the blue/pink lines are the respective average values.

5.5 Efficiency and Costs Analysis

To evaluate the efficiency and costs of HiRAG, we also report the token costs, the number of API calls, and the time costs of indexing and retrieval of HiRAG and the baselines in Table 2. For indexing, we record the total costs of the entire process. Although HiRAG needs more time and resources to conduct indexing for better performance, we remark that indexing is offline and the total cost is only about 7.55 USD for the Mix dataset using DeepSeek-V3. In terms of retrieval, we calculate the average costs per query. Unlike KAG and LightRAG, HiRAG does not cost any tokens for retrieval. Therefore, HiRAG is more efficient for online retrieval.

Table 2: Comparisons in terms of tokens, API calls and time cost.

Method	Token Cost		API Calls		Time Cost (s)	
	Indexing	Retrieval	Indexing	Retrieval	Indexing	Retrieval
GraphRAG	8,507,697	0.00	2,666	1.00	6,696	0.70
LightRAG	3,849,030	357.76	1,160	2.00	3,342	3.06
KAG	6,440,668	110,532.00	831	9.17	8,530	58.47
HiRAG	21,898,765	0.00	6,790	1.00	17,208	0.85

6 Conclusions

We presented a new approach to enhance RAG systems by effectively utilizing graph structures with hierarchical knowledge. By developing (1) HiIndex which enhances structural and semantic connectivity across hierarchical layers, and (2) HiRetrieval which effectively bridges global conceptual abstractions with localized entity descriptions, HiRAG achieves superior performance than existing methods.

7 Limitations

HiRAG has the following limitations. Firstly, constructing a high-quality hierarchical KG may incur

substantial token consumption and time overhead, as LLMs need to perform entity summarization in each layer. However, the monetary cost of using LLMs may not be the major concern as the cost is decreasing rapidly recently, and therefore we may consider parallelizing the indexing process to reduce the indexing time. Secondly, the retrieval module requires more sophisticated query-aware ranking mechanisms. Currently, our HiRetrieval module relies solely on LLM-generated weights for relation ranking, which may affect query relevance. We will research for more effective ranking mechanisms to further improve the retrieval quality.

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*.

Salvatore Carta, Alessandro Giuliani, Leonardo Piano, Alessandro Sebastian Podda, Livio Pompianu, and Sandro Gabriele Tiddia. 2023. Iterative zero-shot llm prompting for knowledge graph construction. *arXiv preprint arXiv:2307.01128*.

Circlemind. 2024. fast-graphrag. https://github.com/circlemind-ai/fast-graphrag.

