HydraRAG: Structured Cross-Source Enhanced Large Language Model Reasoning

Xingyu Tan[1,2], Xiaoyang Wang[1,*], Qing Liu[2], Xiwei Xu[2], Xin Yuan[2], Liming Zhu[2], Wenjie Zhang[1]
[1]University of New South Wales, Australia
[2]Data61, CSIRO, Australia
{xingyu.tan, xiaoyang.wang1, wenjie.zhang}@unsw.edu.au
{q.liu, xiwei.xu, xin.yuan, liming.zhu}@data61.csiro.au

Abstract

Retrieval-augmented generation (RAG) enhances large language models (LLMs) by incorporating external knowledge. Current hybrid RAG system retrieves evidence from both knowledge graphs (KGs) and text documents to support LLM reasoning. However, it faces challenges like handling multi-hop reasoning, multi-entity questions, multi-source verification, and effective graph utilization. To address these limitations, we present **HydraRAG**, a training-free framework that unifies graph topology, document semantics, and source reliability to support deep, faithful reasoning in LLMs. HydraRAG handles multi-hop and multi-entity problems through agent-driven exploration that combines structured and unstructured retrieval, increasing both diversity and precision of evidence. To tackle multi-source verification, HydraRAG uses a tri-factor cross-source verification (source trustworthiness assessment, cross-source corroboration, and entity-path alignment), to balance topic relevance with cross-modal agreement. By leveraging graph structure, HydraRAG fuses heterogeneous sources, guides efficient exploration, and prunes noise early. Comprehensive experiments on seven benchmark datasets show that HydraRAG achieves overall state-of-the-art results on all benchmarks with GPT-3.5, outperforming the strong hybrid baseline ToG-2 by an average of 20.3% and up to 30.1%. Furthermore, HydraRAG enables smaller models (e.g., Llama-3.1-8B) to achieve reasoning performance comparable to that of GPT-4-Turbo. The source code is available on https://stevetantan.github.io/HydraRAG/.

1 Introduction

Large Language Models (LLMs) have achieved remarkable performance by scaling to billions of parameters and pre-training on vast and diverse corpora (Brown, 2020; Chowdhery et al., 2023). However, the prohibitive expense of full-model training for LLMs makes continual retraining infeasible, causing static parametric knowledge to quickly become obsolete and resulting in factual gaps and hallucinations (Besta et al., 2024; Touvron et al., 2023). This issue is alleviated by retrieval-augmented generation (RAG), which fetches external evidence at inference time. (Gao et al., 2023).

Many RAG systems rely on vector retrieval over text, embedding question and documents into a dense space and selecting semantically similar passages (Baek et al., 2023; Jiang et al., 2023; Huang et al., 2024a,b). While effective for measuring text similarity, such approaches struggle with complex reasoning that requires integrating heterogeneous clues across multiple documents (Ma et al., 2025b). Specifically, (i) different passages may reference distinct entities that share the same underlying concept, such as, Evolar and Evolar AB in Figure 1(a) refer to the same start-up company; (ii) a single passage often covers only one facet of an entity, omitting other critical attributes found in other texts or documents. In Figure 1(a), with the real-time web information implementation, the naive RAG could find the answer to the first part of the question, but could not relate this entity to other text corpora.

To address these challenges, incorporating external knowledge sources, like Knowledge Graphs (KGs), is promising as KGs offer abundant factual knowledge in a structured format, serving as a reliable source to improve LLM capabilities (Sun et al., 2024; Tan et al., 2025). KG-based RAG approaches prompt LLMs with retrieved KG triples or paths relevant to the question, and their effectiveness in dealing with complex reasoning tasks has been demonstrated by researchers (Tan et al., 2025). Although they benefit from the structural and factual nature of KGs, they inherently suffer from inner incompleteness, lack of information beyond their ontology, and high cost of updating (Ma et al., 2025b). For example, as shown in Figure 1(b), the

*Corresponding author.

Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing, pages 14443–14471
November 4-9, 2025

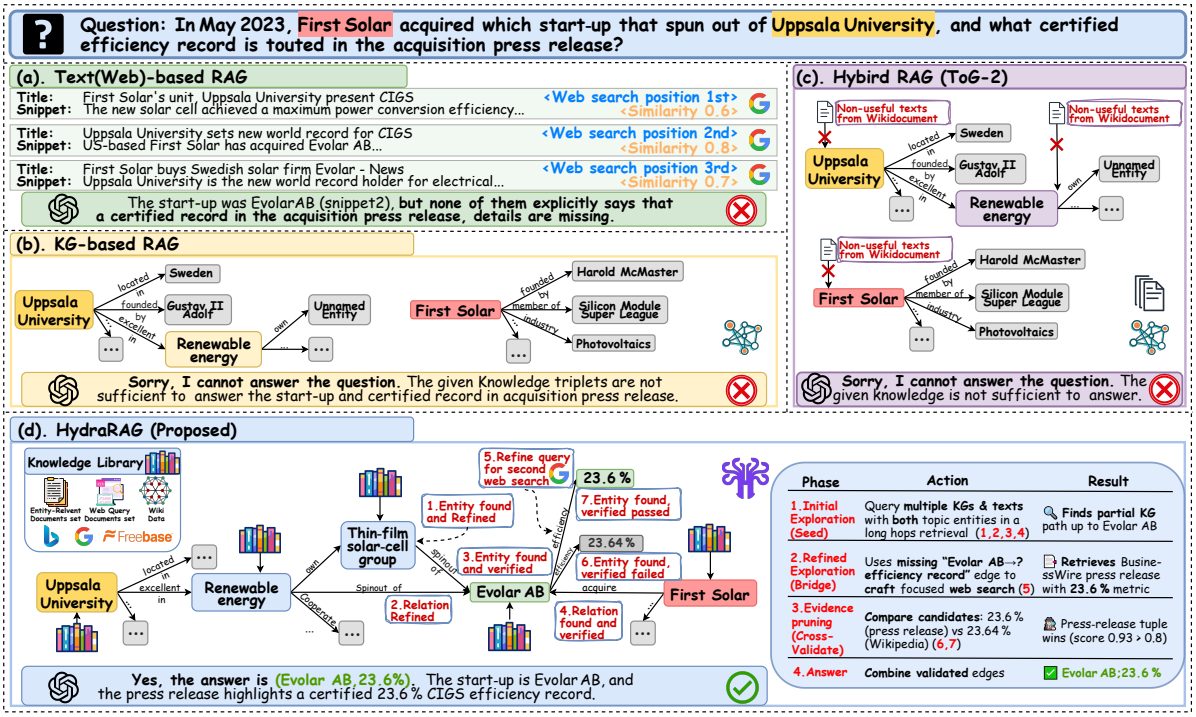


Figure 1: Representative workflow of four LLM reasoning paradigms.

KG search is limited by being unable to provide further information about "Renewable Energy" and "First Solar". Some recent works focus on integrating text and KG as a hybrid RAG system (Li et al., 2024c; Ma et al., 2025b).

Limitations of existing methods. Current approaches typically follow a simple `retrieve-and-select` routine. For instance, CoK alternates between different Knowledge Bases (KBs), choosing one source at each step and retrieving an answer directly (Li et al., 2024c). ToG-2, shown in Figure 1(c), simultaneously queries text and KG, extracting one-hop triples for each question keyword and using an LLM to select the best answer (Ma et al., 2025b). This strategy suffers from four limitations:

Multi-source verification. When faced with multiple sources, many approaches simply concatenate evidence and let the LLM decide. This over-relies on the LLM's semantics without accounting for source reliability or cross-source consistency, leading to both under- and over-pruning of evidence.

Multi-hop reasoning. Existing methods typically retrieve only one-hop relations in text and KG per step and rely on LLMs for semantically relevant candidates pruning. This greedy, local strategy may prune the correct multi-hop path prematurely and fail to consider the global reasoning structure.

Multi-entity questions. Typical pipelines explore each topic entity independently. For questions involving several entities, this produces large candidate sets containing paths unrelated to the other entities, reducing precision and introducing noise.

Graph structure utilization. Current methods fetch triples from each source and pass them to the LLM without merging them into a single graph. Lacking this global structure, the LLM cannot perform efficient graph-based exploration or pruning, so all direct neighbors from KGs and text remain, adding substantial noise.

Contributions. We present **HydraRAG**, shown in Figure 1(d), a structured source-aware retrieval-augmented framework that brings together graph topology, document semantics, and source reliability signals to support deep, faithful reasoning in LLMs. Unlike methods that treat KG triples and text passages as separate evidence, HydraRAG extracts joint KG–text reasoning paths that cover every topic entity and trace multi-hop relations across heterogeneous sources. These paths form interpretable chains of thought, revealing both answers and their cross-source support.

To address multi-source verification, HydraRAG computes a tri-factor score, combining source trustworthiness, cross-source corroboration, and entity-to-evidence alignment. Low-scoring branches are discarded before LLM calls, reducing token usage and preventing source-specific noise.

To address multi-hop reasoning, HydraRAG gen-

erates an indicator in the question analysis stage that predicts the relationship depth between each topic entity and the answer. Guided by it, the system retrieves multi-hop paths from a predicted depth in the KG, enabling dynamic structured search. The same path requirement guides unstructured retrieval to connect related text chains across documents. Unlike approaches that restart retrieval at every step, HydraRAG enhances LLMs to follow coherent reasoning paths that lead to the answer.

To address multi-entity questions, HydraRAG uses a three-phase exploration process over the question subgraph, documents, and web results. All paths must include every topic entity in the order given by the skyline indicator. In structured retrieval, the paths are logical and faithful; in unstructured retrieval, keywords and their connections are searched across text. Each path yields one answer candidate and serves as an interpretable reasoning chain, leveraging both LLM and KG knowledge.

To address graph-structure under-utilization, HydraRAG forms a question subgraph by expanding topic entities to their maximal-depth neighbors and merging subgraphs from multiple KGs. We apply node clustering and graph reduction to cut the search costs and inject high-confidence text edges to dynamically fill KG gaps. During evidence exploration, a semantics-gated, multi-source-verified, bidirectional BFS prunes low-confidence branches early. Inspired by GoT (Besta et al., 2024), HydraRAG prompts the LLM to summarize the top-$W_{\max}$ paths before answer evaluation to further reduce hallucinations. In summary, the advantages of HydraRAG can be abbreviated as:

Structured source-aware retrieval: HydraRAG integrates heterogeneous evidence from diverse sources into a unified structured representation, enabling seamless reasoning.

Multi-source verification: HydraRAG prunes candidate paths based on both question relevance and cross-source corroboration before any LLM call, generating a compact, high-confidence context that reduces hallucinations and lowers LLM costs.

Interpretable cross-source reasoning: The extracted reasoning paths trace how facts from different modalities converge on the answer, providing transparent, step-by-step justification and enhancing the faithfulness of LLM outputs.

Efficiency and adaptability: a) HydraRAG is a plug-and-play framework that can be seamlessly applied to various LLMs, KGs, and texts. b) HydraRAG is auto-refresh. New information is incorporated instantly via web retrieval instead of costly LLM fine-tuning. c) HydraRAG achieves state-of-the-art results on all the tested datasets, surpasses the strong hybrid baseline ToG-2 by an average of 20.3% and up to 30.1%, and enables smaller models to achieve reasoning performance comparable to GPT-4-Turbo.

2 Related Work

Text-based RAG. Early text-based RAG systems embed queries and texts in a shared vector space and retrieve the closest chunks (Gao et al., 2023; Wang et al., 2025a; Ding et al., 2025). Iterative methods such as ITERRETGEN alternate between retrieval and generation to add context (Shao et al., 2023), but coarse passages often mix relevant facts with noise, weakening the signal for reasoning. CoT prompts can guide retrieval toward deeper clues (Wei et al., 2022), but they still rely on semantic similarity and ignore the structure of relations, so long-range connections may be missed or require many iterations to uncover.

KG-based RAG. Graphs are widely used to model complex relationships among different entities (Sima et al., 2025; Wang et al., 2024a,b, 2025b; Tan et al., 2023a,b). KGs store triples, making entity links explicit (Hu et al., 2025; Li et al., 2024b,a). Agent-based methods let an LLM walk the graph hop by hop. ToG asks the LLM to choose the next neighbour at each step (Sun et al., 2024), and StructGPT reformulates a structured query into repeated read-reason cycles (Jiang et al., 2023). Plan-on-Graph and DoG run several LLM calls to rank candidate neighbours (Chen et al., 2024b; Ma et al., 2025a). But a walk starts from a single entity can miss answers that involve several topic entities and becomes fragile on long chains. Paths-over-Graph (Tan et al., 2025) focuses on multi-hop reasoning but relies solely on the KG, so it inherits KG gaps and rising update costs.