DeepSeek-AI, Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, Damai Dai, Daya Guo, Dejian Yang, Deli Chen, Dongjie Ji, Erhang Li, Fangyun Lin, Fucong Dai, Fuli Luo, Guangbo Hao, Guanting Chen, Guowei Li, H. Zhang, Han Bao, Hanwei Xu, Haocheng Wang, Haowei Zhang, Honghui Ding, Huajian Xin, Huazuo Gao, Hui Li, Hui Qu, J. L. Cai, Jian Liang, Jianzhong Guo, Jiaqi Ni, Jiashi Li, Jiawei Wang, Jin Chen, Jingchang Chen, Jingyang Yuan, Junjie Qiu, Junlong Li, Junxiao Song, Kai Dong, Kai Hu, Kaige Gao, Kang Guan, Kexin Huang, Kuai Yu, Lean Wang, Lecong Zhang, Lei Xu, Leyi Xia, Liang Zhao, Litong Wang, Liyue Zhang, Meng Li, Miaojun Wang, Mingchuan Zhang, Minghua Zhang, Minghui Tang, Mingming Li, Ning Tian, Panpan Huang, Peiyi Wang, Peng Zhang, Qiancheng Wang, Qihao Zhu, Qinyu Chen, Qiushi Du, R. J. Chen, R. L. Jin, Ruiqi Ge, Ruisong Zhang, Ruizhe Pan, Runji Wang, Runxin Xu, Ruoyu Zhang, Ruyi Chen, S. S. Li, Shanghao Lu, Shangyan Zhou, Shanhuang Chen, Shaoqing Wu, Shengfeng Ye, Shengfeng Ye, Shirong Ma, Shiyu Wang, Shuang Zhou, Shuiping Yu, Shunfeng Zhou, Shuting Pan, T. Wang, Tao Yun, Tian Pei, Tianyu Sun, W. L. Xiao, Wangding Zeng, Wanjia Zhao, Wei An, Wen Liu, Wenfeng Liang, Wenjun Gao, Wenqin Yu, Wentao Zhang, X. Q. Li, Xiangyue Jin, Xianzu Wang, Xiao Bi, Xiaodong Liu, Xiaohan Wang, Xiaojin Shen, Xiaokang Chen, Xiaokang Zhang, Xiaosha Chen, Xiaotao Nie, Xiaowen Sun, Xiaoxiang Wang, Xin Cheng, Xin Liu, Xin Xie, Xingchao Liu, Xingkai Yu, Xinnan Song, Xinxia Shan, Xinyi Zhou, Xinyu Yang, Xinyuan Li, Xuecheng Su, Xuheng Lin, Y. K. Li, Y. Q. Wang, Y. X. Wei, Y. X. Zhu, Yang Zhang, Yanhong Xu, Yanhong Xu, Yanping Huang, Yao Li, Yao Zhao, Yaofeng Sun, Yaohui Li, Yaohui Wang, Yi Yu, Yi Zheng, Yichao Zhang, Yifan Shi, Yiliang Xiong, Ying He, Ying Tang, Yishi Piao, Yisong Wang, Yixuan Tan, Yiyang Ma, Yiyuan Liu, Yongqiang Guo, Yu Wu, Yuan Ou, Yuchen Zhu, Yuduan Wang, Yue Gong, Yuheng Zou, Yujia He, Yukun Zha, Yunfan Xiong, Yunxian Ma, Yuting Yan, Yuxiang Luo, Yuxiang You, Yuxuan Liu, Yuyang Zhou, Z. F. Wu, Z. Z. Ren, Zehui Ren, Zhangli Sha, Zhe Fu, Zhean Xu, Zhen Huang, Zhen Zhang, Zhenda Xie, Zhengyan Zhang, Zhewen Hao, Zhibin Gou, Zhicheng Ma, Zhigang Yan, Zhihong Shao, Zhipeng Xu, Zhiyu Wu, Zhongyu Zhang, Zhuoshu Li, Zihui Gu, Zijia Zhu, Zijun Liu, Zilin Li, Ziwei Xie, Ziyang Song, Ziyi Gao, and Zizheng Pan. 2024. Deepseek-v3 technical report. *Preprint*, arXiv:2412.19437.

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

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.

Luyu Gao, Xueguang Ma, Jimmy Lin, and Jamie Callan. 2022. Precise zero-shot dense retrieval without relevance labels. *arXiv preprint arXiv:2212.10496*.

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

Team GLM, :, Aohan Zeng, Bin Xu, Bowen Wang, Chenhui Zhang, Da Yin, Dan Zhang, Diego Rojas, Guanyu Feng, Hanlin Zhao, Hanyu Lai, Hao Yu, Hongning Wang, Jiadai Sun, Jiajie Zhang, Jiale Cheng, Jiayi Gui, Jie Tang, Jing Zhang, Jingyu Sun, Juanzi Li, Lei Zhao, Lindong Wu, Lucen Zhong, Mingdao Liu, Minlie Huang, Peng Zhang, Qinkai Zheng, Rui Lu, Shuaiqi Duan, Shudan Zhang, Shulin Cao, Shuxun Yang, Weng Lam Tam, Wenyi Zhao, Xiao Liu, Xiao Xia, Xiaohan Zhang, Xiaotao Gu, Xin Lv, Xinghan Liu, Xinyi Liu, Xinyue Yang, Xixuan Song, Xunkai Zhang, Yifan An, Yifan Xu, Yilin Niu, Yuantao Yang, Yueyan Li, Yushi Bai, Yuxiao Dong, Zehan Qi, Zhaoyu Wang, Zhen Yang, Zhengxiao Du, Zhenyu Hou, and Zihan Wang. 2024. Chatglm: A family of large language models from glm-130b to glm-4 all tools. *Preprint*, arXiv:2406.12793.

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, Yu Gu, Michihiro Yasunaga, and Yu Su. 2025. Hipporag: Neurobiologically inspired long-term memory for large language models. *Preprint*, arXiv:2405.14831.

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

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.

Zhuoqun Li, Xuanang Chen, Haiyang Yu, Hongyu Lin, Yaojie Lu, Qiaoyu Tang, Fei Huang, Xianpei Han, Le Sun, and Yongbin Li. 2024. Structrag: Boosting knowledge intensive reasoning of llms via inference-time hybrid information structurization. *arXiv preprint arXiv:2410.08815*.

Lei Liang, Mengshu Sun, Zhengke Gui, Zhongshu Zhu, Zhouyu Jiang, Ling Zhong, Yuan Qu, Peilong Zhao, Zhongpu Bo, Jin Yang, Huaidong Xiong, Lin Yuan, Jun Xu, Zaoyang Wang, Zhiqiang Zhang, Wen Zhang, Huajun Chen, Wenguang Chen, and Jun Zhou. 2024. Kag: Boosting llms in professional domains via knowledge augmented generation. *Preprint*, arXiv:2409.13731.

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

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

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

Hongjin Qian, Peitian Zhang, Zheng Liu, Kelong Mao, and Zhicheng Dou. 2024. Memorag: Moving towards next-gen rag via memory-inspired knowledge discovery. *arXiv preprint arXiv:2409.05591*.

Bahareh Sarrafzadeh and Edward Lank. 2017. Improving exploratory search experience through hierarchical knowledge graphs. In *Proceedings of the 40th international ACM SIGIR conference on research and development in information retrieval*, pages 145–154.

Parth Sarthi, Salman Abdullah, Aditi Tuli, Shubh Khanna, Anna Goldie, and Christopher D Manning. 2024. Raptor: Recursive abstractive processing for tree-organized retrieval. *arXiv preprint arXiv:2401.18059*.

Yixuan Tang and Yi Yang. 2024. Multihop-rag: Benchmarking retrieval-augmented generation for multi-hop queries. *Preprint*, arXiv:2401.15391.

Vincent A Traag, Ludo Waltman, and Nees Jan Van Eck. 2019. From louvain to leiden: guaranteeing well-connected communities. *Scientific reports*, 9(1):1–12.

Qinggang Zhang, Shengyuan Chen, Yuanchen Bei, Zheng Yuan, Huachi Zhou, Zijin Hong, Junnan Dong, Hao Chen, Yi Chang, and Xiao Huang. 2025. A survey of graph retrieval-augmented generation for customized large language models. *Preprint*, arXiv:2501.13958.

Yue Zhang, Yafu Li, Leyang Cui, Deng Cai, Lemao Liu, Tingchen Fu, Xinting Huang, Enbo Zhao, Yu Zhang, Yulong Chen, et al. 2023. Siren's song in the ai ocean: a survey on hallucination in large language models. *arXiv preprint arXiv:2309.01219*.

Appendix

In this section, we delve into the construction of the hierarchical KG with the HiIndex module, accompanied by illustrative pseudo-codes. We present statistics and a simple case study to demonstrate the enhanced connectivity among entities in the hierarchical KG. Additionally, we give well designed prompt templates used in HiRAG.

A Experimental Datasets

Table 3: Statistics of datasets.

Dataset	Mix	CS	Legal	Agriculture
# of Documents	61	10	94	12
# of Tokens	625948	2210894	5279400	2028496

Table 3 presents the statistical characteristics of the experimental datasets, where all documents were consistently tokenized using Byte Pair Encoding (BPE) tokenizer "cl100k_base".

B Implementation Details of HiRAG

We give a more detailed and formulated expression of hierarchical indexing (HiIndex) and hierarchical retrieval (HiRetrieval). As described in Algorithm 1, the hierarchical knowledge graph is constructed iteratively. The number of clustered layers depends on the rate of change in the cluster

Algorithm 1: HiIndex

Input: Basic knowledge graph $\mathcal{G}_0$ extracted by the LLM; Predefined threshold ϵ;

Output: Hierarchical knowledge graph $\mathcal{G}_k$;