Hybrid RAG. Recent work combines structured and unstructured sources. GraphRAG builds a document-level KG to guide passage retrieval (Edge et al., 2024), CoK mixes multiple sources to ground outputs (Li et al., 2024c), and HybridRAG unifies vector and KG retrieval in a single pipeline (Sarmah et al., 2024). Although these methods improve coverage, they retrieve each source separately and simply concatenate results, which can introduce redundant or low-quality evidence. Agentic approaches like ReAct interleave reasoning with retrieval actions to reduce errors (Yao et al., 2023),

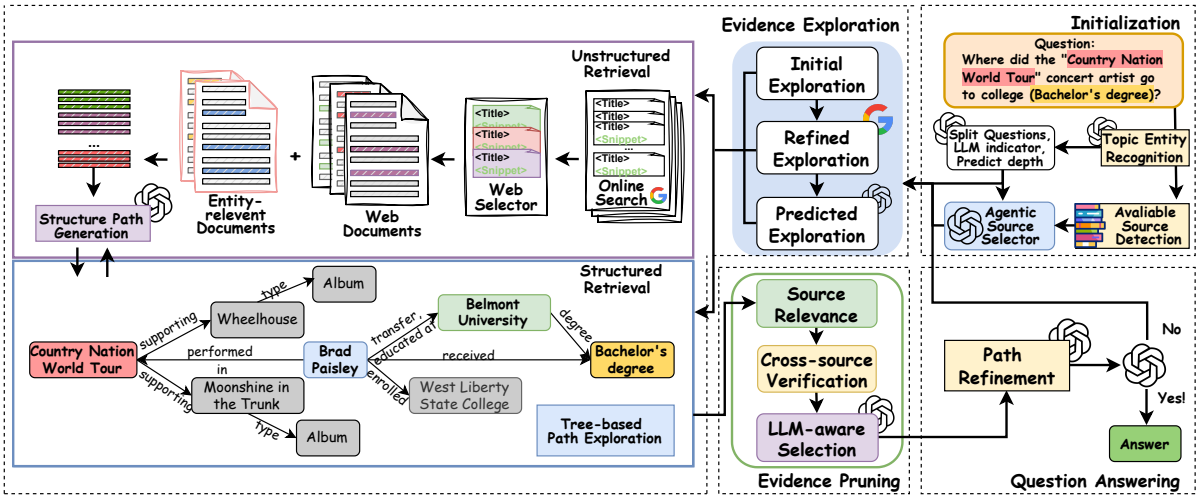


Figure 2: Overview of the HydraRAG architecture. Evidence exploration: after initialization (detailed in Figure 3), the model retrieves entity paths from diverse sources through three exploration phases. Evidence Pruning: HydraRAG applies a three-step evidence pruning procedure after each exploration phase. Question Answering: the pruned paths are then evaluated for question answering.

but their modules still face the same coverage and granularity limitations. ToG-2 (Ma et al., 2025b) queries all sources simultaneously, but it only retrieves one-hop neighbours and does not assess source reliability or cross-source consistency, making it unsuitable for multi-hop complex questions.

3 Preliminary

Consider a Knowledge Graph (KG) $\mathcal{G}(\mathcal{E}, \mathcal{R})$, where $\mathcal{E}$ and $\mathcal{R}$ represent the set of entities and relations, respectively. $\mathcal{G}(\mathcal{E}, \mathcal{R})$ contains abundant factual knowledge in the form of triples, i.e., $\mathcal{G}(\mathcal{E}, \mathcal{R}) = \{(e_h, r, e_t) \mid e_h, e_t \in \mathcal{E}, r \in \mathcal{R}\}$.

Definition 1 (Reasoning Path)**.** *Given a KG $\mathcal{G}$, a reasoning path within $\mathcal{G}$ is defined as a connected sequence of knowledge triples, represented as:* $\text{path}_{\mathcal{G}}(e_1, e_{l+1}) = \{(e_1, r_1, e_2), (e_2, r_2, e_3), ..., (e_l, r_l, e_{l+1})\}$*, where l denotes the length of the path, i.e.,* $\text{length}(\text{path}_{\mathcal{G}}(e_1, e_{l+1})) = l$.

Definition 2 (Entity Path)**.** *Given a KG $\mathcal{G}$ and an entity list $list_e = [e_1, e_2, e_3, \ldots, e_l]$, the entity path of $list_e$ is defined as a connected sequence of reasoning paths, which is denoted as* $\text{path}_{\mathcal{G}}(list_e) = \{\text{path}_{\mathcal{G}}(e_1, e_2), \text{path}_{\mathcal{G}}(e_2, e_3), \ldots, \text{path}_{\mathcal{G}}(e_{l-1}, e_l)\} = \{(e_s, r, e_t) | (e_s, r, e_t) \in \text{path}_{\mathcal{G}}(e_i, e_{i+1}) \wedge 1 \leq i < l\}$.

Knowledge Base Question Answering (KBQA) is a fundamental reasoning task based on KBs. Given a natural language question q and a KB $\mathcal{B}$, the objective is to devise a function f that predicts answers $a \in \text{Answer}(q)$ utilizing knowledge encapsulated in $\mathcal{B}$, i.e., $a = f(q, \mathcal{B})$.

4 Method

The HydraRAG framework integrates multiple knowledge sources to ensure comprehensive and reliable retrieval. The overview of HydraRAG is presented in Figure 2. All sources are first detected and agentically selected in Section 4.1, and then fully retrieved and augmented in Section 4.2. These sources include three categories. First, the **knowledge graph** provides the most accurate and structured evidence. For each question, we first extract an evidence subgraph $\mathcal{G}_q^s$ from every KG source (i.e., Freebase and WikiKG) and then merge these subgraphs into a single global evidence subgraph $\mathcal{G}_q$. Second, **wikipedia documents** supply semi-structured information[1]. We retrieve question-relevant Wiki document set using the topic entity set $\text{Topic}(q)$, forming $\text{Wiki} = \{\text{Doc}(e) \mid e \in \text{Topic}(q)\}$. Third, **web documents** capture real-time online results[2]. We issue an online search result set for q, yielding $\text{Web} = \text{OnlineSearch}(q)$, where each search result includes a web page title, description snippet, and URL. The faithfulness of web evidence is later assessed in Section 4.3.

4.1 Step I: Initialization

The initialization has three main stages, i.e., available evidence detection, question analysis, and agentic source selector. The framework is shown in Figure 3.

[1]HydraRAG uses the Wikipedia page of each topic entity $e \in \mathcal{G}_{\text{WikiKG}}$ as the initial document.

[2]HydraRAG uses Google Search by SeripAPI for online retrieval.

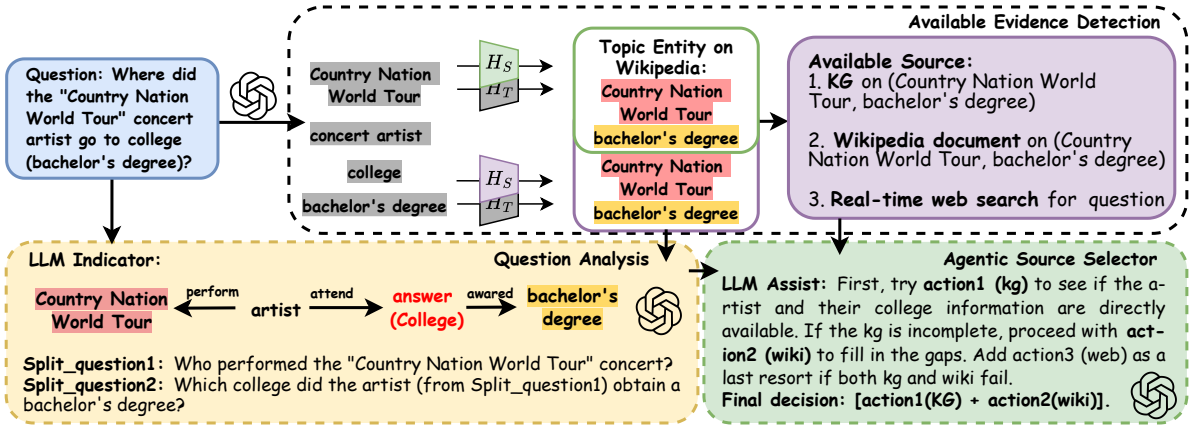


Figure 3: Overview of the initialization section. The initialization workflow diagram of three stages, i.e., available evidence detection, question analysis and agentic source selector.

Available evidence detection. Given a question q, HydraRAG first identifies candidate KBs, including knowledge graphs, web pages, and documents. To determine which sources are relevant to q, HydraRAG uses an LLM to extract potential topic entities. It then applies BERT-based similarity matching to align these entities with those in each source (e.g., $\mathcal{E} \in \{\mathcal{G}_{\text{freebase}}, \mathcal{G}_{\text{WikiKG}}\}$). As shown in Figure 3, we encode the extracted entities and all entities from a source into dense embeddings H_T and H_S, and compute a cosine similarity matrix to identify matches. For each extracted entity and each knowledge source, entities whose similarity exceeds a threshold form the set $\text{Topic}(q)$. Each source maintains its own $\text{Topic}(q)$; if $|\text{Topic}(q)| > 0$, the source is marked relevant and added to the total sources list $S_t \subseteq \{\text{KG, Wiki, Web}\}$ for further agentic selection. The $S_t = \{\text{Web}\}$ is considered as the initial setting. This set underlies the construction of the question-related subgraph and the preparation of documents in later steps.

Question analysis. To reduce hallucinations, the question analysis phase is divided into two parts and executed within a single LLM call using an example-based prompt (detailed in Appendix E). First, it breaks the complex question q into sub-questions, each linking one topic entity to the potential answer; solving these sub-questions together grounds the original query. Second, a solving skyline is generated, which lists all topic entities and predicts the answer's position in a single chain of thought derived from q. This skyline captures the relationships and order among the entities and the answer, transforming the complex question into a concise, simplified reasoning path. From this, we compute a predicted depth D_{predict}, defined as the maximum distance between the predicted answer and any topic entity. An example of question analysis, with $D_{\text{predict}} = 2$, is shown in Figure 3.

Agentic source selector. Most existing systems operate on a single KG or KB. Hybrid RAG methods (Ma et al., 2025b; Li et al., 2024c) can combine multiple information sources, but they typically query a fixed set (usually one or two) and ignore the question-specific trade-off between coverage and cost. Blindly querying every possible source greatly increases latency and computation.

To address this limitation, we introduce an agentic source selector. Given the total evidence source list S_t and question analysis result, an LLM-selected agent analyses the incoming question and chooses an initial source combination S_a that best balances three factors: (i) time sensitivity, (ii) reasoning complexity, and (iii) domain relevance. Only the selected sources $S_a \subseteq S_t$ are used in the initial exploration stage in Section 4.2.1, reducing cost while preserving answer quality.

4.2 Step II: Evidence Exploration

As discussed in Section 1, finding reasoning paths that include all topic entities is essential for deriving accurate answers. These paths act as **interpretable chains of thoughts**, showing both the answer and the inference steps leading to it.

However, the evidence needed to complete such paths is often distributed across sources. Combining these heterogeneous sources is therefore as important as path-finding itself. To discover high-quality paths while unifying evidence in a common format, the exploration is divided into three phases: initial exploration, refined exploration, and pre-

dicted exploration. In each exploration, retrievals from different sources are processed in parallel; After each phase, we apply path pruning and attempt to answer the question. If a valid path is found, the search terminates; otherwise, it proceeds to the next phase. Due to space constraints, the pseudo-code for exploration is provided in Appendix A.1.

4.2.1 Initial Exploration

To reduce LLM usage and narrow the search space, HydraRAG first explores agent-selected knowledge sources in parallel. Structured and unstructured inputs are processed independently: structured retrieval captures explicit relational facts, whereas unstructured retrieval supports more complex or implicit reasoning.

Structured retrieval. For structured retrieval, we first detect an evidence subgraph from KGs, then explore topic-entity paths.

Subgraph detection. Inspired by (Tan et al., 2025), we construct a $D_{\max}$-hop global evidence subgraph $\mathcal{G}_q$. For each topic entity, we retrieve all triples involving its $D_{\max}$-hop neighbors to incorporate relevant and faithful KG information into $\mathcal{G}_q^s$ from each knowledge source, i.e., $s \in \{\text{Freebase, WikiKG}\}$. To enhance knowledge coverage, we also merge multiple $\mathcal{G}_q^s$ into a global graph $\mathcal{G}_q$. To control information overload and reduce computation, we apply node and relation clustering, along with graph reduction techniques, to prune $\mathcal{G}_q$ effectively.

Tree-based path retrieval. Instead of the maximum depth $D_{\max}$, HydraRAG performs initial exploration at the predicted depth D_{predict}. Given the subgraph $\mathcal{G}_q$, the ordered topic entity set $\text{Topic}(q)$, the skyline indicator I_{sky}, and the depth $D = \min(D_{\text{predict}}, D_{\max})$, we identify candidate reasoning paths that include all topic entities in order. To avoid exhaustive search, we apply a tree-structured bidirectional breadth-first search (BiBFS) from each topic entity to extract a set of all potential entity paths, defined as: $\text{Paths}_I = \{p \mid |\text{Topic}(q)| \cdot (D-1) < \text{length}(p) \leq |\text{Topic}(q)| \cdot D\}$.

At each step, a **cross-score** (introduced in Section 4.3) is computed between the path, the skyline indicator, and retrieved documents to prune unpromising branches. Only the top-W_1 paths are retained as seeds for further expansion. This method enables efficient construction of high-quality candidate paths while maintaining interpretability. The pseudo-code for structured retrieval is detailed in Algorithm 1 of Appendix A.1.

Unstructured retrieval. For each document $\text{Doc}(e)$ associated with $e \in \text{Topic}(q)$, we retrieve text blocks, split them into smaller passages, and select the top-$W_{\max}$ sentences using a dense retrieval model (DRM). Instead of embedding the full query, HydraRAG uses the skyline indicator to emphasize structural relevance. Unlike ToG-2.0, which targets only one-hop relations, ours captures more complex reasoning, i.e., transitive multi-hop relations. The resulting sentences are used to prompt the LLM to construct new knowledge paths, which are summarized and added to Paths_I.

Web document retrieval. When offline documents and KGs are insufficient, HydraRAG performs online retrieval by issuing the question q to a search engine and prompting the LLM to select the top-$W_{\max}$ web results. These documents are then processed using the same DRM-based screening and path construction as in the offline setting. The pseudo-code and prompting for unstructured retrieval are detailed in Algorithm 2 of Appendix A.1 and Appendix E.

By combining KG-based, document-based, and web-based retrieval, HydraRAG generates a rich and interpretable path set as evidence, which is passed to the subsequent pruning stages.

4.2.2 Refined Exploration

Traditional KG-based reasoning typically reuses stored facts through a complex retrieval process. However, this approach often falls into fast-evolving or emerging information, which may not be adequately represented in the KG. To overcome this limitation, HydraRAG introduces a novel mechanism that leverages the LLM's ability to generate follow-up questions and refine the knowledge search. Specifically, HydraRAG prompts the LLM to generate a follow-up question, q_{new}, along with a new skyline indicator, I_{new}, which signals the additional information required beyond what is currently represented in the knowledge graph. The follow-up question q_{new} is designed to explicitly target the new information or emerging concepts, ensuring that the retrieval process captures relevant, up-to-date data. From this exploration, all the available knowledge sources S_t will be utilized for retrieval. Using q_{new}, we extract $\text{Topic}(q_{\text{new}})$ and perform unstructured retrieval: both new and historical documents are ranked according to I_{new}. For structured retrieval, we set the search depth $D = D_{\max}$ and use I_{new} to guide exploration within the KG. All paths retrieved in this phase are added to the refined entity path set Paths_R for further pruning.

4.2.3 Predict Exploration

In many RAG systems, LLMs merely rephrase facts rather than leveraging their own implicit knowledge. To address this, HydraRAG encourages LLMs to generate predictions using their path understanding and implicit knowledge, offering additional valuable insights. This involves creating new skyline indicators, I_{Pred}, for the predicted entities, $e \in \text{Predict}(q)$, and using text similarity to confirm and align them with $\mathcal{E}q \in \mathcal{G}q$. An entity list, $\text{List}_P(e) = \text{Topic}(q) + e$, is formed and ranked based on I_{pred} to enhance reasoning effectiveness.

For structured retrieval, predicted entity paths Paths_P are extracted from $\mathcal{G}_q$ at a fixed depth $D_{\max}$: $\text{Paths}_P = \{p \mid \text{length}(p) \leq |\text{Topic}(q)| \cdot D_{\max}\}$, where $p = \text{Path}_{\mathcal{G}_q}(\text{List}_P(e))$. For unstructured retrieval, the pair $(q, I_{\text{pred}(q)})$ is used to retrieve and score relevant sentences. The resulting paths are added to Paths_P. These paths with new indicators are evaluated similarly to the initial exploration and refined exploration phases. The prompting template is shown in Appendix E.

4.3 Step III: Evidence Pruning

Multi-source verification in pruning. Traditional LLM–QA pipelines typically perform two-step pruning: an embedding filter narrows down the candidate set, followed by an LLM agent that selects the most relevant evidence. However, this method assumes uniform evidence sources. When the corpus includes diverse modalities, such as structured knowledge graphs, semi-structured Wiki pages, and unstructured web content, pruning solely by relevance can either discard crucial facts or retain redundant information (over- or under-pruning).

The HydraRAG addresses this by adding a multi-source verification term to the relevance score. This term up-weights paths that are corroborated across heterogeneous sources and down-weights isolated claims from less reliable modalities. As a result, pruning balances topic relevance with cross-modal agreement, producing a compact yet reliable evidence set for downstream reasoning[3]. Due to space constraints, the pseudo-code for evidence pruning is summarized in Algorithm 4 of Appendix A.2.

Formally, let $\mathcal{C} = \{p_i\}_{i=1}^{N}$ as the candidate evidence paths, each associated with three scores $(s_i^{\text{rel}},\ s_i^{\text{ver}},\ s_i^{\text{llm}}) \in [0,1]^3$ denoting relevance, verification, and LLM compatibility, respectively. The derivation of each score is described below.

Source relevance. Given a query skyline indicator I and its topic-entity set $\text{Topic}(q)$, we compute a hybrid relevance score:

$$s_i^{\text{rel}} = \lambda_{\text{sem}} \cdot \underbrace{\cos\big(\mathbf{h}(I), \mathbf{h}(p_i)\big)}_{\text{semantic}} + \lambda_{\text{ent}} \cdot \underbrace{\text{Jaccard}\big(\text{Topic}(q), \text{Ent}(p_i)\big)}_{\text{entity overlap}},$$

where $\mathbf{h}(\cdot)$ denotes sentence-level embeddings by DRM, $\text{Ent}(p_i)$ extracts linked entities in p_i, and $\lambda_{\text{sem}} + \lambda_{\text{ent}} = 1$. The top-$W_1$ paths form a candidate pool $\widetilde{\mathcal{C}}$ for cross-source evaluation.

Cross-source verification. We estimate the reliability of each candidate path using three reliability features: (i) source reliability, (ii) corroboration from independent sources, and (iii) consistency with existing KG facts. Candidates in $\widetilde{\mathcal{C}}$ are grouped by provenance into $\mathcal{C}_{\text{KG}}$, $\mathcal{C}_{\text{Wiki}}$, and $\mathcal{C}_{\text{Web}}$. For each path p_i, the supporting external sources are: $\text{Supp}(p_i) = \{\text{src}(p_j) \mid \text{Sim}(p_i, p_j) \geq \gamma\}$, where $\text{src}(\cdot)$ returns the source type, and γ is a cosine similarity threshold. The reliability features inside are defined as:

$$f_1(p_i) = \rho_{\text{src}(p_i)}, \quad \rho_{\text{KG}} > \rho_{\text{Wiki}} > \rho_{\text{Web}},$$

$$f_2(p_i) = \frac{\min(|\,\text{Supp}(p_i)|, W_{\max})}{W_{\max}},$$

$$f_3(p_i) = \frac{|\text{Ent}(p_i) \cap \mathcal{E}_q|}{|\text{Ent}(p_i)|}, \quad \mathcal{E}_q \in \mathcal{G}_q.$$

The verification score is computed as $s_i^{\text{ver}} = \sum_{k=1}^{3} \alpha_k\ f_k(p_i)$, where coefficients α_k are non-negative and $\sum_k \alpha_k = 1$. Each candidate path in $\widetilde{\mathcal{C}}$ is then ranked by a **cross-score** that combines relevance and verification: $\text{cross-score}(p_i) = \alpha_{\text{cross}} \cdot s_i^{\text{rel}} + (1 - \alpha_{\text{cross}}) \cdot s_i^{\text{ver}}$. The top-$W_2$ paths are selected for the final LLM-driven pruning.

LLM-aware selection. At this stage, we prompt the LLM to score and select the top-$W_{\max}$ reasoning paths most likely to contain the correct answer. The specific prompt used to guide LLM in the selection phase can be found in Appendix E.

4.4 Step IV: Question Answering

Utilizing the pruned paths obtained in Section 4.3, we propose a two-step question-answering strategy, emphasizing deep thinking and slow reasoning.

Path Refinement. To ensure accurate reasoning and mitigate hallucinations, we prompt LLMs to refine the provided paths. By evaluating and selecting

[3]This module is model-agnostic; we demonstrate it with HydraRAG, but it can be inserted into any KG+RAG pipeline.

Table 1: Results of HydraRAG across all datasets, compared with the state-of-the-art (SOTA) with GPT-3.5-Turbo. The highest scores are highlighted in bold, while the second-best results are underlined for each dataset.

Type	Method	LLM	Multi-Hop KBQA				Single-Hop KBQA	Slot Filling	Open-Domain QA
			CWQ	WebQSP	AdvHotpotQA	QALD10-en	SimpleQA	ZeroShot RE	WebQuestions
LLM-only	IO prompt	GPT-3.5-Turbo	37.6	63.3	23.1	42.0	20.0	27.7	48.7
	CoT (Wei et al., 2022)		38.8	62.2	30.8	42.9	20.3	28.8	48.5
	SC (Wang et al., 2023)		45.4	61.1	34.4	45.3	18.9	45.4	50.3
Vanilla RAG	Web-based	GPT-3.5-Turbo	41.2	56.8	28.9	36.0	26.9	62.2	46.8
	Text-based		33.8	67.9	23.7	42.4	21.4	29.5	35.8
KG-based RAG	ToG (Sun et al., 2024)	GPT-3.5-Turbo	58.9	76.2	26.3	50.2	53.6	88.0	54.5
	ToG (Sun et al., 2024)	GPT-4	69.5	82.6	-	54.7	66.7	88.3	57.9
	PoG (Tan et al., 2025)	GPT-3.5-Turbo	74.7	93.9	-	-	80.8	-	81.8
Hybrid RAG	CoK (Li et al., 2024c)	GPT-3.5-Turbo	-	77.6	35.4	47.1	-	75.5	-
	ToG-2 (Ma et al., 2025b)		-	81.1	42.9	54.1	-	91.0	-
Proposed	HydraRAG-E	Llama-3.1-70B	71.3	89.7	48.4	70.9	80.4	95.6	76.8
	HydraRAG		75.6	93.0	55.2	76.0	85.9	94.2	81.4
	HydraRAG-E	GPT-3.5-Turbo	76.8	94.0	51.3	81.1	81.7	96.9	85.2
	HydraRAG		**81.2**	**96.1**	**58.9**	**84.2**	**88.8**	**97.7**	**88.3**

only relevant facts, the paths are summarized into concise, focused evidence, suitable for subsequent reasoning. Prompt details are in Appendix E.

CoT Answering. Following path refinement, the LLM employs a CoT prompting method to reason systematically through the refined evidence paths. It first checks whether they answer each subquestion and the full question. If the evaluation is positive, LLM generates the answer using the paths, along with the question and question analysis results as inputs, as shown in Figures 2. The prompts for evaluation and generation are in Appendix E. If negative, another exploration round begins. When all rounds end without a valid answer, the LLM replies using the given paths and its inherent knowledge. Additional details on the prompts can be found in Appendix E.

5 Experiment

In this section, we evaluate HydraRAG on seven benchmark KBQA datasets. Besides the HydraRAG proposed in this paper, we introduce **HydraRAG-E**, which randomly selects one relation from each edge in the clustered question subgraph to evaluate the impact of graph structure on KG involved LLM reasoning. The detailed experimental settings, including datasets, baselines, and implementations, can be found in Appendix C.

5.1 Main Results

Since HydraRAG leverages external knowledge, we first compare it against other RAG-based methods. As shown in Table 1, HydraRAG achieves SOTA results across all datasets, outperforming prior SOTA by an average of 10.8% and up to 30.1% on QALD10-en. Compared with ToG-2, a strong hybrid RAG baseline, HydraRAG achieves average improvements of 20.3%, up to 30.1% on QALD10-en. Against Llama3.1-70B, a weaker reasoning model, HydraRAG shows an average gain of 9.2% on 5 datasets, up to 21.9% on QALD10-en compared to previous GPT-3.5-based methods, and even surpasses the powerful GPT-4-based ToG baseline by 14. 4% on average, up to 23.5% on WebQuestions. This indicates HydraRAG significantly enhances the reasoning abilities of less powerful LLMs by providing faithful and interpretable cross-source knowledge paths. Additionally, compared to vanilla text/web-based RAG methods, HydraRAG shows average gains of 45.5%, up to 68.2% on ZeroShot RE.

When compared to methods without external knowledge (IO, CoT, SC), HydraRAG improves accuracy by 41.5% on average, up to 68.5% on SimpleQA. Notably, while vanilla RAG methods and LLM-only approaches show similar performance due to overlapping training corpora, HydraRAG achieves superior results using the almost same corpus, highlighting its advanced unstructured retrieval capability. The variant HydraRAG-E also surpasses existing SOTA methods by 6.8% on average, up to 24.0% on QALD10-en. These findings demonstrate HydraRAG is excellent for reasoning tasks, particularly for complex logical reasoning. By retrieving deeply and integrating the structural information of the question from diverse knowledge sources, it enhances the deep reasoning capabilities of LLMs, leading to superior performance.

5.2 Ablation Study

How does the effectiveness of HydraRAG vary with different LLM capabilities? We evaluated

Table 2: Performance of the IO baseline and HydraRAG across four datasets on different backbone models. The highest improvement is highlighted in bold, while the second-best results are underlined for each model.