$\mathcal{L}_0 \leftarrow \mathcal{V}_0$;
$\mathcal{Z}_0 \leftarrow \{Embedding(v)|v \in \mathcal{L}_0\}$;
$i \leftarrow 1$;
while True **do**
  */*Perform semantical clustering*/*
  $\mathcal{C}_{i-1} \leftarrow GMM(\mathcal{G}_{i-1}, \mathcal{Z}_{i-1})$;
  */*Calculate cluster sparsity*/*
  $CS_i \leftarrow 1 - \frac{\sum_{\mathcal{S}_x \in \mathcal{C}_{i-1}} |\mathcal{S}_x|(|\mathcal{S}_x|-1)}{|\mathcal{L}_{i-1}|(|\mathcal{L}_{i-1}|-1)}$;
  if change rate of $CS_i \leq \epsilon$ **then**
    $i \leftarrow i - 1$;
    break;
  end if
  */*Generate summary entities and relations*/*
  $\mathcal{L}_i \leftarrow \{\}$;
  $\mathcal{E}_{\{i-1,i\}} \leftarrow \{\}$;
  for $\mathcal{S}_x$ in $\mathcal{C}_{i-1}$ **do**
    $\mathcal{L}, \mathcal{E} \leftarrow LLM(\mathcal{S}_x, \mathcal{X})$;
    $\mathcal{L}_i \leftarrow \mathcal{L}_i \cup \mathcal{L}$;
    $\mathcal{E}_{\{i-1,i\}} \leftarrow \mathcal{E}_{\{i-1,i\}} \cup \mathcal{E}$;
  end for
  $\mathcal{Z}_i = \{Embedding(v)|v \in \mathcal{L}_i\}$;
  */*Update KG*/*
  $\mathcal{E}_i \leftarrow \mathcal{E}_{i-1} \cup \mathcal{E}_{\{i-1,i\}}$;
  $\mathcal{V}_i \leftarrow \mathcal{V}_{i-1} \cup \mathcal{L}_i$;
  $\mathcal{G}_i \leftarrow \{(h, r, t)|h, t \in \mathcal{V}_i, r \in \mathcal{E}_i\}$
  $i \leftarrow i + 1$;
end while
$k \leftarrow i$;
$\mathcal{G}_k \leftarrow \{(h, r, t)|h, t \in \mathcal{V}_k, r \in \mathcal{E}_k\}$;

Algorithm 2: HiRetrieval

Input: The hierarchical knowledge graph $\mathcal{G}_k$; The detected community set $\mathcal{P}$ in $\mathcal{G}_k$; The number of retrieved entities n; The number of selected key entities m in each retrieved community;

Output: The generated answer a;

*/*The local-layer knowledge context*/*
$\hat{\mathcal{V}} \leftarrow TopN(\{v \in \mathcal{V}_k|Sim(v, q)\}, n)$;
*/*The global-layer knowledge context*/*
$\hat{\mathcal{P}} \leftarrow \bigcup_{p \in \mathcal{P}} \{p|p \cap \hat{\mathcal{V}} \neq \phi\}$;
$\hat{\mathcal{R}} \leftarrow \{\}$;
$\hat{\mathcal{V}}_{\hat{\mathcal{P}}} \leftarrow \{\}$;
*/*Select key entities*/*
for p in $\hat{\mathcal{P}}$ **do**
  $\hat{\mathcal{V}}_{\hat{\mathcal{P}}} \leftarrow \hat{\mathcal{V}}_{\hat{\mathcal{P}}} \cup TopN(\{v \in p|Sim(v, q)\}, m)$;
end for
*/*Find the reasoning path*/*
for i in $[1, |\hat{\mathcal{V}}_{\hat{\mathcal{P}}}| - 1]$ **do**
  $\mathcal{R} \leftarrow \mathcal{R} \cup ShortestPath_{\mathcal{G}_k}(\hat{\mathcal{V}}_{\hat{\mathcal{P}}}[i], \hat{\mathcal{V}}_{\hat{\mathcal{P}}}[i+1])$;
end for
*/*The bridge-layer knowledge context*/*
$\hat{\mathcal{R}} \leftarrow \{(h, r, t) \in \mathcal{G}_k|h, t \in \mathcal{R}\}$;
*/*Generate the answer*/*
$a \leftarrow LLM(q, \hat{\mathcal{V}}, \hat{R}, \hat{\mathcal{P}})$;

sparsity at each layer. As shown in Algorithm 2, we retrieve knowledge of three layers (local layer, global layer, and bridge layer) as contexts for LLM to generate more comprehensive and accurate answers.

C The Clustering Coefficients of HiIndex

We calculate and compare the clustering coefficients of GraphRAG, LightRAG and HiRAG in Figure 5. HiRAG shows a higher clustering coefficient than other baseline methods, which means that more entities in the hierarchical KG constructed by the HiIndex module tend to cluster together. And this is also the reason why the HiIndex module can improve the performance of RAG systems.