Dataset	Llama-3.1-8B			Llama-3.1-70B			DeepSeek-v3			GPT-3.5-Turbo			GPT-4-Turbo		
	IO	HydraRAG	%↑	IO	HydraRAG	%↑	IO	HydraRAG	%↑	IO	HydraRAG	%↑	IO	HydraRAG	%↑
AdvHotpotQA	16.9	35.6	111	21.7	48.4	123	27.8	55.4	99.0	23.1	56.2	143	46.4	67.9	46.0
WebQSP	38.5	86.0	123	56.2	95.2	69.0	68.0	97.7	44.0	66.3	96.9	46.0	75.4	98.2	30.0
CWQ	29.8	62.4	109	35.4	83.2	135	38.7	84.5	118	39.2	84.0	114	45.3	89.7	**98.0**
ZeroShot RE	27.2	77.5	**185**	34.6	97.5	**182**	38.6	97.0	**151**	37.2	97.7	**163**	49.8	98.5	**98.0**

HydraRAG with five LLM backbones (LLama-3.1-8B, Llama-3.1-70B, Deepseek-v3, GPT-3.5-Turbo, GPT-4-Turbo) on three multi-hop datasets (AdvHotpotQA, WebQSP, CWQ) and one slot-filling dataset (ZeroShot RE). As shown in Table 2, HydraRAG improves performance across all models and datasets by an average of 109%. Notably, it boosts Llama-3.1-8B by 132% on average, up to 185% on ZeroShot RE. This brings weaker models close to and even surpasses the direct reasoning accuracy of GPT-4-Turbo, confirming that HydraRAG alleviates knowledge and comprehension bottlenecks. Stronger models also benefit from HydraRAG. GPT-3.5-Turbo and GPT-4-Turbo are improved on complex reasoning tasks, although the improvement decreases slightly as their inherent reasoning is already strong. Even so, HydraRAG yields a 98% improvement on CWQ and ZeroShot RE with the most capable LLMs. Overall, HydraRAG enables deeper knowledge retrieval and more reliable and interpretable reasoning across LLMs of varying strength, rather than relying solely on their inherent knowledge.

To further evaluate the performance of HydraRAG, we conduct additional ablation studies on search depth, agentic source selector, prompt setting, and knowledge sources. The detailed results are shown in Appendix B.1.

5.3 Effectiveness Evaluation

Effectiveness on incomplete KG. To evaluate how HydraRAG addresses KG incompleteness and the impact of graph quality on reasoning performance, we constructed KGs with varying completeness levels (0%, 30%, 50%, 80%, and 100%) on the AdvHotpotQA and CWQ. For each completeness level, we randomly selected a corresponding proportion of triples to build a new KG, with the remainder removed. Results in Figure 4 indicate that accuracy decreases slightly, rather than dramatically, as incompleteness increases. To investigate this trend, we analyze contributions from different KG completeness levels, with detailed analyses presented in Appendix B.3. The analysis reveals that at lower KG completeness, answers predominantly rely on wiki and web documents; as completeness increases, KG-based answers become dominant. This demonstrates that HydraRAG does not solely depend on KG data and effectively mitigates KG incompleteness issues, highlighting its adaptability.

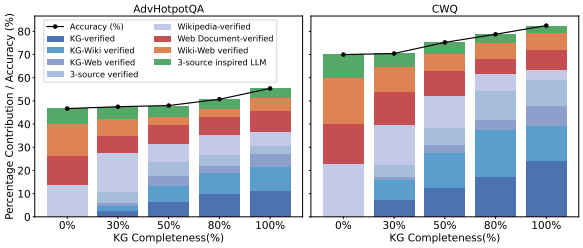


Figure 4: Accuracy and answer source composition by varying KG completeness on AdvHotpotQA and CWQ.

To further evaluate the performance, we perform additional experiments, including additional effectiveness evaluation on cross-source verification, multi-hop reasoning, multi-entity questions, and graph structure pruning in Appendix B.2; reasoning faithfulness analysis in Appendix B.3; error analysis in Appendix B.4; efficiency analysis in Appendix B.5; and case study on cross-verified interpretable reasoning in Appendix D. A detailed outline is shown in Appendix Outline.

6 Conclusion

In this work, we introduce HydraRAG, a structured source-aware retrieval method for faithful and transparent LLM reasoning. HydraRAG answers complex questions with agent-driven, structured and unstructured, multi-hop evidence exploration, ensuring every topic entity is linked across all knowledge corpora. Efficiency is enhanced by a tri-factor cross-source verification, scoring, and early pruning discards low-quality branches before any generation step. Extensive experiments on 7 datasets show that HydraRAG outperforms existing baselines, showcasing its superior reasoning capabilities and interoperability.

7 Ethics Statement

In this work, we employ LLMs as the final selector through LLM-aware selection, rather than for open-ended text generation. As a result, the ethical risks associated with our method are expected to be lower than using LLMs for text generation. However, recent studies indicate that CoT prompting may introduce ethical biases (Shaikh et al., 2023). Additionally, integrating evidence from multiple retrieval sources may also introduce or amplify ethical biases. In future work, we plan to systematically investigate the manifestation and impact of these biases in our method.

8 Limitation

The primary limitation of our proposed HydraRAG framework is its exclusive focus on character-based knowledge sources. HydraRAG does not incorporate external modalities such as images or videos, which can also contain substantial factual information. Integrating visual sources alongside textual evidence remains an important direction for future work and could further enhance the reasoning capabilities of the framework.

9 Acknowledgment

Xiaoyang Wang is supported by the Australian Research Council DP230101445 and DP240101322. Wenjie Zhang is supported by the Australian Research Council DP230101445 and FT210100303.

References

Jinheon Baek, Alham Fikri Aji, and Amir Saffari. 2023. Knowledge-augmented language model prompting for zero-shot knowledge graph question answering. *arXiv preprint arXiv:2306.04136*.

Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. 2013. Semantic parsing on Freebase from question-answer pairs. In *EMNLP*.

Maciej Besta, Nils Blach, Ales Kubicek, Robert Gerstenberger, Michal Podstawski, Lukas Gianinazzi, et al. 2024. Graph of thoughts: Solving elaborate problems with large language models. In *AAAI*.

Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: a collaboratively created graph database for structuring human knowledge. In *SIGMOD*, page 1247–1250.

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

Kaiyu Chen, Dong Wen, Hanchen Wang, Zhengyi Yang, Wenjie Zhang, and Xuemin Lin. 2025. Covering k-cliques in billion-scale graphs. In *WWW*.

Kaiyu Chen, Dong Wen, Wenjie Zhang, Ying Zhang, Xiaoyang Wang, and Xuemin Lin. 2024a. Querying structural diversity in streaming graphs. *Proc. VLDB Endow.*

Liyi Chen, Panrong Tong, Zhongming Jin, Ying Sun, Jieping Ye, and Hui Xiong. 2024b. Plan-on-graph: Self-correcting adaptive planning of large language model on knowledge graphs. In *NeurIPS*.

Yin Chen, Xiaoyang Wang, and Chen Chen. 2024c. Hyperedge importance estimation via identity-aware hypergraph attention network. In *CIKM*.

Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. 2023. Palm: Scaling language modeling with pathways. *JMLR*, 24(240):1–113.

Ziqi Ding, Gelei Deng, Yi Liu, Junchen Ding, Jieshan Chen, et al. 2025. Illusioncaptcha: A captcha based on visual illusion. In *WWW*.

Darren Edge, Ha Trinh, Newman Cheng, Joshua Bradley, et al. 2024. From local to global: A graph rag approach to query-focused summarization. *arXiv preprint arXiv:2404.16130*.

Yunfan Gao, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yixin Dai, Jiawei Sun, Haofen Wang, and Haofen Wang. 2023. Retrieval-augmented generation for large language models: A survey. *arXiv preprint arXiv:2312.10997*, 2:1.

Yizhang He, Kai Wang, Wenjie Zhang, Xuemin Lin, and Ying Zhang. 2023. Scaling up k-clique densest subgraph detection. In *SIGMOD*.

Yizhang He, Kai Wang, Wenjie Zhang, Xuemin Lin, Ying Zhang, and Wei Ni. 2025. Robust privacy-preserving triangle counting under edge local differential privacy. In *SIGMOD*.

Yiheng Hu, Xiaoyang Wang, Qing Liu, Xiwei Xu, Qian Fu, Wenjie Zhang, and Liming Zhu. 2025. Mmapg: A training-free framework for multimodal multi-hop question answering via adaptive planning graphs. *arXiv preprint arXiv:2508.16051*.

Chengkai Huang, Yu Xia, Rui Wang, Kaige Xie, Tong Yu, Julian McAuley, and Lina Yao. 2024a. Embedding-informed adaptive retrieval-augmented generation of large language models. In *COLING*.

Chengkai Huang, Tong Yu, Kaige Xie, Shuai Zhang, Lina Yao, and Julian McAuley. 2024b. Foundation models for recommender systems: A survey and new perspectives. *arXiv preprint arXiv:2402.11143*.

Jinhao Jiang, Kun Zhou, Zican Dong, Keming Ye, Wayne Xin Zhao, and Ji-Rong Wen. 2023. Structgpt: A general framework for large language model to reason over structured data. In *EMNLP*.

Fan Li, Xiaoyang Wang, Dawei Cheng, Wenjie Zhang, Ying Zhang, and Xuemin Lin. 2024a. Hypergraph self-supervised learning with sampling-efficient signals. In *IJCAI*.

Fan Li, Zhiyu Xu, Dawei Cheng, and Xiaoyang Wang. 2024b. Adarisk: Risk-adaptive deep reinforcement learning for vulnerable nodes detection. *IEEE TKDE*.

Xingxuan Li, Ruochen Zhao, Yew Ken Chia, Bosheng Ding, et al. 2024c. Chain-of-knowledge: Grounding large language models via dynamic knowledge adapting over heterogeneous sources. In *ICLR*.

Jie Ma, Zhitao Gao, Qi Chai, Wangchun Sun, Pinghui Wang, Hongbin Pei, et al. 2025a. Debate on graph: a flexible and reliable reasoning framework for large language models. In *AAAI*, pages 24768–24776.

Shengjie Ma, Chengjin Xu, Xuhui Jiang, Muzhi Li, et al. 2025b. Think-on-graph 2.0: Deep and interpretable large language model reasoning with knowledge graph-guided retrieval. In *ICLR*.

Michael Petrochuk and Luke Zettlemoyer. 2018. SimpleQuestions nearly solved: A new upperbound and baseline approach. In *EMNLP*, pages 554–558.

Fabio Petroni, Aleksandra Piktus, Angela Fan, Patrick Lewis, Majid Yazdani, Nicola De Cao, et al. 2020. Kilt: a benchmark for knowledge intensive language tasks. *arXiv preprint arXiv:2009.02252*.

Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence embeddings using Siamese BERT-networks. In *EMNLP*.

Bhaskarjit Sarmah, Dhagash Mehta, Benika Hall, Rohan Rao, Sunil Patel, and Stefano Pasquali. 2024. Hybridrag: Integrating knowledge graphs and vector retrieval augmented generation for efficient information extraction. In *Proceedings of the 5th ACM International Conference on AI in Finance*, pages 608–616.

Omar Shaikh, Hongxin Zhang, William Held, Michael Bernstein, and Diyi Yang. 2023. On second thought, let's not think step by step! bias and toxicity in zero-shot reasoning. In *ACL*, pages 4454–4470.

Zhihong Shao, Yeyun Gong, Yelong Shen, Minlie Huang, Nan Duan, and Weizhu Chen. 2023. Enhancing retrieval-augmented large language models with iterative retrieval-generation synergy. *arXiv preprint arXiv:2305.15294*.

Qing Sima, Jianke Yu, Xiaoyang Wang, Wenjie Zhang, Ying Zhang, and Xuemin Lin. 2025. Deep overlapping community search via subspace embedding. In *SIGMOD*.

Jiashuo Sun, Chengjin Xu, Lumingyuan Tang, Saizhuo Wang, Chen Lin, Yeyun Gong, Lionel Ni, Heung-Yeung Shum, and Jian Guo. 2024. Think-on-graph: Deep and responsible reasoning of large language model on knowledge graph. In *ICLR*.

Alon Talmor and Jonathan Berant. 2018. The web as a knowledge-base for answering complex questions. In *NAACL*.

Xingyu Tan, Chengyuan Guo, Xiaoyang Wang, Wenjie Zhang, and Chen Chen. 2023a. Maximum fairness-aware (k, r)-core identification in large graphs. In *Australasian Database Conference*. Springer.

Xingyu Tan, Jingya Qian, Chen Chen, Sima Qing, Yanping Wu, Xiaoyang Wang, and Wenjie Zhang. 2023b. Higher-order peak decomposition. In *CIKM*.

Xingyu Tan, Xiaoyang Wang, Qing Liu, Xiwei Xu, Xin Yuan, and Wenjie Zhang. 2025. Paths-over-graph: Knowledge graph empowered large language model reasoning. In *Proceedings of the ACM on Web Conference 2025*, pages 3505–3522.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Ricardo Usbeck, Xi Yan, Aleksandr Perevalov, Longquan Jiang, Julius Schulz, Angelie Kraft, et al. 2024. Qald-10–the 10th challenge on question answering over linked data: Shifting from dbpedia to wikidata as a kg for kgqa. *Semantic Web*.

Denny Vrandečić and Markus Krötzsch. 2014. Wikidata: a free collaborative knowledgebase. *Communications of the ACM*.

Jianwei Wang, Kai Wang, Xuemin Lin, Wenjie Zhang, and Ying Zhang. 2024a. Efficient unsupervised community search with pre-trained graph transformer. *arXiv preprint arXiv:2403.18869*.

Jianwei Wang, Kai Wang, Ying Zhang, Wenjie Zhang, Xiwei Xu, and Xuemin Lin. 2025a. On llm-enhanced mixed-type data imputation with high-order message passing. *arXiv preprint arXiv:2501.02191*.

Jinghao Wang, Yanping Wu, Xiaoyang Wang, Chen Chen, Ying Zhang, and Lu Qin. 2025b. Effective influence maximization with priority. In *WWW*.

Jinghao Wang, Yanping Wu, Xiaoyang Wang, Ying Zhang, Lu Qin, Wenjie Zhang, and Xuemin Lin. 2024b. Efficient influence minimization via node blocking. *Proc. VLDB Endow.*

Xuezhi Wang, Jason Wei, Dale Schuurmans, Quoc Le, Ed Chi, Sharan Narang, Aakanksha Chowdhery, and Denny Zhou. 2023. Self-consistency improves chain of thought reasoning in language models. In *ICLR*.

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

Yanping Wu, Renjie Sun, Xiaoyang Wang, Dong Wen, Ying Zhang, Lu Qin, and Xuemin Lin. 2024. Efficient maximal frequent group enumeration in temporal bipartite graphs. *Proc. VLDB Endow.*

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 *ICLR*.

Xi Ye and Greg Durrett. 2022. The unreliability of explanations in few-shot prompting for textual reasoning. In *NeurIPS*.

Wen-tau Yih, Matthew Richardson, Chris Meek, Ming-Wei Chang, and Jina Suh. 2016. The value of semantic parse labeling for knowledge base question answering. In *ACL*, pages 201–206.

Haozhe Yin, Kai Wang, Wenjie Zhang, Ying Zhang, Ruijia Wu, and Xuemin Lin. 2025. Efficient computation of hyper-triangles on hypergraphs. *arXiv preprint arXiv:2504.02271*.

Zian Zhai, Fan Li, Xingyu Tan, Xiaoyang Wang, and Wenjie Zhang. 2025. Graph is a natural regularization: Revisiting vector quantization for graph representation learning. *arXiv preprint arXiv:2508.06588*.

Zian Zhai, Sima Qing, Xiaoyang Wang, and Wenjie Zhang. 2024. Adapting unsigned graph neural networks for signed graphs: A few-shot prompt tuning approach. *arXiv preprint arXiv:2412.12155*.

Jiujing Zhang, Shiyu Yang, Dian Ouyang, Fan Zhang, Xuemin Lin, and Long Yuan. 2023. Hop-constrained st simple path enumeration on large dynamic graphs. In *ICDE*, pages 762–775.

Yunrui Zhang, Gustavo Batista, and Salil S Kanhere. 2025. Instance-wise monotonic calibration by constrained transformation. In *UAI*.

Appendix Outline

A Algorithm

A.1 Exploration

We summarize the comprehensive algorithmic procedure for evidence exploration detailed in Section 4.2 as presented in Algorithm 1-3.

Algorithm 1: Structured_Retrieval

Input : Source KG ($\mathcal{G}$), Question evidence subgraph $\mathcal{G}_q$, select source (S_c), topic entities (List$_T$), skyline indicator (I), depth (D), width (W)
Output : Reasoning KG paths (Paths$_{\text{KG}}$), evidence subgraph($\mathcal{G}_q$)

if *KG* $\in S_c$ **then**
  if $\mathcal{G}_q$ *is* $\emptyset$ **then**
    $\mathcal{G}_q \leftarrow$ Subgraph_Detection($\mathcal{G}$, List$_T$, D_{max});
    $\mathcal{G}_q \leftarrow$ KG_Summary($\mathcal{G}_q$);
  Paths$_{\text{KG}} \leftarrow$ Tree_based_Path_Retrieval($Q, I, W, D,$ List$_T, \mathcal{G}_q$);
Return Paths$_{\text{KG}}$, $\mathcal{G}_q$;

Tree_based_Path_Retrieval($Q, I, W, D_{\max}$, List$_T, \mathcal{G}_q$)
$D \leftarrow 1$; $Paths \leftarrow \emptyset$; $E_{\text{outter}} \leftarrow$ List$_T$;
while $D \leq D_{\max}$ **do**
  $E_{\text{outter}'} \leftarrow \emptyset$;
  for each $e \in E_{outter}$ **do**
    P, outter $\leftarrow$ Expand_One_Hop(e);
    Paths $\leftarrow$ Paths $\cup$ P;
    $E_{\text{outter}'} \leftarrow E_{\text{outter}'} \cup$ outter;
  while $|Paths| > W$ **do**
    Relevant_Pruning(Paths, Q, I, W);
    $E_{\text{outter}'} \leftarrow$ IntersectMatchUpdate(Paths, $E_{\text{outter}'}$);
  $E_{\text{outter}} \leftarrow E_{\text{outter}'}$; $D \leftarrow D + 1$;
Return Paths;

Algorithm 2: Unstructured_Retrieval

Input : Select source (S_c), topic entities (Topic(q)), question (q), skyline indicator (I), width (W)
Output : Summarized wiki structured paths (Paths$_{\text{Wiki}}$), summarized web structured paths(Paths$_{\text{Web}}$)

if *Web* $\in S_c$ **then**
  WebLinks $\leftarrow$ OnlineSearch(q);
  TopURLs $\leftarrow$ Prompt$_{\text{select}}$(WebLinks, W, q, I);
  Docs $\leftarrow$ URLs_Process(TopURLs);
  SelectSentence $\leftarrow$ DRM(Docs, I,W) ;
  Paths$_{\text{Web}}$ $\leftarrow$Prompt$_{\text{StructuredPathGen}}$(SelectSentence, Topic($q$), I) ;
if *Wiki* $\in S_c$ **then**
  for each e $\in$ Topic(q) **do** Docs $\leftarrow$ Docs $\cup$ Doc(e);
  SelectSentence $\leftarrow$ DRM(Docs, I,W) ;
  Paths$_{\text{Wiki}}$$\leftarrow$Prompt$_{\text{StructuredPathGen}}$(SelectSentence, Topic($q$), I) ;
Prompt$_{\text{PathSummary}}$(Paths$_{\text{Wiki}}$, Paths$_{\text{Web}}$, I);
Return Paths$_{\text{Wiki}}$, Paths$_{\text{Web}}$;

Algorithm 3: Evidence_Exploration

Input : Source KG ($\mathcal{G}$),question and split question ($Q = q$ + q_{split}), agentic select source (S_a), total available source (S_t), topic entities (Topic(q)), skyline indicator (I_{Sky}), predict depth (D_{predict}), maximum depth ($D_{\max}$), maximum width ($W_{\max}$)
Output : HydraRAG answers ($a(q)$), final reasoning path (Paths$_F(q)$)

`/* Initial exploration procedure */`
List$_T \leftarrow$ Reorder(Topic(q), I_{Sky});
$D_{\text{predict}} \leftarrow$ min($D_{\text{predict}}, D_{\max}$); $\mathcal{G}_q \leftarrow \emptyset$;
Paths$_{\text{KG}}, \mathcal{G}_q \leftarrow$ **Structured_Retrieval** ($\mathcal{G}, \mathcal{G}_q, S_a,$ List$_T, I_{\text{Sky}}, D_{\text{predict}}, W_1$);
Paths$_{\text{Wiki}}$, Paths$_{\text{Web}} \leftarrow$ **Unstructured_Retrieval** (S_a, Topic(q), $q, I_{\text{Sky}}, W_{\max}$);
Paths$_I \leftarrow$ Paths$_{\text{KG}}$ + Paths$_{\text{Wiki}}$ + Paths$_{\text{Web}}$;
Paths$_I \leftarrow$ Evidence_Pruning(Paths$_I, Q, I_{\text{Sky}}, W_{\max}$, Topic($q$), $\mathcal{G}_q$);
Answer, Paths$_I \leftarrow$ Question_Answering(Paths$_{\text{I}}$, Q, I_{Sky});
if `"{Yes}"` in Answer **then return** Answer, Paths$_I$;

`/* Refined exploration procedure */`
$q_{\text{new}}, I_{new} \leftarrow$ Prompt$_{\text{newQ}}$(Paths$_I, Q, I_{\text{Sky}}$, Topic(q));
Paths$_{\text{KG}}, \mathcal{G}_q \leftarrow$ **Structured_Retrieval** ($\mathcal{G}, \mathcal{G}_q, S_t,$ List$_T, I_{\text{new}}, D_{\max}, W_1$);
Paths$_{\text{Wiki}}$, Paths$_{\text{Web}} \leftarrow$ **Unstructured_Retrieval** (S_t, Topic(q_{new}), $q_{\text{new}}, I_{\text{new}}, W_{\max}$);
Paths$_R \leftarrow$ Paths$_{\text{KG}}$ + Paths$_{\text{Wiki}}$ + Paths$_{\text{Web}}$;
Paths$_R \leftarrow$ Evidence_Pruning(Paths$_R, Q, I_{\text{Sky}}, W_{\max}$, Topic($q$), $\mathcal{G}_q$);
Answer, Paths$_R \leftarrow$ Question_Answering(Paths$_{\text{R}}$, Q, I_{Sky});
if `"{Yes}"` in Answer **then return** Answer, Paths$_R$;

`/* Predicted exploration procedure */`
Paths$_P \leftarrow \emptyset$;
Predict(q) $\leftarrow$LLMPredict(Paths$_I$ + Paths$_R, Q, I_{\text{Sky}}$);
for each $e, I_{Pred(e)} \in$ *Predict*(q) **do**
  List$_P \leftarrow$ Reorder (Topic$_q$ + e, $I_{\text{Pred}(e)}$);
  Paths$_{\text{KG}}, \mathcal{G}_q \leftarrow$ **Structured_Retrieval** ($\mathcal{G}, \mathcal{G}_q, S_t,$ List$_P, I_{\text{Pred}(e)}, D_{\max}, W_1$);
  Paths$_{\text{Wiki}}$, Paths$_{\text{Web}} \leftarrow$ **Unstructured_Retrieval** (S_t, List$_P$, $q, I_{\text{Pred}(e)}, W_{\max}$);
  Paths$_P \leftarrow$ Paths$_P$ + Paths$_{\text{KG}}$ + Paths$_{\text{Wiki}}$ + Paths$_{\text{Web}}$;
Paths$_P \leftarrow$ Evidence_Pruning(Paths$_P, Q, I_{\text{Sky}}, W_{\max}$, Topic($q$), $\mathcal{G}_q$);
Answer, Paths$_P \leftarrow$ Question_Answering(Paths$_{\text{P}}$, Q, I_{Sky});
if `"{Yes}"` in Answer **then return** Answer, Paths$_P$;

Paths$_F \leftarrow$ Paths$_I$ + Paths$_R$ + Paths$_P$;
Paths$_F \leftarrow$ Evidence_Pruning(Paths$_F, Q, I_{\text{Sky}}, W_{\max}$, Topic($q$), $\mathcal{G}_q$);
Answer, Paths$_F \leftarrow$ Question_Answering(Paths$_{\text{F}}$, Q, I_{Sky});
Return Answer, Paths$_F$;

A.2 Evidence pruning

We summarize the comprehensive algorithmic procedure of evidence pruning detailed in Section 4.3 as presented in Algorithm 4.

Algorithm 4: Evidence_Pruning

Input : candidate paths (C), question and split question ($Q = q$ + q_{split}), skyline indicator (I), width($W_{\max}$), topic entities (Topic(q)), KG ($\mathcal{G}_q$)

Output : Pruned candidate paths ($Paths_c$)

/* **Step 1: Compute relevance scores** */

$S_{\text{rel}}, S_{\text{ver}}$, Cross_Score $\leftarrow \emptyset$;

for each $p_i \in C$ **do**

  semantic_score $\leftarrow$ Semantic_DRM(I, p_i);

  entity_overlap $\leftarrow$ Jaccard(Topic(q), Ent(p_i));

  $S_{\text{rel}[p_i]} \leftarrow$ $\lambda_{\text{sem}} \cdot$ semantic_score + λ_{ent}·entity_overlap;

C_{tilde} = Select_Top_Paths(C, S_{rel}, W_1);

/* **Step 2: Compute cross-source verification scores** */

for each $p_i \in C_{tilde}$ **do**

  source_prior $\leftarrow$ get_source_prior(p_i);

  supporting_sources$\leftarrow$ get_supporting_sources(p_i, C_{tilde});

  source_agreement$\leftarrow$ $min(|\text{supporting\_sources}|, W_{\max})/W_{\max}$;

  entity_alignment $\leftarrow$ |Ent(p_i) $\cap$ E_q| / |Ent(p_i)|;

  $S_{\text{ver}}[p_i] \leftarrow \alpha_1 \cdot$ source_prior + $\alpha_2 \cdot$ source_agreement + $\alpha_3 \cdot$ entity_alignment;

for each $p_i \in C_{tilde}$ **do**

  Cross_Score[p_i] $\leftarrow$ $\alpha_{\text{cross}} \cdot S_{\text{rel}}[p_i] + (1 - \alpha_{\text{cross}}) \cdot S_{\text{ver}}[p_i]$;

Paths$_F$ = Select_Top_Paths(C_{tilde}, Cross_Score, W_2);

/* **Step 3: LLM-aware final selection** */

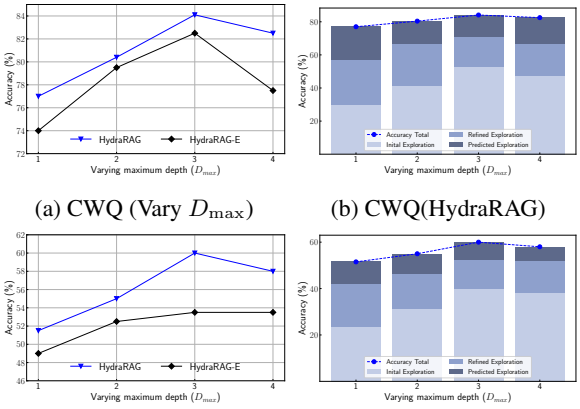
Paths$_F$ = Prompt$_{\text{SelectPath}}$(Paths$_F$,Q, I, $W_{\max}$);

Return Paths$_F$;

B Experiment

B.1 Addtioanl Ablation Study

Does search depth matter? As described, the dynamic deep search in HydraRAG is limited by the maximum depth, $D_{\max}$. To analyze how $D_{\max}$ affects performance, we conducted experiments varying depth from 1 to 4. Results (Figures 5(a) and (c)) show that deeper searches improve performance, but gains diminish beyond depth 3, as excessive depth increases hallucinations and complicates path management. Figures 5(b) and (d), showing which exploration phase the answer is generated from, reveal that higher depths reduce the effectiveness of both refined and predicted exploration. Hence, we set $D_{\max} = 3$ for optimal balance between performance and efficiency. Notably, even at lower depths, HydraRAG maintains strong performance by effectively integrating diverse sources and leveraging LLMs' inherent knowledge through the refined and predictive exploration procedures.