D A Simple Case of Hierarchical KG

As shown in Figure 6, we fix the issues mentioned in Section 1 with a hierarchical KG. This case demonstrates that the GMMs clustered semantically similar entities "BIG DATA" and "RECOMMENDATION SYSTEM" together. The LLM summarizes "DISTRIBUTED COMPUTING" as their shared summary entities in the next layer. As a consequence, the connections between these related entities can be enhanced from a semantic perspective.

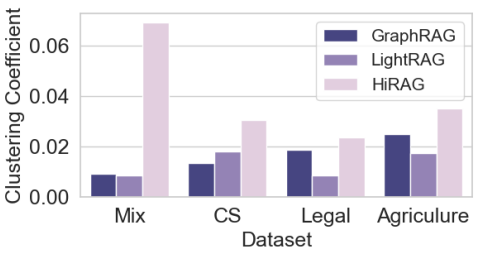

Figure 5: Comparisons between the clustering coefficients of GraphRAG, LightRAG and HiRAG across four datasets.

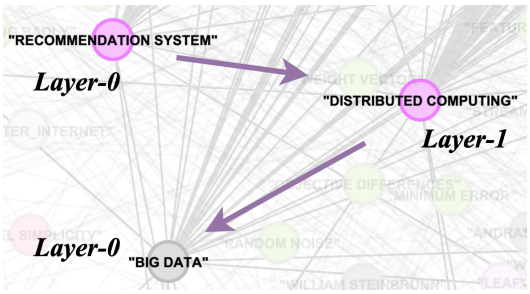

Figure 6: The shortest path with hierarchical KG between the entities in the case mentioned in the introduction.

E Prompt Templates used in HiRAG

E.1 Prompt Templates for Entity Extraction

As shown in Figure 7, we used that prompt template to extract entities from text chunks. We also give three examples to guide the LLM to extract entities with higher accuracy.

E.2 Prompt Templates for Relation Extraction

As shown in Figure 8, we extract relations from the entities extracted earlier and the corresponding text chunks. Then we can get the triples in the basic knowledge graph, which is also the 0-th layer of the hierarchical knowledge graph.

E.3 Prompt Templates for Entity Summarization

As shown in Figure 9, we generate summary entities in each layer of the hierarchical knowledge graph. We will not only let the LLM generate the summary entities from the previous layer, but also let it generate the relations between the entities of these two layers. These relations will clarify the reasons for summarizing these entities.

E.4 Prompt Templates for RAG Evaluation

In terms of the prompt templates we use to conduct evaluations, we utilize the same prompt design as that in LightRAG. The prompt will let the LLM generate both evaluation results and the reasons in JSON format to ensure clarity and accuracy.

Entity Extraction Prompt

-Goal-
Given a text document that is potentially relevant to a list of entity types, identify all entities of those types.

-Steps-
1. Identify all entities. For each identified entity, extract the following information:
- entity_name: Name of the entity, capitalized
- entity_type: One of the following types: [{entity_types}], normal_entity means that doesn't belong to any other types.
- entity_description: Comprehensive description of the entity's attributes and activities
Format each entity as ("entity"{tuple_delimiter}<entity_name>{tuple_delimiter}<entity_type>{tuple_delimiter}<entity_description>

2. Return output in English as a single list of all the entities identified in step 1. Use **{record_delimiter}** as the list delimiter.

3. When finished, output {completion_delimiter}

######################
-Examples-
######################
Example 1:

Entity_types: [person, technology, mission, organization, location]
Text:
while Alex clenched his jaw, the buzz of frustration dull against the backdrop of Taylor's authoritarian certainty. It was this competitive undercurrent that kept him alert, the sense that his and Jordan's shared commitment to discovery was an unspoken rebellion against Cruz's narrowing vision of control and order.

Then Taylor did something unexpected. They paused beside Jordan and, for a moment, observed the device with something akin to reverence. “If this tech can be understood..." Taylor said, their voice quieter, "It could change the game for us. For all of us.”

The underlying dismissal earlier seemed to falter, replaced by a glimpse of reluctant respect for the gravity of what lay in their hands. Jordan looked up, and for a fleeting heartbeat, their eyes locked with Taylor's, a wordless clash of wills softening into an uneasy truce.