(a) CWQ (Vary $D_{\max}$) (b) CWQ(HydraRAG)

(c) AdvHotpot(Vary $D_{\max}$) (d) AdvHotpot(HydraRAG)

Figure 5: The accuracy of HydraRAG and HydraRAG-E among CWQ and AdvHotpotQA datasets by varying different $D_{\max}$.

How does the agentic source selector affect performance? To reduce redundant computational cost when using multiple sources, we incorporate an agentic source selector that adaptively selects sources based on the evolving needs of the reasoning process. To assess its impact, we perform an ablation study comparing HydraRAG and HydraRAG-E with and without the source selector. We evaluate both the accuracy and the average token input during the path pruning stage. As shown in Table 3, integrating the agentic source selector substantially improves performance. For instance, on the CWQ dataset, HydraRAG-E achieves a 24.0% absolute accuracy improvement, while reducing token input by 36.8%. Similar trends are observed across other settings. These improvements stem from the HydraRAG 's ability to dynamically identify and invoke only the most relevant sources. During initial exploration, the selector analyzes the question intent to determine the most suitable sources. In subsequent stages, it further distinguishes between sources used to expand coverage (refined exploration) and those used to increase depth for precise answer prediction (predicted exploration). This adaptive strategy avoids the naïve composition of all sources and leads to more efficient and effective reasoning.

Table 3: Performance comparison of HydraRAG and HydraRAG-E with and without agentic source selector on CWQ and WebQSP datasets.

Method	Evaluation	CWQ	WebQSP
HydraRAG			
w/ agentic source selector	Accuracy	**87.0**	**95.0**
	Token Input	73,240	91,097
w/o agentic source selector	Accuracy	71.0	92.0
	Token Input	98,748	145,411
HydraRAG-E			
w/ agentic source selector	Accuracy	**85.0**	**92.2**
	Token Input	73,519	97,055
w/o agentic source selector	Accuracy	61.0	87.0
	Token Input	116,240	49,399

Table 5: Performance comparison of HydraRAG with different knowledge sources and retrieval components across four multi-hop datasets.

Source Setting	CWQ	AdvHotpotQA	WebQSP	QALD
w/ all sources	78.2	55.2	90.3	78.0
w/o Freebase	63.7	51.0	79.2	70.0
w/o WikiKG	70.0	54.0	86.7	69.0
w/o Web document	75.0	52.3	86.0	75.3
w/o Wiki document	74.0	50.4	84.0	68.2
w/o Freebase & WikiKG	60.4	50.1	74.0	64.0
w/o Web & Wiki document	73.6	42.4	86.0	71.7

How do path refinement prompts affect performance? Inspired by GoT (Besta et al., 2024), we use path refinement prompts to integrate information from all sources, reduce LLM hallucinations from irrelevant or lengthy paths, and decrease computational costs. To assess their impact, we conduct an ablation study comparing HydraRAG and HydraRAG-E with and without path refinement, measuring both accuracy and average token input during path pruning. As shown in Table 4, path refinement increases accuracy by up to 11% (on CWQ with HydraRAG-E), meanwhile reducing token input by 54%. These results indicate that path refinement could effectively minimize LLM hallucinations, improve LLM understanding of explored paths, facilitate answer retrieval, enable earlier termination, and reduce overall cost.

Table 4: Performance comparison of HydraRAG and HydraRAG-E with and without path refinement on CWQ and WebQSP datasets.

Method	Evaluation	CWQ	WebQSP
HydraRAG			
w/ Path refinement	Accuracy	**87.0**	**95.0**
	Token Input	73,240	91,097
w/o Path refinement	Accuracy	79.0	93.0
	Token Input	134,554	107,516
HydraRAG-E			
w/ Path refinement	Accuracy	**85.0**	**92.2**
	Token Input	73,519	97,055
w/o Path refinement	Accuracy	74.0	90.0
	Token Input	159,678	107,762

How do different knowledge sources affect the performance of HydraRAG? To evaluate the impact of different knowledge sources and retrieval components, we conduct ablation experiments by excluding individual sources and modules on all multi-hop QA datasets. Results show that Freebase contributes most to CWQ and WebQSP, while WikiKG and wiki documents are more important for AdvHotpotQA and QALD, likely due to varying knowledge backgrounds and overlaps in each dataset. Notably, removing any single source or retrieval module does not cause a dramatic drop in performance, demonstrating the robustness of our framework in integrating heterogeneous evidence. HydraRAG effectively leverages complementary information from both structured and unstructured sources, mitigating the impact of missing components. Even without structured retrieval (Freebase and WikiKG), HydraRAG maintains high accuracy and still outperforms naive text and web-based RAG methods using the same corpus. This highlights the strength of our structure-aware integration in extracting and organizing information from unstructured evidence, bridging the gap between text-based and structure-based approaches. Overall, these results underline the benefit of our unified multi-source framework, which ensures stable, high performance by flexibly combining evidence from diverse sources.

B.2 Additional Effectiveness Evaluation

Effectiveness on cross-source verification. To evaluate the effectiveness of cross-source verification, we compare it with the standard question-relevant approach commonly used in hybrid RAG. For a fair comparison, we use the same embedding model (SBERT) and beam search width W_2, replacing only the first two evidence pruning steps (source relevance and cross-source verification). We report accuracy, average token input, and number of LLM calls for both pruning strategies on the CWQ and AdvHotpotQA datasets (Table 6). Results show that cross-source verification improves accuracy by up to 22% on CWQ and reduces token cost by up to 41.8% on AdvHotpotQA, using the same knowledge corpus. This improvement arises because relevance-only pruning often retains noisy paths and prunes correct ones, forcing extra exploration and incurring higher LLM costs. These results demonstrate the effectiveness of cross-source verification and its potential as a solution for efficient multi-source RAG.

Table 6: Evaluation Results for CWQ and AdvHotpotQA with cross-source verification and question relevance pruning.

Method	Evaluation	CWQ	AdvHotpotQA
w/ Cross-source verification	Accuracy	84.0	60.0
	Token Input	114,023	14,089
	LLM Calls	8.0	9.0
w/ Question Relevant Only	Accuracy	62.0	52.1
	Token Input	157,850	24,193
	LLM Calls	7.9	9.6

Effectiveness on multi-hop reasoning. To assess HydraRAG 's performance on multi-hop reasoning tasks, we analyze accuracy by grouping questions according to the length of their ground-truth SPARQL queries. We randomly sample 1,000 questions each from the CWQ and WebQSP datasets and determine reasoning length by counting the number of relations in each ground-truth query (see Figure 6). We then evaluate HydraRAG and HydraRAG-E across varying reasoning lengths to understand their effectiveness under varying query complexities. As shown in Figure 7, both models maintain high and stable accuracy across different lengths, with HydraRAG achieving up to 98.6% accuracy even at the highest length levels in WebQSP. Notably, HydraRAG can correctly answer questions with ground-truth lengths of eight or more by exploring novel paths and integrating LLM knowledge, rather than strictly matching the ground-truth path. These results highlight the effectiveness of HydraRAG in handling complex multi-hop reasoning tasks.

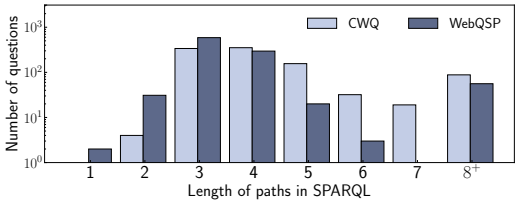
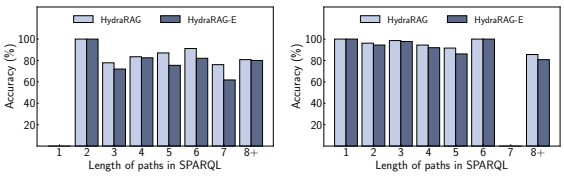


Figure 6: The lengths of the ground-truth SPARQL queries within the CWQ and WebQSP datasets.



(a) CWQ (b) WebQSP

Figure 7: Accuracy of HydraRAG and HydraRAG-E on the CWQ and WebQSP datasets, categorized by the different lengths of the ground-truth answers for each question.

Table 7: Average number of entities from Freebase, WikiKG, and after graph fusion and reduction for three datasets.

	CWQ	AdvHotpotQA	QALD10-en
Ave. Entity Number from Freebase	2,289,881	1,329,012	2,753,230
Ave. Entity Number from WikiKG	160,762	128,766	389,360
Ave. Entity Number after Reduction	128,352	399,785	587,110

Table 8: Performance of HydraRAG and HydraRAG-E on multi-entity and single-entity questions of all datasets. The symbol '-' indicates no multi-entity question inside.

Question Set	CWQ	WebQSP	AdvHotpot QA	QALD10-en	Simple Questions	ZeroShot RE	Web Questions
HydraRAG w/ GPT-3.5-Turbo							
Single-entity	71.9	96.2	56.8	83.1	89.0	97.7	88.2
Multi-entity	92.0	93.1	61.5	86.5	-	83.6	82.8
HydraRAG-E w/ GPT-3.5-Turbo							
Single-entity	68.6	94.0	51.5	79.1	87.3	97.3	85.4
Multi-entity	89.7	89.7	57.1	84.2	-	80.3	82.8

Effectiveness on graph structure pruning. To assess the effectiveness of our graph fusion and reduction strategy, we report the average number of unique entities from Freebase and WikiKG before fusion, and the total number of entities remaining after fusion and graph reduction, as shown in Table 7. For each dataset, we first fuse overlapping entities from multiple knowledge sources, then apply the graph reduction method described in Section 4.1 to remove irrelevant nodes prior to path exploration. The results demonstrate a substantial reduction in the number of entities across all datasets. For example, in CWQ, the initial combined entity count from Freebase and WikiKG exceeds 2.4 million, but this is reduced to only 128,352 after fusion and pruning. Similar trends are observed for AdvHotpotQA and QALD10-en. This reduction indicates that a significant portion of entities are either redundant or irrelevant to the questions under consideration. By eliminating such entities before downstream reasoning, our approach improves computational efficiency and focuses exploration on the most relevant subgraphs. Overall, these results verify the effectiveness of combining graph fusion and reduction for constructing compact and informative question-specific subgraphs.

Effectiveness on multi-entity questions. Graphs are widely used to model complex relationships among different entities (Chen et al., 2025, 2024a,c; Wu et al., 2024; Zhang et al., 2023, 2025). KGs store triples, making entity links explicit (He et al., 2023, 2025; Zhai et al., 2024, 2025; Yin et al., 2025). Building on this foundation, we further examine how well HydraRAG can leverage such structural representations when dealing with questions that involve multiple entities.

To evaluate the performance of HydraRAG on multi-entity questions, we report the accuracy on all test sets by categorizing questions based on the number of topic entities. The results, shown in Table 8, demonstrate that, despite the increased complexity of multi-entity questions compared to single-entity ones, HydraRAG maintains excellent accuracy, achieving up to 93.1% on the WebQSP dataset. This underscores the effectiveness of our structure-based model in handling complex multi-entity queries.

B.3 Reasoning Faithfulness Analysis

Evidence of answer exploration sources. We analyze the sources of evidence supporting correct answers on four multi-hop datasets to assess the effectiveness of cross-verification and the distribution of knowledge supervision in HydraRAG, as shown in Figure 8. Specifically, all generated answers are classified based on the verification source: KG-verified, Wikipedia-verified, web document-verified, as well as combinations such as KG-Wiki, KG-Web, Wiki-Web, and those verified by all three sources. In addition, when the paths generated from all external sources are insufficient to reach the answer, and the LLM supplements the reasoning using its inherent knowledge, such answers are categorized as LLM-inspired. The analysis reveals that over 95% of answers are supported by external knowledge supervision, confirming that HydraRAG primarily grounds its reasoning in verifiable sources. Furthermore, up to 56% of correct answers are jointly verified by at least two distinct knowledge sources. This highlights the strength of HydraRAG in leveraging multi-source evidence, an essential for faithful and interpretable reasoning.

Among answers with only single-source support, knowledge graph (KG) evidence dominates, accounting for as much as 95.7% of sole-source supervision in WebQSP. This underscores the high reliability and factual precision of KGs compared to other sources. Compared with previous methods that simply combine LLM internal knowledge with external sources (Ma et al., 2025b), HydraRAG further enhances reliability by enabling mutual cross-verification between all sources. This multi-source evaluation mechanism reduces the risk of unsupported or spurious answers. These results highlight that HydraRAG is a faithful reasoning framework that not only prioritizes evidence-based answers but also ensures high accuracy and interpretability by integrating and cross-validating structured and unstructured knowledge.