It was a small transformation, barely perceptible, but one that Alex noted with an inward nod. They had all been brought here by different paths
################
Output:
("entity"{tuple_delimiter}"Alex"{tuple_delimiter}"person"{tuple_delimiter}"Alex is a character who experiences frustration and is observant of the dynamics among other characters."){record_delimiter}
("entity"{tuple_delimiter}"Taylor"{tuple_delimiter}"person"{tuple_delimiter}"Taylor is portrayed with authoritarian certainty and shows a moment of reverence towards a device, indicating a change in perspective."){record_delimiter}
("entity"{tuple_delimiter}"Jordan"{tuple_delimiter}"person"{tuple_delimiter}"Jordan shares a commitment to discovery and has a significant interaction with Taylor regarding a device."){record_delimiter}
("entity"{tuple_delimiter}"Cruz"{tuple_delimiter}"person"{tuple_delimiter}"Cruz is associated with a vision of control and order, influencing the dynamics among other characters."){record_delimiter}
("entity"{tuple_delimiter}"The Device"{tuple_delimiter}"technology"{tuple_delimiter}"The Device is central to the story, with potential game-changing implications, and is revered by Taylor."){record_delimiter}
#############################
Example 2:
....
#############################
Example 3:
....
#############################
-Real Data-
######################
Entity_types: {entity_types}
Text: {input_text}
######################
Output:

Figure 7: The prompt template designed to extract entities from text chunks.

Relation Extraction Prompt

-Goal-
Given a text document that is potentially relevant to a list of entities, identify all relationships among the given identified entities.

-Steps-
1. From the entities given by user, identify all pairs of (source_entity, target_entity) that are *clearly related* to each other.
For each pair of related entities, extract the following information:
- source_entity: name of the source entity, as identified in step 1
- target_entity: name of the target entity, as identified in step 1
- relationship_description: explanation as to why you think the source entity and the target entity are related to each other
- relationship_strength: a numeric score indicating strength of the relationship between the source entity and target entity
Format each relationship as ("relationship"{tuple_delimiter}<source_entity>{tuple_delimiter}<target_entity>{tuple_delimiter}<relationship_description>{tuple_delimiter}<relationship_strength>)

2. Return output in English as a single list of all the entities and relationships identified in steps 1 and 2. Use **{record_delimiter}** as the list delimiter.

3. When finished, output {completion_delimiter}

######################
-Examples-
######################
Example 1:

Entities: ["Alex", "Taylor", "Jordan", "Cruz", "The Device"]
Text:
while Alex clenched his jaw, the buzz of frustration dull against the backdrop of Taylor's authoritarian certainty. It was this competitive undercurrent that kept him alert, the sense that his and Jordan's shared commitment to discovery was an unspoken rebellion against Cruz's narrowing vision of control and order.

Then Taylor did something unexpected. They paused beside Jordan and, for a moment, observed the device with something akin to reverence. “If this tech can be understood..." Taylor said, their voice quieter, "It could change the game for us. For all of us.”

The underlying dismissal earlier seemed to falter, replaced by a glimpse of reluctant respect for the gravity of what lay in their hands. Jordan looked up, and for a fleeting heartbeat, their eyes locked with Taylor's, a wordless clash of wills softening into an uneasy truce.

It was a small transformation, barely perceptible, but one that Alex noted with an inward nod. They had all been brought here by different paths
################
Output:
("relationship"{tuple_delimiter}"Alex"{tuple_delimiter}"Taylor"{tuple_delimiter}"Alex is affected by Taylor's authoritarian certainty and observes changes in Taylor's attitude towards the device."{tuple_delimiter}7){record_delimiter}
("relationship"{tuple_delimiter}"Alex"{tuple_delimiter}"Jordan"{tuple_delimiter}"Alex and Jordan share a commitment to discovery, which contrasts with Cruz's vision."{tuple_delimiter}6){record_delimiter}
("relationship"{tuple_delimiter}"Taylor"{tuple_delimiter}"Jordan"{tuple_delimiter}"Taylor and Jordan interact directly regarding the device, leading to a moment of mutual respect and an uneasy truce."{tuple_delimiter}8){record_delimiter}
("relationship"{tuple_delimiter}"Jordan"{tuple_delimiter}"Cruz"{tuple_delimiter}"Jordan's commitment to discovery is in rebellion against Cruz's vision of control and order."{tuple_delimiter}5){record_delimiter}
("relationship"{tuple_delimiter}"Taylor"{tuple_delimiter}"The Device"{tuple_delimiter}"Taylor shows reverence towards the device, indicating its importance and potential impact."{tuple_delimiter}9){completion_delimiter}
#############################
Example 2:
....
#############################
Example 3:
....
#############################
-Real Data-
######################
Entities: {entities}
Text: {input_text}
######################
Output:

Figure 8: The prompt template designed to extract relations from entities and text chunks.

Entity Summarization Prompt

-Goal-
You are tasked with analyzing a set of entity descriptions and a given list of meta attributes. Your goal is to summarize at least one attribute entity for the entity set in the given entity descriptions. And the summarized attribute entity must match the type of at least one meta attribute in the given meta attribute list (e.g., if a meta attribute is "company", the attribute entity could be "Amazon" or "Meta", which is a kind of meta attribute "company"). And it shoud be directly relevant to the entities described in the entity description set. The relationship between the entity set and the generated attribute entity should be clear and logical.

-Steps-
1. Identify at least one attribute entity for the given entity description list. For each attribute entity, extract the following information:
- entity_name: Name of the entity, capitalized
- entity_type: One of the following types: [{meta_attribute_list}], normal_entity means that doesn't belong to any other types.
- entity_description: Comprehensive description of the entity's attributes and activities
Format each entity as ("entity"{tuple_delimiter}<entity_name>{tuple_delimiter}<entity_type>{tuple_delimiter}<entity_description>

2. From each given entity, identify all pairs of (source_entity, target_entity) that are *clearly related* to the summary entities identified in step 1. And there should be no relations between the summary entities.
For each pair of related entities, extract the following information:
- source_entity: name of the source entity, as given in entity list
- target_entity: name of the target entity, as identified in step 1
- relationship_description: explanation as to why you think the source entity and the target entity are related to each other
- relationship_strength: a numeric score indicating strength of the relationship between the source entity and target entity
Format each relationship as ("relationship"{tuple_delimiter}<source_entity>{tuple_delimiter}<target_entity>{tuple_delimiter}<relationship_description>{tuple_delimiter}<relationship_strength>)

3. Return output in English as a single list of all the entities and relationships identified in steps 1 and 2. Use **{record_delimiter}** as the list delimiter.

4. When finished, output {completion_delimiter}

######################
-Examples-
######################
Example1:

Input:
Meta summary entity list: ["company", "location"]
Entity description list: [("Instagram", "Instagram is a software developed by Meta, which captures and shares the world's moments. Follow friends and family to see what they're up to, and discover accounts from all over the world that are sharing things you love."), ("Facebook", "Facebook is a social networking platform launched in 2004 that allows users to connect, share updates, and engage with communities. Owned by Meta, it is one of the largest social media platforms globally, offering tools for communication, business, and advertising."), ("WhatsApp", "WhatsApp Messenger: A messaging app of Meta for simple, reliable, and secure communication. Connect with friends and family, send messages, make voice and video calls, share media, and stay in touch with loved ones, no matter where they are")]
#######
Output:
("entity"{tuple_delimiter}"Meta"{tuple_delimiter}"company"{tuple_delimiter}"Meta, formerly known as Facebook, Inc., is an American multinational technology conglomerate. It is known for its various online social media services."){record_delimiter}
("relationship"{tuple_delimiter}"Instagram"{tuple_delimiter}"Meta"{tuple_delimiter}"Instagram is a software developed by Meta."{tuple_delimiter}8.5){record_delimiter}
("relationship"{tuple_delimiter}"Facebook"{tuple_delimiter}"Meta"{tuple_delimiter}"Facebook is owned by Meta."{tuple_delimiter}9.0){record_delimiter}
("relationship"{tuple_delimiter}"WhatsApp"{tuple_delimiter}"Meta"{tuple_delimiter}"WhatsApp Messenger is a messaging app of Meta."{tuple_delimiter}8.0){record_delimiter}
##############################
Example2:
....
##############################
Example3:
....
##############################
-Real Data-
######################
Input:
Meta summary entity list: {meta_attribute_list}
Entity description list: {entity_description_list}
#######
Output:

Figure 9: The prompt template designed to generate summary entities and the corresponding relations.