Overlap ratio between explored paths and ground-truth paths. We analysis correctly answered samples from CWQ and WebQSP to examine the overlap ratio between paths P explored by HydraRAG and ground-truth paths P_G from SPARQL queries. The overlap ratio is defined as the proportion of shared relations to total relations in the ground-truth SPARQL path:

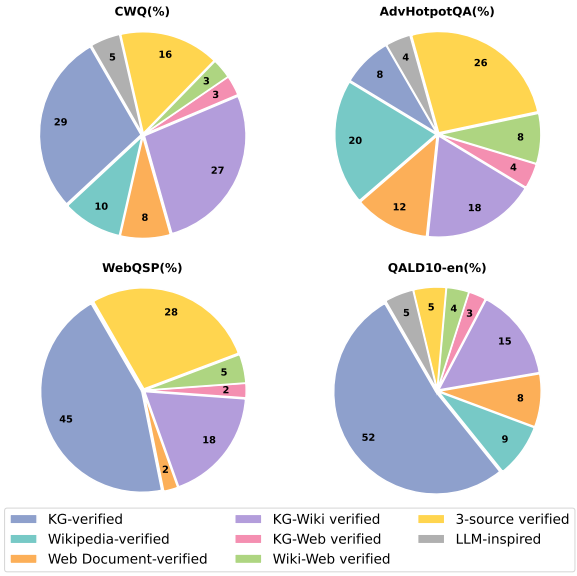
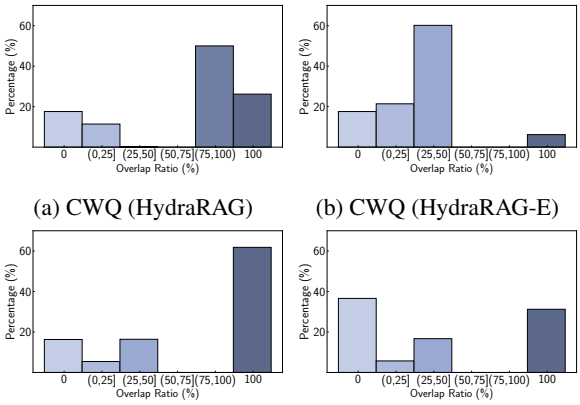


Figure 8: The proportions of answer evidence and cross validation of HydraRAG among CWQ, WebQSP, AdvHotpotQA, and QALD10-en datasets.



(a) CWQ (HydraRAG) (b) CWQ (HydraRAG-E)

(c) WebQSP (HydraRAG) (d) WebQSP (HydraRAG-E)

Figure 9: The path overlap ratio of HydraRAG and HydraRAG-E among CWQ, and WebQSP datasets.

$$Ratio(P) = \frac{|Relation(P) \cap Relation(P_G)|}{|Relation(P_G)|},$$

where $Relation(P)$ is the set of relations in path P. Figure 9 shows the distribution of overlap ratios. For WebQSP, HydraRAG achieves the highest proportion of fully overlapping paths (about 61%), while HydraRAG-E shows the most paths with up to 37% non-overlapping relations, indicating that HydraRAG-E explores novel paths to derive the answers. This difference is due to HydraRAG-E's approach of randomly selecting one related edge from each cluster. These results highlight the effectiveness of our structure-based exploration in generating both accurate and diverse reasoning paths.

B.4 Error Analysis

To further examine the integration of LLMs with KGs, we conduct an error analysis on the CWQ, WebQSP, and GrailQA datasets. Errors are categorized into four types: (1) answer generation errors, (2) refusal errors, (3) format errors, and (4) other hallucination errors. An answer generation error is defined as the case where HydraRAG provides a correct reasoning path, but the LLM fails to extract the correct answer from it.

Figure 10 shows the distribution of these error types. The results indicate that more advanced LLMs generally reduce the incidence of "other hallucination errors", "refusal errors", and "answer generation errors", as improved reasoning capabilities allow the model to make better use of the retrieved data. The reduction in "answer generation errors" in particular demonstrates that advanced LLMs can more effectively utilize the reasoning paths generated by HydraRAG. However, we also observe an increase in "format errors" with stronger LLMs, which may be due to their increased creative flexibility in generating outputs.

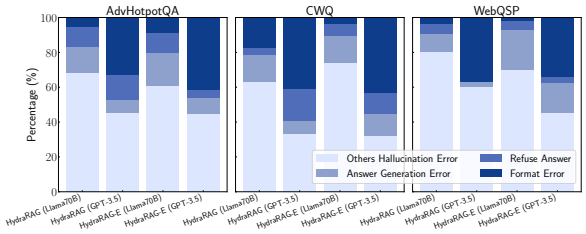


Figure 10: The error instances and categories of HydraRAG and HydraRAG-E in the AdvHotpotQA, CWQ, and WebQSP datasets.

B.5 Efficiency Analysis

LLM calls cost analysis. To evaluate the cost and efficiency of utilizing LLMs, we conducted an analysis of LLM calls on the CWQ, WebQSP, and AdvHotpotQA datasets. Initially, we examined the proportion of questions answered with varying numbers of LLM calls, as depicted in Figure 11. The results indicate that the majority of questions are answered within nine LLM calls across all datasets, with approximately 60% and 70% of questions being resolved within six calls on CWQ and WebQSP, respectively. These findings demonstrate HydraRAG's efficiency in minimizing LLM costs.

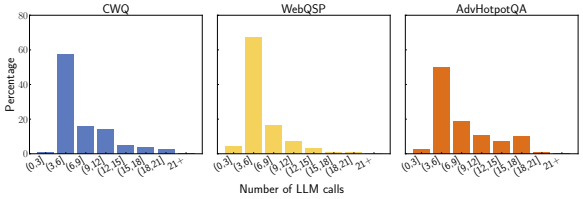


Figure 11: The proportion of questions of HydraRAG and HydraRAG-E by different LLM Calls among CWQ, WebQSP, and AdvHotpotQA datasets.

Table 9: Efficiency analysis of different methods on AdvHotpotQA.

Method	Average Total Time	API Calls	Accuracy
HydraRAG	43.0	8.7	60.7
ToG-2	27.3	5.4	42.9
ToG	69.3	16.3	26.3
CoK	30.1	11.0	45.4

Efficiency analysis on AdvHotpotQA. We compare the efficiency and effectiveness of different multi-hop QA methods on the AdvHotpotQA dataset by reporting average processing time, number of API calls per question, and answer accuracy, as shown in Table 9. Among all methods, HydraRAG achieves the highest accuracy (60.71%) while maintaining a moderate average total processing time (43 seconds) and relatively low API call cost (8.7 per question). Compared to ToG-2 and CoK, which exhibit lower accuracy (42.9% and 45.4%, respectively), HydraRAG offers a clear advantage in answer quality without excessive time or API usage. While ToG-2 achieves the lowest average time and API calls, its accuracy lags significantly behind HydraRAG. Conversely, ToG has the highest processing time and API usage with the lowest accuracy among all compared methods. These results demonstrate that HydraRAG effectively balances efficiency and answer quality, providing a more accurate solution than previous methods while controlling computation and LLM call costs.

C Experiment Details

Experiment datasets. To evaluate the capability of HydraRAG on complex, knowledge-intensive reasoning tasks, we evaluate it on seven KBQA benchmarks. These include four multi-hop datasets: ComplexWebQuestions (CWQ) (Talmor and Berant, 2018), WebQSP (Yih et al., 2016), AdvHotpotQA (Ye and Durrett, 2022), and QALD10-en (Usbeck et al., 2024), a single-hop dataset: Simple Questions (SimpleQA) (Petrochuk and Zettlemoyer, 2018), a slot filling dataset: ZeroShot RE (Petroni et al., 2020), and an open-domain QA dataset: WebQuestions (Berant et al., 2013), to examine HydraRAG on more general tasks. For fair comparison with strong prompt-based baselines, we use the same test splits reported in (Tan et al., 2025; Sun et al., 2024; Ma et al., 2025b).

As background knowledge, we employ the full Freebase (Bollacker et al., 2008), Wikipedia, and Wikidata (Vrandečić and Krötzsch, 2014). Using the complete knowledge setting, rather than a distractor subset, makes retrieval more challenging and better evaluates each method's reasoning ability (Ma et al., 2025b). The statistics of the datasets utilized in this paper are detailed in Table 10. The source code is publicly available [4].

Experiment baselines. We compare HydraRAG to four categories of baselines under an unsupervised setting with GPT-3.5-turbo as the LLM:

- LLM-only methods without external knowledge, include standard prompting (IO), Chain-of-Thought prompting (CoT) (Wei et al., 2022), and Self-Consistency prompting (SC) (Wang et al., 2023) with six in-context examples;
- Vanilla RAG, covers text-based retrieval from entity documents and web-based retrieval from the top three web search results (title and snippets, same as the sample in Figure 1);
- KG-based RAG, includes Think-on-Graph (ToG) (Sun et al., 2024) and Paths-over-Graph (PoG) (Tan et al., 2025);
- Hybrid RAG, consists of Chain-of-Knowledge (CoK) (Li et al., 2024c) and Think-on-Graph-2.0 (ToG-2) (Ma et al., 2025b), which retrieve from both Wikipedia and Wikidata.

For the statistics of existing SOTA, we directly refer to their results and those of other baselines reported in their paper for comparison. Following prior studies (Sun et al., 2024; Ma et al., 2025b; Tan et al., 2025; Chen et al., 2024b; Ma et al., 2025a), we use exact match accuracy (Hits@1) as the evaluation metric. Recall and F1 scores are not used since knowledge sources are not limited to document databases (Sun et al., 2024; Ma et al., 2025b; Tan et al., 2025).

Experiment implementation. All experiments use GPT-3.5-Turbo as the primary LLM. To demonstrate plug-and-play flexibility, we also run HydraRAG with GPT-4-Trubo, Deepseek-v3, Llama-3.1-70B, and Llama-3.1-8B. Following ToG-2 and PoG, we set the temperature to 0.4 during evidence exploration (to increase diversity) and to 0 during path pruning and answer generation (to ensure reproducibility). We use SentenceBERT (Reimers and Gurevych, 2019) as the dense retrieval model (DRM). The maximum generation length is 256 tokens. We fix $W_{\max} = 3$, $D_{\max} = 3$, $W_1 = 100$, and $W_2 = 20$ for evidence pruning. In evidence pruning, we use $\lambda_{\text{sem}} = 0.7$, $\alpha_{\text{cross}} = 0.7$, $(\rho_{\text{KG}}, \rho_{\text{Wiki}}, \rho_{\text{Web}}) = (1.0, 0.8, 0.7)$, and equal weights $\alpha_k = 0.33$ for each feature f_k.

Table 10: Statistics and license information for the datasets used in this paper. * denotes that we utilize the sampled tests reported by existing SOTA work for fairly comparing (Sun et al., 2024; Tan et al., 2025; Ma et al., 2025b).

Dataset	Answer Format	License	Test	Train
ComplexWebQuestions (CWQ)*	Entity	Apache-2.0	1,000	27,734
WebQSP	Entity/Number	MSR-LA	1,639	3,098
AdvHotpotQA	Entity/Number	CC BY-SA 4.0	308	2,312
QALD10-en	Entity/Number	MIT	333	–
Simple Questions*	Entity/Number	CC BY 3.0	1,000	14,894
Zero-Shot RE	Entity/Number	CC BY-SA 4.0	3,724	147,909
WebQuestions	Entity/Number	CC-BY 4.0	2,032	3,778

[4] https://stevetantan.github.io/HydraRAG/

D Case Study: Multi-Source Cross-Verified Interpretable Reasoning

In this section, we present Tables 11-14 to illustrate how HydraRAG combines evidence from the KG, Wikipedia, and the Web for cross-verified reasoning. Through case studies involving questions with multiple entities, and verification across KG-Wiki, KG-Web, and three-source combinations, we show how HydraRAG generates transparent, faithful, and interpretable chains of facts to enhance LLM reasoning. Paths from different sources are color-coded. Showing HydraRAG's effectiveness in multi-entity and multi-hop question answering by providing clear, understandable reasoning paths that support accurate answers.

Table 11: Multi-source interpretable reasoning for "What is the nationality of the wrestler who sang on A Jingle with Jillian?".Paths from KG, Wiki, and Web are color-coded: KG, Wiki, Web.

Field	Content
Question	What is the nationality of the wrestler who sang on A Jingle with Jillian?
Answer	American
Topic Entity	{A Jingle with Jillian}
LLM Indicator	"wrestler" – sang on – "A Jingle with Jillian" – has nationality – answer(nationality)
Split Questions	split_question 1: What wrestler sang on "A Jingle with Jillian"? split_question 2: What is the nationality of the performer of "A Jingle with Jillian"?
Source Evidence	**KG Path:** {A Jingle with Jillian} $\xrightarrow{\text{music.album.artist}}$ {Jillian Hall} $\xrightarrow{\text{people.person.nationality}}$ {**United States of America**} **Web Excerpt:** *"Jillian Hall was born on September 6, 1980, in Ashland, Kentucky, United States. She holds an American nationality."* **Web (converted KG-path):** {A Jingle with Jillian} $\xrightarrow{\text{music.album.artist}}$ {Jillian Hall} $\xrightarrow{\text{birthplace}}$ {Ashland, Kentucky, United States} $\xrightarrow{\text{implies nationality}}$ {**American**}
HydraRAG Answer	**answer:** {**American**} **reason:** Both the knowledge graph and the web source independently support the answer {**American**}. The KG path directly connects "A Jingle with Jillian" to its artist, Jillian Hall, and further to her nationality as "United States of America". The web source confirms this by stating her birthplace is in the United States and she holds American nationality. After converting the web statement to a KG-style path, both sources cross-validate the answer, strengthening its correctness and trustworthiness.

Table 12: Multi-source interpretable reasoning for "What movie was Logan Lerman in that was decorated by Barry Greaves?". Paths from KG, Wiki, and Web are color-coded: KG, Wiki, Web.

Field	Content
Question	What movie was Logan Lerman in that was decorated by Barry Greaves?
Answer	Fury
Topic Entity	{Logan Lerman, Barry Greaves}
LLM Indicator	"Logan Lerman" – acted in – movie – decorated by – "Barry Greaves"
Split Questions	split_question 1: What movie was Logan Lerman in? split_question 2: Which person decorated the movie?
Source Evidence	**KG Path:** {Logan Lerman} $\xrightarrow{\text{film.actor.film}}$ {**Fury**} $\xleftarrow{\text{film.film.film\_set\_decoration\_by}}$ {Barry Greaves} **Wiki Excerpt:** *"Lerman next starred in David Ayer's World War II-set film, Fury; in the film, Lerman's character is an American soldier fighting Nazi forces."* **Wiki (converted KG-path):** {Logan Lerman} $\xrightarrow{\text{starred in}}$ {**Fury**} *(The Wikipedia article confirms Logan Lerman's participation in **Fury**. The decoration by Barry Greaves is verified by the KG path.)*
HydraRAG Answer	**answer:** {**Fury**} **reason:** Both the knowledge graph and Wikipedia independently identify **{Fury}** as the movie that Logan Lerman acted in. The KG path links Logan Lerman and Barry Greaves directly to the film, while Wikipedia supports Logan Lerman's involvement in the same work. Together these sources provide cross-validated evidence for the answer.

Table 13: Multi-source interpretable reasoning for "What member of the Republican Party fought in the Battle of Vicksburg?". Paths from KG, Wiki, and Web are color-coded: KG, Wiki, Web.

Field	Content
Question	What member of the Republican Party fought in the Battle of Vicksburg?
Answer	Ulysses S. Grant
Topic Entities	{Siege of Vicksburg, Republican Party}
LLM Indicator	"Siege of Vicksburg" – also known as – "Battle of Vicksburg" – fought by – answer(member) – member of – "Republican Party"
Split Questions	split_question 1: What battle is also known as the "Siege of Vicksburg"? split_question 2: What member of the "Republican Party" fought in the "Battle of Vicksburg"?
Source Evidence	**KG Path:** {Siege of Vicksburg} $\xrightarrow{\text{also known as}}$ {Battle of Vicksburg} $\xrightarrow{\text{fought by}}$ {Ulysses S. Grant} $\xrightarrow{\text{member of}}$ {Republican Party} **Wiki Excerpt:** *"The Siege of Vicksburg (May 18 – July 4, 1863) was the final major military action in the Vicksburg campaign of the American Civil War. In a series of maneuvers, Union Major General Ulysses S. Grant and his Army..."* **Wiki (converted KG-path):** {Siege of Vicksburg} $\xrightarrow{\text{also known as}}$ {Battle of Vicksburg} $\xrightarrow{\text{fought by}}$ {Ulysses S. Grant} **Web Excerpt:** *"The Battle of Vicksburg, or Siege of Vicksburg, was the final significant battle... Union Maj. Gen. Ulysses S. Grant and his Army of the Tennessee crossed the Mississippi River..."* **Web (converted KG-path):** {Battle of Vicksburg} $\xrightarrow{\text{fought by}}$ {Ulysses S. Grant}
HydraRAG Answer	**answer:** {**Ulysses S. Grant**} **reason:** All three sources—KG, Wikipedia, and Web—support that Ulysses S. Grant fought in the Battle (Siege) of Vicksburg. The KG path further confirms his Republican Party membership. Wiki and Web sources confirm his role as a military leader in the battle, and after conversion to KG-path style, all sources consistently point to **{Ulysses S. Grant}** as the answer, demonstrating robust multi-source verification.

Table 14: Multi-source interpretable reasoning for "What team that has a mascot named Mariner Moose is in the American League West?". Paths from KG, Wiki, and Web are color-coded: KG, Wiki, Web.

Field	Content
Question	What team that has a mascot named Mariner Moose is in the American League West?
Answer	Seattle Mariners
Topic Entities	{Mariner Moose, American League West}
LLM Indicator	"Mariner Moose" – mascot of – team – division – answer(team) – located in – "American League West"
Split Questions	split_question 1: Which team has a mascot named "Mariner Moose"? split_question 2: Which team is in the "American League West" division?
Source Evidence	**KG Path:** {Mariner Moose} $\xrightarrow{\text{sports.mascot.team}}$ {Seattle Mariners} $\xrightarrow{\text{baseball.baseball\_team.division}}$ {American League West} **Wiki Excerpt:** *"The Mariner Moose is the team mascot of the Seattle Mariners, a Major League Baseball team... The Seattle Mariners are an American professional baseball team based in Seattle. The Mariners compete in Major League Baseball (MLB) as a member club of the American League (AL) West Division."* **Wiki (converted KG-path):** {Mariner Moose} $\xrightarrow{\text{mascot of}}$ {Seattle Mariners} $\xrightarrow{\text{member of}}$ {American League West} **Web Excerpt:** *"Their mascot is the Mariner Moose. The Seattle Mariners are an American professional baseball team based in Seattle. The Mariners compete in Major League Baseball (MLB) as a member club of the American League (AL) West Division."* **Web (converted KG-path):** {Mariner Moose} $\xrightarrow{\text{team mascot of}}$ {Seattle Mariners} $\xrightarrow{\text{compete in}}$ {American League West}
HydraRAG Answer	**answer:** {Seattle Mariners} **reason:** All three sources—KG, Wikipedia, and Web—consistently support that the **{Seattle Mariners}** have Mariner Moose as their mascot and are a team in the American League West division. The KG path provides a direct multi-hop link; the Wiki and Web evidence, after conversion to KG-path style, corroborate both the team and its division membership. This provides strong cross-source verification of the answer.

E Prompts

In this section, we detail the prompts required for our main experimental procedures.

Question Analysis Prompt Template

You will receive a multi-hop question, which is composed of several interconnected queries, along with a list of topic entities that serve as the main keywords for the question. Your task is to break the question into simpler parts, using each topic entity once and provide a Chain of Thought (CoT) that shows how the topic entities are related. Note: Each simpler question should explore how one topic entity connects to others or the answer. The goal is to systematically address each entity to derive the final answer.

`In-Context Few-shot`

Q: {Query}
Topic Entity: {Topic Entity}
A:

Agentic Source Selector Prompt Template

You are a source selection agent. Your task is to decide the most appropriate knowledge source(s) to answer a user's question. You will be provided with up to three sources: Local Knowledge Graph (KG) Wiki Documents (Wiki) Web Search (Web). Follow these steps carefully:
I. Analyze the question thoroughly.
II. Prioritize KG if available: If KG alone is sufficient, select KG. If KG is incomplete, check if Wiki can fill the missing information. If so, combine KG with Wiki. If neither KG nor Wiki suffices, include Web search.
III. If KG is unavailable: Choose between Wiki and Web based on recency and the likely completeness of the Wiki documents.
Clearly state your reasoning, and then indicate your decision using these actions:
action1 for KG
action2 for Wiki
action3 for Web
Noted, combinations allowed (e.g., [action1 + action2]).

`In-Context Few-shot`

Q: {Query}
Provided sources: {Provided sources}
Question analysis: {Question analysis}
A:

From Paragraph to Knowledge Path Prompt Template

You will receive a multi-hop question, which consists of several interrelated queries, a list of subject entities as the main keywords of the question, three related questions and answers returned by Google search, and three online related search results from Google search. Your task is to summarize these search results, find sentences that may be related to the answer, and organise them into a knowledge graph path for each paragraph. Note that at least one path for each paragraph should contain the main topic entities. Please answer the question directly in the format below: [Brad Paisley - enrolled at - West Liberty State College - transferred to - Belmont University - earned - Bachelor's degree]

`In-Context Few-shot`

Q: {Query}
Topic Entity: {Topic Entity}
Paragraph 1:{Paragraph 1}
Paragraph 2:{Paragraph 2}
Paragraph 3:{Paragraph 3}
A:

where {Skyline Indicator}, and {Split Question} are obtained in Section 4.1. {Existing Knowledge Paths} and {Candidate Paths} denote the retrieved reasoning paths, which are formatted as a series of structural sentences, where, i and j in r_{1_i}, r_{1_i} represent the i-th, j-th relation from each relation edge in the clustered question subgraph.

$$\{e_{0x}, ..., e_{0z}\} \rightarrow r_{1_i} \rightarrow \{e_{1x}, ..., e_{1z}\} \rightarrow \ldots \rightarrow r_{l_j} \rightarrow \{e_{lx}, ..., e_{lz}\}$$
$$\ldots$$
$$\{e_{0x}, ..., e_{0z}\} \rightarrow r_{1_i} \rightarrow \{e_{1x}, ..., e_{1z}\} \rightarrow \ldots \rightarrow r_{l_j} \rightarrow \{e_{lx}, ..., e_{lz}\},$$

Refined Exploration Prompt Template

Given a main question, an uncertain LLM-generated thinking Cot that considers all the entities, a few split questions that you can use and finally obtain the final answer, the associated accuracy retrieved knowledge paths from the Related_path section, and the main topic entities Please predict the additional evidence that needs to be found to answer the current question, then provide a suitable query for retrieving this potential evidence. and give the possible Chains of Thought that can lead to the predicted result in the same format below, by the given knowledge path and your own knowledge.

`In-Context Few-shot`

Q: {Query}
Topic Entity: {Topic Entity}
Skyline Indicator:{Skyline Indicator}
Split Question:{Split Question}
Existing Knowledge Paths:{Existing Knowledge Paths}
A:

Predict Exploration Prompt Template

Using the main question, a possibly uncertain chain of thought generated by a language model, some related split questions, paths from the "Related_paths" section, and main topic entities: please first provide three predicted results, and second offer three possible chains of thought that could lead to these results, using the provided knowledge paths and your own knowledge. If any answers are unclear, suggest alternative answers to fill in the gaps in the chains of thought, following the same format as the provided examples.

`In-Context Few-shot`

Q: {Query}
Topic Entity: {Topic Entity}
Skyline Indicator:{Skyline Indicator}
Split Question:{Split Question}
Existing Knowledge Paths:{Existing Knowledge Paths}
A:

LLM-aware Paths Select Prompt Template

Given a main question, a LLM-generated thinking Cot that considers all the entities, a few split questions that you can use one by one and finally obtain the final answer, and the associated retrieved knowledge graph path, {set of entities (with id start with "m.")} -> {set of relationships} -> {set of entities(with id start with "m.")}, Please score and give me the top three lists from the candidate paths set that are highly likely to be the answer to the question.

`In-Context Few-shot`

Q: {Query}
Skyline Indicator:{Skyline Indicator}
Split Question:{Split Question}
Candidate Paths:{Candidate Paths}
A:

Path Refinement Prompt Template

Given a main question, an uncertain LLM-generated thinking Cot that considers all the entities, a few split questions that you can use one by one and finally obtain the final answer, the associated accuracy retrieved knowledge paths from the Related paths section, and main topic entities. Your task is to summarize the provided knowledge triple in the Related paths section and generate a chain of thoughts by the knowledge triple related to the main topic entities of the question, which will be used for generating the answer for the main question and splitting the question further. You have to make sure you summarize correctly by using the provided knowledge triple, you can only use the entity with the id from the given path, and you can not skip steps.

`In-Context Few-shot`

Q: {Query}
Skyline Indicator:{Skyline Indicator}
Split Question:{Split Question}
Related Paths:{Related Paths}
A:

CoT Answering Evaluation Prompt Template

Given a main question, an uncertain LLM-generated thinking Cot that considers all the entities, a few split questions that you can use and finally obtain the final answer, and the associated retrieved knowledge graph path, {set of entities (with id start with "m.")} -> {set of relationships} -> {set of entities(with id start with "m.")}. Your task is to determine if this knowledge graph path is sufficient to answer the given split question first then the main question. If it's sufficient, you need to respond {Yes} and provide the answer to the main question. If the answer is obtained from the given knowledge path, it should be the entity name from the path. Otherwise, you need to respond {No}, then explain the reason.

`In-Context Few-shot`

Q: {Query}
Skyline Indicator:{Skyline Indicator}
Split Question:{Split Question}
Existing Knowledge Paths:{Existing Knowledge Paths}
A:

CoT Answering Generation Prompt Template

Given a main question, an uncertain LLM-generated thinking Cot that considers all the entities, a few split questions that you can use one by one and finally obtain the final answer, and the associated retrieved knowledge graph path, {set of entities (with id start with "m.")} -> {set of relationships} -> {set of entities(with id start with "m.")}, Your task is to generate the answer based on the given knowledge graph path and your own knowledge.

`In-Context Few-shot`

Q: {Query}
Skyline Indicator:{Skyline Indicator}
Split Question:{Split Question}
Related Paths:{Related Paths}
A: