Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

Integrating Knowledge Graphs with Adaptive Schema into Wikipedia-based Question Answering Systems

Cong Tran
Posts and Telecommunications Institute of Technology
Linh Nguyen-Thi-Dieu
Hanoi University of Science and Technology
Hai Van Pham
haipv@soict.hust.edu.vn

Hanoi University of Science and Technology

Research Article

Keywords: GraphRAG, retrieval-augmented generation, multi-hop question answering, knowledge graphs, adaptive schema discovery, post-hoc triple injection, cross-encoder reranking.

Posted Date: November 11th, 2026

DOI: https://doi.org/10.21203/rs.3.rs-10822922/v1

License: This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

Additional Declarations: No competing interests reported.

Integrating Knowledge Graphs with Adaptive Schema into Wikipedia-based Question Answering Systems

Cong Tran
Faculty of Artificial Intelligence,
Posts and Telecommunications Institute of Technology, Hanoi, Vietnam

Linh Nguyen-Thi-Dieu
School of Information and Communications Technology,
Hanoi University of Science and Technology, Hanoi, Vietnam

Hai Van Pham
School of Information and Communications Technology,
Hanoi University of Science and Technology, Hanoi, Vietnam

September 2, 2026

Abstract – Knowledge-graph-based retrieval can improve multi-hop question answering by linking evidence that is weakly connected to the query in embedding space. This paper presents an adaptive-schema, graph-enhanced retrieval framework for Wikipedia-based question answering that replaces open-ended LLM-based graph construction with local GLiNER2-based schema-guided extraction. The framework discovers missing corpus-specific relations from uncovered entity pairs and injects the resulting triples post hoc, avoiding repeated corpus-wide extraction. Candidate evidence is expanded with Personalized PageRank, reranked by a local cross-encoder, and selected using Graph-Path, a bounded topology-aware passage selection method. On 1,000 questions from MuSiQue, 2WikiMultihopQA, and HotpotQA, the framework achieves F1 scores of 0.4577, 0.6637, and 0.7154, improving over a BGE-based HippoRAG2 baseline. On MuSiQue, adaptive expansion adds 93 relation types and 1,691 triples, raising F1 by 0.0225 over the seed-only graph and achieving Recall@10 of 0.6423 versus 0.5955 for RAPTOR. These results show that controlled schema expansion and topology-aware evidence selection improve evidence coverage while reducing dependence on LLM-based graph construction; only final answer generation remains API-based.

Keywords: GraphRAG, retrieval-augmented generation, multi-hop question answering, knowledge graphs, adaptive schema discovery, post-hoc triple injection, cross-encoder reranking.

1 Introduction

Multi-hop question answering requires retrieving and combining evidence from multiple documents

connected through intermediate entities and relations. Conventional dense retrieval methods rank passages according to semantic similarity between query and document embeddings, but often struggle when relevant evidence is separated from the query by several reasoning steps. In such cases, intermediate passages may be essential for reaching the final answer even though they share little direct semantic similarity with the original query [1, 30].

Graph-based retrieval has emerged as an effective approach for addressing this limitation. By explicitly modeling entities and semantic relations as a graph, retrieval systems can propagate relevance through connected evidence chains rather than relying solely on query-document similarity [5, 11]. Frameworks such as GraphRAG and HippoRAG have demonstrated that graph-enhanced retrieval can substantially improve evidence discovery for complex reasoning tasks.

Despite these advances, the effectiveness of graph-based retrieval remains highly dependent on the completeness of the underlying knowledge graph. Recent systems such as HippoRAG construct graph structures through LLM-based open information extraction, enabling diverse semantic relations to be captured without predefined constraints [11]. While this approach provides flexible graph construction, it also introduces substantial inference costs and dependence on commercial API services during corpus-scale graph generation.

A natural alternative is to replace LLM-based extraction with a lightweight local extractor operating under a predefined relation schema. Such schema-guided extraction improves reproducibility, deployment practicality, and computational efficiency. However, it introduces a coverage limitation: relations expressed in the corpus but absent from the predefined schema cannot be extracted even when they occur repeatedly in source documents [2]. As a result, semantically related entities may remain disconnected in the graph, producing fragmented graph topology and incomplete reasoning paths.

This limitation is particularly problematic in multi-hop question answering, where retrieval depends on traversing chains of interconnected evidence. Missing relations can interrupt these chains, preventing graph-based retrieval methods from reaching relevant supporting documents despite their presence in the corpus. The problem addressed in this work is therefore how to construct a knowledge graph using a local schema-guided extraction pipeline while automatically recovering corpus-specific relations omitted by the initial schema, without requiring repeated corpus-scale extraction or reverting to LLM-based graph construction.

To address this problem, this work proposes a graph-enhanced retrieval-augmented generation framework for Wikipedia-based multi-hop question answering. The proposed framework combines local schema-guided graph construction and adaptive schema expansion on top of the graph retrieval architecture of HippoRAG. Starting from a predefined relation schema, the framework identifies extraction gaps, discovers additional relation types from corpus evidence, and integrates newly discovered relations into the graph through a post-hoc augmentation process without requiring a second corpus-wide extraction pass. In addition, a graph-guided context analysis module is introduced to examine the impact of graph connectivity on downstream retrieval and question answering performance.

The main contributions of this work are as follows:

- A local schema-guided knowledge graph construction framework is developed to replace LLM-based corpus-scale extraction, eliminating dependence on external API services while maintaining compatibility with graph-based retrieval.

- Adaptive Schema Discovery with Post-hoc Triple Injection is introduced to identify extraction gaps from uncovered entity pairs, discover corpus-specific relation types through contextual clustering, and integrate the resulting relations into the graph without rerunning corpus-level extraction.
- Graph-Path is introduced as a topology-aware context analysis mechanism for evaluating graph connectivity and investigating the contribution of structural reachability to multi-hop retrieval performance.

Experiments on MuSiQue, 2WikiMultihopQA, and HotpotQA evaluate the effectiveness of the proposed framework and analyze the impact of adaptive schema expansion and graph connectivity on retrieval and question answering performance.

The remainder of the paper is organized as follows. Section 2 reviews related work on retrieval-augmented generation, graph-based retrieval, and adaptive schema construction. Section 3 presents the proposed framework. Section 4 describes the experimental setup and results. Finally, Section 6 concludes the paper and discusses future directions.

2 Theoretical Foundations and Related Work

Recent advances in retrieval-augmented generation (RAG) demonstrate that external retrieval can substantially improve the factual grounding of large language models in knowledge-intensive tasks. As question answering benchmarks increasingly emphasize multi-hop reasoning, researchers have explored graph-based retrieval architectures that explicitly model entities and semantic relations to support evidence aggregation across multiple documents. Meanwhile, recent studies on adaptive schema learning and automatic knowledge graph construction seek to improve the flexibility and scalability of graph-based retrieval systems. This section reviews the theoretical foundations and representative studies related to retrieval grounding, graph-based reasoning, graph construction, and adaptive schema learning that motivate the proposed framework.

2.1 Challenges in Retrieval-augmented Generation

Retrieval-augmented generation was introduced to improve factual grounding by integrating external retrieval systems into the generation process of large language models [8, 18]. Recent variants further study corrective retrieval and adaptive retrieval control for handling noisy evidence and varying question complexity [15, 34]. Rather than relying solely on parametric memory, RAG systems retrieve relevant documents from external corpora and incorporate them into the model context before answer generation. This retrieval-grounded generation process has been shown to reduce hallucination and improve factual consistency in knowledge-intensive tasks.

Most existing RAG systems rely heavily on dense retrieval architectures such as Dense Passage Retrieval (DPR), Contriever, and Sentence-BERT embeddings [14, 17, 27]. These approaches project both queries and candidate documents into a shared semantic embedding space, where retrieval relevance is estimated through vector similarity. Documents with embeddings more closely aligned with the query representation are assigned higher retrieval scores and are subsequently incorporated into the generation context.

Although dense retrieval methods perform effectively for semantic matching, embedding similarity alone is often insufficient for multi-hop reasoning problems [1, 21, 30]. In many question answering

scenarios, supporting evidence is distributed across multiple documents connected through implicit semantic relationships rather than direct semantic proximity. Consequently, intermediate bridge documents may receive relatively low retrieval scores despite playing an essential role in the reasoning chain.

Another challenge arises in long-context utilization in transformer-based language models. Liu et al. demonstrated that LLMs often underutilize information located in the middle region of long contexts, commonly referred to as the "Lost in the Middle" phenomenon [19]. This limitation highlights the importance of effective retrieval ordering, context compression, and reasoning-aware context selection strategies.

2.2 Graph-Based Retrieval and Multi-hop Reasoning

To address the limitations of purely vector-based retrieval, recent research has increasingly explored graph-based retrieval architectures that explicitly model semantic relationships between entities and documents. Knowledge graphs (KGs) provide structured semantic representations in the form of interconnected entities and relations [5, 6, 26].

Formally, a knowledge graph can be represented as a graph $G = (V, E)$, where V denotes a set of entities and E denotes semantic relationships between entities. Unlike dense retrieval systems that rely primarily on embedding similarity, graph-based retrieval methods utilize graph topology and relational connectivity to traverse semantically related evidence chains [20].

Recent GraphRAG architectures have demonstrated that graph-aware retrieval can substantially improve reasoning quality in complex question answering tasks. Microsoft's GraphRAG framework organizes documents into semantic communities and performs graph-guided summarization, while HippoRAG combines knowledge graph traversal with Personalized PageRank propagation for multi-hop reasoning [5, 11]. Recent surveys and analyses further examine when graph-based RAG is useful and how graph structure affects retrieval behavior [32, 38, 40]. Other recent systems explore graph-enhanced agents, hypergraph retrieval, and adaptive reasoning structures for reasoning-intensive QA tasks [3, 29, 31]. Direction-aware traversal has also been studied as a way to guide multi-hop retrieval over graph-structured evidence [39].

Lightweight retrieval frameworks such as LightRAG and MiniRAG investigate simplified graph retrieval pipelines for resource-constrained deployment environments [7, 10]. Collectively, these studies indicate that graph structures provide useful mechanisms for modeling semantic reasoning paths beyond vector similarity alone. Related graph-oriented RAG work also explores retrieval over textual graphs for question answering, further supporting the use of graph structure as a retrieval substrate [12].

2.3 Reasoning-Aware Graph Traversal

Graph-based retrieval systems commonly use graph traversal and ranking algorithms to propagate query relevance across semantically connected entities. Among these approaches, Personalized PageRank (PPR) has become one of the most widely adopted mechanisms for reasoning-aware retrieval in graph-based question answering systems [11, 24].

Unlike standard retrieval methods that rank passages independently, graph traversal methods estimate relevance through iterative propagation over graph structure. Classical PageRank models node importance through recursive propagation, allowing structurally influential nodes to accumulate higher relevance scores over connected neighborhoods [24]. However, because this process remains

query-independent, it cannot directly capture task-specific information needs.

Personalized PageRank (PPR) extends classical PageRank by incorporating query-specific information into graph traversal. Let $\mathbf{r}_t$ denote the relevance score vector at iteration t, $\mathbf{A}$ denote the normalized adjacency matrix, and $\mathbf{s}$ represent the query-specific seed distribution. The propagation process is defined as

$$\mathbf{r}_{t+1} = (1-\alpha)\mathbf{A}^T\mathbf{r}_t + \alpha\mathbf{s},$$

where α denotes the restart probability. The propagation term allows relevance scores to spread through connected entities and passages, while the restart term preserves alignment with the original query. Consequently, nodes that are both structurally connected and query-relevant receive higher scores, making PPR effective for multi-hop retrieval tasks.

Beyond graph traversal itself, recent studies have further explored reasoning-aware reranking and context selection strategies to improve retrieval consistency in long-context generation settings [8, 19]. These approaches emphasize that effective retrieval depends not only on identifying relevant evidence but also on organizing retrieved information in ways that preserve semantic coherence during downstream generation.

2.4 Adaptive Schema Construction and Local Information Extraction

Constructing a knowledge graph requires reliable extraction pipelines capable of identifying entities and semantic relations from unstructured text. Traditional extraction systems often rely on manually designed schemas or supervised sequence labeling approaches that require domain-specific adaptation and schema engineering [16].

Recent studies have increasingly investigated the use of large language models for open information extraction, schema induction, and relation extraction tasks. AutoSchemaKG demonstrates that knowledge graphs can be constructed through autonomous schema induction without predefined ontologies, enabling semantic structures to evolve dynamically across heterogeneous domains [2,9]. Such adaptive schema mechanisms improve flexibility and scalability in graph construction pipelines.

Despite their effectiveness, many LLM-based extraction systems still depend heavily on large proprietary models and cloud-based APIs, introducing computational overhead, operational costs, and unstable structured outputs. To improve deployment practicality, lightweight extraction frameworks such as GLiNER introduce flexible label-aware extraction mechanisms that can operate efficiently on consumer hardware [37].

In this work, we combine local extraction mechanisms with adaptive schema evolution strategies to support dynamic semantic relation expansion during knowledge graph construction. This design enables the retrieval system to maintain graph flexibility while reducing dependence on external extraction services.

2.5 Discussion and Research Gap

Existing graph-based retrieval systems demonstrate that graph structures can improve multi-hop reasoning by connecting semantically related evidence across multiple documents. However, retrieval effectiveness remains strongly dependent on the quality of the underlying knowledge graph.

Current graph construction approaches typically face a trade-off between relation consistency and relation coverage. Open-schema extraction methods provide broad semantic coverage but often generate fragmented relation spaces, whereas schema-guided approaches maintain more structured relation inven-

tories at the cost of missing corpus-specific relations. Consequently, important semantic connections may remain absent from the graph despite being present in the source documents.

Motivated by this limitation, this study investigates whether adaptive schema expansion can improve graph coverage while preserving a coherent relational structure. To address this challenge, a graph-enhanced question answering framework combining schema-guided extraction, adaptive schema discovery, and graph-based retrieval is proposed.

3 Proposed Framework

This section presents the proposed graph-enhanced retrieval framework for Wikipedia-based multi-hop question answering. As shown in Figure 1, the framework integrates local knowledge graph construction, graph-based retrieval and reasoning, and LLM-based answer generation within a unified retrieval-augmented generation (RAG) pipeline [8, 18]. The overall design aims to improve retrieval quality for reasoning-intensive tasks while reducing dependence on external extraction and reranking services.

The overall architecture of the proposed local GraphRAG pipeline is organized into three major phases: (i) offline knowledge graph construction, (ii) online graph-based retrieval and reasoning, and (iii) grounded answer generation. Phase (i) takes the document corpus D and seed schema R_0 as input and outputs the final graph $G = (\mathcal{V}, \mathcal{A})$. Phase (ii) takes a query Q and G as input and outputs a ranked candidate passage set $\mathcal{P}$. Phase (iii) takes Q, the selected passages, and their supporting triples as input and outputs the final answer.

In the first phase, the document corpus is transformed into a local typed knowledge graph through schema-guided entity and relation extraction using GLiNER2 [36]. Beyond standard extraction, this phase incorporates deterministic graph filtering, adaptive schema discovery, and post-hoc triple injection. These components allow the graph schema to evolve dynamically and enable newly discovered relations to be integrated without rerunning the entire extraction pipeline.

The second phase performs retrieval and reasoning over the constructed graph. Given a user query, query mentions are extracted using the same entity recognizer as graph construction and linked to graph nodes by normalized surface-form matching; unmatched mentions are ignored during graph traversal. The system first uses Personalized PageRank (PPR) to expand query-entity neighborhoods, then reranks the resulting facts with a local cross-encoder, and finally applies Graph-Path BFS routing to select compact, structurally connected passages for multi-hop reasoning [24, 26].

Finally, the answer generation phase converts the selected passages and supporting triples into a grounded reasoning context. Instead of relying on the raw retrieval pool, the LLM operates over this graph-enhanced context, enabling the final answer to be conditioned on both textual evidence and structured entity-relation dependencies. The selector returns the highest-ranked passages until the fixed context budget is reached; supporting triples are those extracted from the selected passages or incident to entities in them. The following subsections first provide a high-level explanation of each phase, then detail the procedures for knowledge graph construction, adaptive schema discovery, post-hoc triple injection, reranking, and reasoning-aware context selection.

3.1 Knowledge Graph Construction

The first stage transforms the unstructured document corpus D into a structured knowledge graph

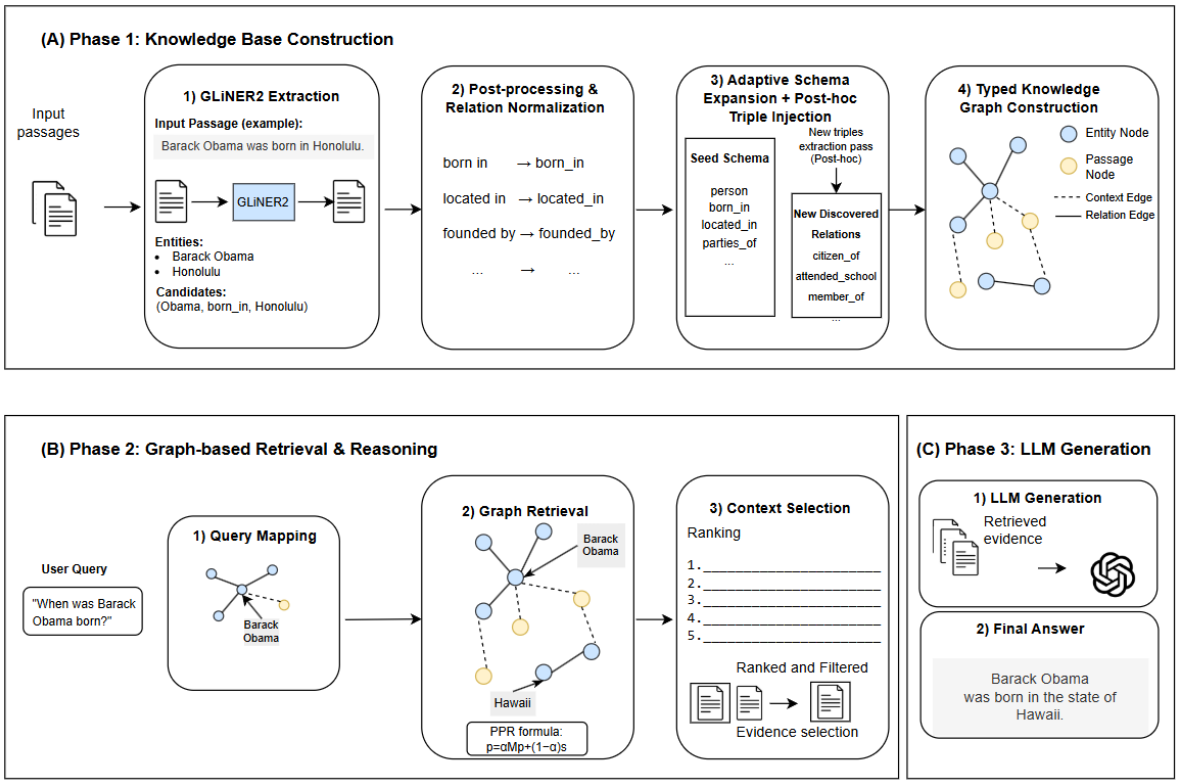


Figure 1: Overall methodology of the proposed graph-enhanced QA framework. The pipeline separates offline knowledge graph construction from online graph-based retrieval, context selection, and final answer generation.

$G = (\mathcal{V}, \mathcal{A})$, where $\mathcal{V}$ denotes the set of canonical entity nodes, $\mathcal{A}$ is the set of directed relation edges represented as triples (u, r, v) with $u, v \in \mathcal{V}$ and $r \in R$, and R denotes the current relation inventory, through schema-guided local entity and relation extraction. Unlike generic OpenIE pipelines that rely on unconstrained generation, our approach adopts a predefined semantic schema to guide extraction and improve graph consistency during corpus-level knowledge graph construction [6, 16].

In this work, GLiNER2 [36] is adopted as the primary extraction framework due to its lightweight architecture, schema-guided extraction mechanism, and suitability for efficient large-scale document processing. GLiNER2 jointly performs entity and relation extraction under explicit semantic constraints, allowing the generated graph structure to remain aligned with predefined entity and relation semantics while avoiding the instability commonly associated with unrestricted extraction pipelines.

3.1.1 Initial Schema Design and Graph Regularization

To improve extraction consistency and reduce noisy graph construction, the pipeline is initialized with a regularized seed schema consisting of predefined entity categories and semantic relation types. The seed schema contains 15 entity categories and more than 50 semantic relation types designed to capture common factual structures encountered in multi-hop reasoning corpora such as MuSiQue [30]. Representative entity categories include *person*, *organization*, *location*, and *event*, while relation types cover person–organization, organization–location, temporal, and event-oriented semantic interactions.

Preliminary inspection of extraction outputs revealed that several semantically ambiguous family-related relations produced disproportionately noisy graph structures. Because GLiNER2 performs extraction independently at the local sentence level, highly ambiguous relations may be over-generalized in linguistically complex contexts, resulting in semantically unreliable connections. In particular, relations such as *parent_of*, *sibling_of*, and *spouse_of* frequently introduce incorrect or weakly supported links between co-occurring entities.

To reduce this source of structural noise, these ambiguous family relations are removed during seed schema design and replaced by more constrained directional relations when appropriate. Rather than relying on post-hoc graph correction, this regularization is applied prior to extraction, allowing semantic constraints to be enforced directly during graph construction. The resulting schema produces cleaner graph connectivity and reduces noisy edge formation that may negatively affect downstream graph traversal and reasoning.

After extraction, the generated triples are further processed using a deterministic filtering pipeline to improve graph quality and suppress extraction artifacts. The filtering stage removes duplicate triples, self-loop relations, purely numerical entities, extremely short entity mentions, and unresolved pronominal references. Entity and relation representations are additionally normalized to reduce lexical noise and maintain structural consistency across the corpus. Such deterministic post-processing steps are widely adopted in information extraction systems to improve graph coherence and reliability [16].

3.1.2 Adaptive Schema Discovery

Although schema-guided extraction improves graph consistency, fixed relation ontologies may limit coverage when previously unseen semantic relations appear in the corpus. To address this limitation, the proposed framework introduces an adaptive schema discovery mechanism that expands relation schemas using recurring corpus-level evidence while preserving the stability benefits of schema-constrained extraction [2, 25].

Rather than using unconstrained relation generation, our approach analyzes extraction gaps already present in the graph to perform controlled schema expansion. As illustrated in Figure 2, the process begins with the initial triple set T_0 and identifies uncovered entity pairs—co-occurring entities that do not yet share an extracted relation edge. Specifically, let $\mathcal{V}_d \subseteq \mathcal{V}$ be the set of canonical entities extracted from document d. An unordered pair $\{u, v\}$ with $u, v \in \mathcal{V}_d$ is called *uncovered* if there is no relation $r \in R_0$ such that either $(u, r, v) \in T_0$ or $(v, r, u) \in T_0$. These uncovered pairs are treated as potential indicators of latent semantic relations.

The intuition is that semantically related entities often appear within informative local textual contexts even when the predefined schema fails to capture their relationship. Adaptive schema discovery therefore examines contextual evidence surrounding uncovered entity pairs instead of depending solely

Algorithm 1 Adaptive schema discovery

Input: D, T_0, R_0, $\{\mathcal{V}_d\}_{d\in D}$, embedding function ϕ, θ_{sup}, θ_{cluster}
Output: R_{new}, $\mathcal{M}$

$\mathcal{C} \leftarrow \emptyset$
for $d \in D$ **do**
  for $\{u,v\} \in \text{uncoveredPairs}(\mathcal{V}_d, T_0, R_0)$ **do**
    $C_{uv}^{(d)} \leftarrow \text{shortestSpan}(d,u,v)$
    $\mathcal{C} \leftarrow \mathcal{C} \cup \{(C_{uv}^{(d)}, u, v, d)\}$
  end for
end for
$\mathcal{C} \leftarrow \text{filterLowSupport}(\mathcal{C}, \theta_{\text{sup}})$
if $\mathcal{C} = \emptyset$ **then**
  return $\emptyset, \emptyset$
end if
$\mathbf{c}_{uv}^{(d)} \leftarrow \phi(C_{uv}^{(d)})$ for each entry in $\mathcal{C}$
$\mathcal{K} \leftarrow \text{AHC}(\{\mathbf{c}_{uv}^{(d)}\})$
$R_{new} \leftarrow \emptyset$; $\mathcal{M} \leftarrow \emptyset$
for $k \in \mathcal{K}$ **do**
  if $|k| \geq \theta_{\text{cluster}}$ **then**
    $r \leftarrow \text{canonicalName}(k)$
    if $r \notin R_0 \cup R_{new}$ **then**
      $R_{new} \leftarrow R_{new} \cup \{r\}$; $\mathcal{M}(k) \leftarrow r$
    end if
  end if
end for
return $R_{new}, \mathcal{M}$

on predefined relation labels. We denote by $\mathcal{U}_d = \{\{u,v\} : u,v \in \mathcal{V}_d, u \neq v, \nexists r \in R_0$ such that $(u,r,v) \in T_0$ or $(v,r,u) \in T_0\}$ the set of uncovered pairs in document d, by $C_{uv}^{(d)}$ the shortest token span in d that contains mentions of both entities u and v, by R_{new} the set of newly discovered relations, and by $\mathcal{M} : \mathcal{K} \to R_{new}$ the mapping from relation clusters to canonical relation labels. Algorithm 1 formalizes the discovery procedure. Its inputs are the corpus D, the initial triple set T_0, the existing relation inventory R_0, the extracted entity sets $\{\mathcal{V}_d\}_{d\in D}$, the support threshold θ_{sup}, and the cluster-size threshold θ_{cluster}; its outputs are R_{new} and $\mathcal{M}$.

The algorithm first scans all documents, extracts the shortest textual span between each uncovered entity pair, and stores it in a multiset cache $\mathcal{C}$ so repeated phrases contribute to support counts. Pairs without a co-occurrence span are skipped. It then discards normalized context phrases whose corpus frequency is below θ_{sup} to avoid spurious relations. Only contextual phrases are embedded; entity identifiers and document IDs are retained for later injection but are not part of the embedding input. The remaining phrases are embedded using Sentence-BERT [27] and grouped via agglomerative hierarchical clustering (AHC) using cosine distance and average linkage [22], which avoids requiring a prespecified number of clusters. Each cluster k that meets the minimum size threshold θ_{cluster} is mapped to a canonical relation name r by selecting the most frequent normalized phrase in the cluster and converting it to a relation label (e.g., "born in" $\to$ *born_in*); labels already present in R_0 are not added again. The resulting set R_{new} extends the original ontology in a data-driven and reproducible manner.

3.1.3 Post-hoc Triple Injection

Conventional extraction pipelines typically require rerunning relation extraction whenever new relation types are introduced, which creates substantial computational overhead during iterative schema refinement [2, 25]. This overhead becomes particularly restrictive in adaptive knowledge graph construction, where relation schemas may evolve over multiple discovery cycles.

To reduce this overhead, the proposed framework adopts a post-hoc triple injection mechanism that separates schema expansion from repeated extraction. Rather than rerunning GLiNER2 after adaptive schema discovery, the framework reuses previously extracted entities and contextual evidence to instantiate new relation edges under the expanded

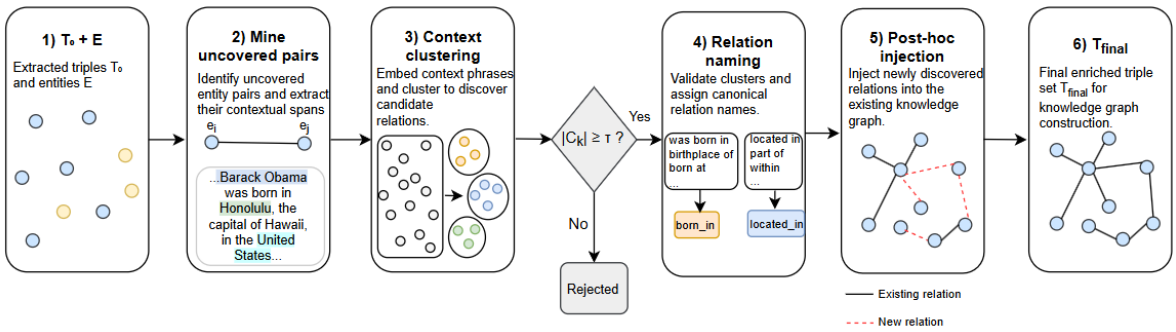


Figure 2: Adaptive schema discovery. The process identifies uncovered entity pairs, extracts contextual phrases, and groups them into candidate relation clusters.

schema [36]. This design enables lightweight graph refinement while preserving consistency with the original extraction results.

The key observation is that entity recognition and contextual evidence are already available from earlier extraction stages. Once adaptive schema discovery assigns canonical relation labels to the discovered relation clusters through the mapping $\mathcal{M}$, the remaining task is relation instantiation rather than full corpus reprocessing. To support this process, we maintain a cache $\mathcal{C}$ of contextual entries collected during the first extraction pass. Each entry is represented as (C_{ab}, v_a, v_b, i), where C_{ab} is the contextual phrase, $v_a, v_b \in \mathcal{V}$ are canonical entity identifiers, and i denotes the source document index. We also define a partial inverse relation map $\iota : R \to R \cup \{\perp\}$, where $\iota(r) = \perp$ indicates that no inverse relation is inserted. Let $T_0[i]$ denote the initial triple set extracted from document i, and let $T_{final}[i]$ denote the enriched triple set after post-hoc injection. Algorithm 2 summarizes the injection procedure.

For each cached context, the algorithm identifies the relation cluster previously assigned during adaptive schema discovery and retrieves the associated canonical relation label r. Contexts belonging to clusters without an assigned relation label are skipped. The algorithm then performs two duplicate checks: it first ensures that the entity pair is not already connected by another relation within the same document, and subsequently verifies that the identical triple does not already exist in the extracted knowledge graph. If both checks pass, the new triple is inserted into the corresponding document-level triple set. When the inverse relation map satisfies $\iota(r) \neq \perp$, the corresponding reverse triple is also inserted to support bidirectional graph traversal. By decoupling schema discovery from relation extraction, post-hoc triple injection enables efficient schema refinement through lightweight graph updates, improving the flexibility and efficiency of knowledge graph construction.

3.2 Cross-encoder-based Fact Reranking

After graph retrieval and PPR propagation, the system obtains a raw pool of candidate facts $\mathcal{F}_{raw}$ for downstream filtering and reasoning. Conventional GraphRAG pipelines often employ generative Large language models to evaluate retrieval relevance through natural language generation [11]. While generative reranking can improve retrieval effectiveness, structured output parsing failures in gen-

Algorithm 2 post-hoc triple injection

Input: Initial triples T_0, context cache $\mathcal{C}$, cluster-to-relation mapping $\mathcal{M}$, inverse relation map ι
Output: Enriched triples T_{final}

$T_{final} \leftarrow T_0$
for $(C_{ab}, v_a, v_b, i) \in \mathcal{C}$ **do**
    Identify the relation cluster k corresponding to C_{ab}
    if k has no assigned relation label **then**
        continue
    end if
    $r \leftarrow \mathcal{M}(k)$
    if (v_a, v_b) already has a relation in $T_{final}[i]$ **then**
        continue
    end if
    if (v_a, r, v_b) already exists in T_{final} **then**
        continue
    end if
    Insert (v_a, r, v_b) into $T_{final}[i]$
    if $\iota(r) \neq \perp$ **then**
        Insert $(v_b, \iota(r), v_a)$ into $T_{final}[i]$
    end if
end for

erative rerankers can silently degrade retrieval quality, and such approaches additionally incur external API costs and inference overhead [8].

To achieve stable and locally executable scoring, the proposed framework adopts a discriminative cross-encoder architecture based on the *cross-encoder/ms-marco-MiniLM-L-6-v2* model [23]. Unlike bi-encoder retrieval methods, which encode queries and candidate texts independently, a cross-encoder jointly processes both inputs in a single Transformer forward pass, enabling fine-grained token-level interactions between the query and the candidate evidence.

Given a query Q and a raw candidate fact set $\mathcal{F}_{raw} = \{f_1, f_2, \ldots, f_m\}$, where each fact is a triple $f_i = (s_i, r_i, o_i)$ with subject s_i, relation r_i, and object o_i, the reranker serializes each triple into a natural language string. The query–fact pair $(Q, \text{serialize}(f_i))$ is fed into the cross-encoder h_θ, which outputs a scalar relevance score $S_{\text{CE}}(Q, f_i) = h_\theta(Q, \text{serialize}(f_i))$. Facts are sorted in descending order of S_{CE} to produce $\mathcal{F}_{ranked}$. For passage-level selection, each passage p receives the maximum cross-encoder score among its associated facts; if no associated fact is present, the passage is ranked using its retrieval and graph-based scores only.

We retain this reranking stage for two reasons. First, the cross-encoder produces deterministic scalar scores, avoiding both the output instability of generative reranking and external API dependencies. Second, joint query–fact encoding captures subtle lexical and relational interactions that independent embedding similarity may miss, which is particularly relevant for multi-hop reasoning where fact relevance often depends on fine-grained alignment with the query [23].

3.3 Graph-Path Topological Routing

After retrieval and reranking, the system selects a compact evidence subset to condition answer generation. In multi-hop QA, effective context selection should preserve connectivity among distributed pieces of evidence rather than relying solely on local passage relevance [30, 35].

We propose Graph-Path Topological Routing as the primary context selection mechanism. For comparison, we evaluate standard alternatives such as Reciprocal Rank Fusion (RRF) [4] and bridging-based expansion [1]. RRF ranks passages by summing reciprocal ranks with its standard stabilizing constant, while bridging expansion greedily adds passages that share canonical entities or graph-adjacent entities with already selected evidence until the context budget is filled. These baselines are included to assess whether graph topology alone can provide useful guidance for evidence selection.

Graph-Path uses bounded breadth-first search (BFS) as a lightweight topological routing mechanism. Unlike relevance-based selectors that rely on learned scores or manually tuned scoring rules, Graph-Path evaluates whether candidate evidence remains structurally reachable from query entities within the knowledge graph. This design deliberately minimizes modeling assumptions, allowing performance to reflect graph quality rather than selector complexity.

Let $\mathcal{P} = \{p_1, \ldots, p_n\}$ be the candidate passage set after retrieval and reranking, where $n = |\mathcal{P}|$. Let $\mathcal{E}(x)$ denote the set of unique canonical entity nodes extracted from text x after entity normalization; therefore $\mathcal{E}(Q)$ is the query-entity set and $\mathcal{E}(p_i)$ is the entity set of passage p_i. BFS traversal is performed on the undirected projection of $G = (\mathcal{V}, \mathcal{A})$, which contains an undirected edge $\{u, v\}$ whenever $(u, r, v) \in \mathcal{A}$ or $(v, r, u) \in \mathcal{A}$ for any relation r, so that both original and inverse-injected triples support connectivity. The traversal is constrained by a maximum hop radius $H \in \mathbb{N}^+$. For each entity $e \in \mathcal{E}(p_i)$, its distance to the query neighborhood is $\text{dist}_G(e, \mathcal{E}(Q)) = \min_{q \in \mathcal{E}(Q)} \text{dist}_G(e, q)$, where $\text{dist}_G(e, q)$ is the shortest-path length between two graph nodes and unreachable nodes have distance $+\infty$. The same hop limit H is used both to bound BFS expansion and to determine whether an entity is considered reachable from the query neighborhood.

Intuitively, passages that remain closely connected to the query region of the graph are more likely to contribute useful evidence for multi-hop reasoning. Graph-Path therefore measures the fraction of unique passage entities that are reachable from the query neighborhood within the hop limit H. Normalizing by $|\mathcal{E}(p_i)|$ prevents passages with many extracted entities from receiving artificially high scores solely because they contain more entities. For passages with no extracted entities, we define $S_{\text{path}}(Q, p_i) = 0$. Otherwise, the passage score is defined as

$$S_{\text{path}}(Q, p_i) = \frac{|\{e \in \mathcal{E}(p_i) \mid \text{dist}_G(e, \mathcal{E}(Q)) \leq H\}|}{|\mathcal{E}(p_i)|}. \tag{1}$$

The numerator counts the unique passage entities reachable from the query neighborhood within H hops, and the denominator is the number of unique entities in the passage. With the empty-entity convention above, the score lies in $[0, 1]$ because the numerator is a subset cardinality of $\mathcal{E}(p_i)$. Candidate passages are sorted in descending order of S_{path}; ties are broken by the cross-encoder rank when available and otherwise by the original dense-retrieval rank. The top passages that fit the fixed context budget are passed to the generator.

We do not present BFS routing as a new retrieval algorithm. Instead, Graph-Path serves as a controlled probe of graph quality. If a simple topology-driven selector performs competitively against stronger relevance-oriented baselines, this provides indirect evidence that the automatically constructed graph preserves meaningful multi-hop connectivity and provides useful structural guidance for evidence selection.

4 Experimental Results

4.1 Experimental Setup

4.1.1 Datasets

To evaluate the effectiveness and robustness of the proposed framework, we select three widely adopted multi-hop QA benchmarks that span diverse reasoning structures and retrieval challenges. All datasets are based on Wikipedia and require aggregating evidence from multiple passages. The secondary datasets are openly accessible from their official sources: MuSiQue at `https:`

Table 1: Evaluation subsets.

Dataset	#Queries	#Documents	Task Characteristic
MuSiQue	1,000	11,656	2–4 hops
2WikiMultihopQA	1,000	6,119	Relational paths
HotpotQA	1,000	9,811	Multi-evidence

//github.com/StonyBrookNLP/musique, 2WikiMultihopQA at https://github.com/Alab-NII/2wikimultihop, and HotpotQA at https://hotpotqa.github.io/.

MuSiQue [30] is a challenging compositional multi-hop QA dataset requiring reasoning chains of approximately two to four supporting documents. Unlike datasets containing shortcut patterns, MuSiQue explicitly minimizes annotation artifacts and enforces genuine compositional reasoning, making it particularly suitable for evaluating graph-based retrieval and reasoning.

2WikiMultihopQA [13] consists of questions constructed from Wikipedia articles that require reasoning across multiple entities and relational paths. The dataset frequently involves biographical, geographical, and hierarchical relations, making it well-suited for assessing knowledge graph traversal and relational reasoning.

HotpotQA [35] is a widely adopted multi-hop QA benchmark containing Wikipedia-based questions that require evidence aggregation from multiple passages. Compared with MuSiQue, HotpotQA often features denser evidence connectivity and more direct reasoning paths, allowing evaluation under a complementary multi-hop setting.

We evaluate on three Wikipedia-based multi-hop QA benchmarks.

4.1.2 Evaluation Metrics

Retrieval. We evaluate retrieval quality using Recall@k for $k \in \{1,2,5,10\}$. For a query Q, let $\mathrm{TopK}(Q)$ denote the top-k retrieved passages and $\mathcal{S}_Q$ denote the gold supporting passages. Recall@k is computed as

$$\text{Recall@}\,k(Q) = \frac{|\,\mathrm{TopK}(Q) \cap \mathcal{S}_Q|}{|\mathcal{S}_Q|}. \qquad (2)$$

Reported Recall@k values are macro-averaged over all evaluation queries.

Answer generation. We report Exact Match (EM) and token-level F1 following the standard QA evaluation protocols used in HotpotQA and MuSiQue [30, 35]. Predictions and references are normalized by lowercasing, removing punctuation and articles, and collapsing repeated whitespace. EM is 1 if the normalized prediction exactly matches a normalized reference answer, and 0 otherwise.

Token-level F1 is computed from token overlap between the normalized prediction and reference answer. For questions with multiple valid references, EM and F1 are computed against each reference and the maximum score is retained. Dataset-level EM and F1 are macro-averaged over all examples, with F1 treated as the primary metric because it is more robust to minor lexical variation.

4.1.3 Implementation Details

All experiments are run on a single consumer-grade machine equipped with an NVIDIA GeForce RTX 2060 GPU (6 GB of VRAM). This resource-constrained setup is intentionally selected to demonstrate the feasibility of deploying GraphRAG architectures on consumer hardware without relying on large-scale cloud infrastructure.

Graph construction. We use GLiNER2 [36] for schema-guided entity and relation extraction, with a seed schema of 55 relation types after regularization. Adaptive schema discovery is applied with minimum context frequency $\theta_{\text{sup}} = 5$ and minimum cluster size

$\theta_{\text{cluster}} = 5$; AHC uses cosine distance and average linkage.

Retrieval and reranking. Dense retrieval is performed with BGE embeddings [33] using FAISS indexing over the Wikipedia corpus. The initial retrieval pool consists of top-200 passages per query, and graph-expanded candidates are obtained from the PPR neighborhood of linked query entities. These candidates are then reranked by a local cross-encoder model *cross-encoder/ms-marco-MiniLM-L-6-v2* [23], which produces a ranked list of passages.

Graph-path selector. We adopt the Graph-Path selector with a traversal radius of $H = 4$ hops (Section 3). This value was selected based on the MuSiQue sensitivity analysis (Section 4.5.3) and reused across all datasets to balance retrieval coverage against topological noise. The underlying knowledge graph is the final output after post-hoc injection, containing both seed and adaptively discovered relations.

Answer generation. For final answer generation, we use GPT-4o-mini via API with temperature 0 and maximum output length 150 tokens. The model receives as input the selected passages and their associated triples, formatted as a grounded reasoning context. All selector variants use the same final context budget and generation settings.

4.2 Graph Topology Analysis

We first compare the structural characteristics of knowledge graphs generated by GPT-4o-mini OpenIE and the proposed GLiNER2-based adaptive schema pipeline on the MuSiQue corpus.

Table 2: Graph topology statistics on MuSiQue.

Metric	GPT-4o-mini OpenIE	GLiNER2 + adaptive schema
Entity nodes	101,641	41,084
Relation triples	127,640	83,680
Relation types	30,210	148
Graph density	0.000100	**0.000334**
Average node degree	11.38	**17.62**

The proposed pipeline produces a substantially more compact graph than GPT-4o-mini OpenIE, reducing both the number of entity nodes and extracted relation triples. Despite this reduction in graph size, the graph retains a rich relational structure through schema-guided relation construction and adaptive schema expansion.

The most significant difference appears in the relation space. GPT-4o-mini OpenIE generates 30,210 distinct relation labels, whereas the proposed schema-driven pipeline produces only 148 relation types. Of these, 55 originate from the seed schema and 93 are discovered through adaptive schema expansion. Rather than allowing unrestricted predicate proliferation, the schema-guided approach constrains relations to a structured inventory while still extending coverage through corpus-specific relation discovery.

Node degree and density are computed over the full graph, including chunk nodes and synonymy edges, not solely entity–relation triples. The resulting graph exhibits higher average degree (17.62 versus 11.38) and density (0.000334 versus 0.000100). While part of this increase can be attributed to the smaller graph size, the substantial reduction in relation fragmentation suggests that entities are organized within a more consistent relation space rather than being distributed across thousands of sparsely used predicate variants.

Overall, the results indicate that adaptive schema discovery primarily improves graph consistency rather than extraction volume. By reducing predi-

cate proliferation and constructing a more coherent relation inventory, the proposed approach produces a graph structure that may better support topology-aware retrieval methods such as Graph-Path, where retrieval quality depends on the ability to traverse semantically coherent paths between related entities.

4.3 Main Results

Table 3 reports end-to-end QA results. We denote the full configuration (GLiNER2 + adaptive schema + Graph-Path, $H = 4$) as *Adaptive Graph-Path*; all results use 1,000 queries per dataset.

Table 3: End-to-end QA results

Dataset	Method	Answer Generation	
		EM	F1
MuSiQue	HippoRAG2 (BGE)	0.3180	0.4498
	Adaptive Graph-Path	**0.3220**	**0.4577**
2WikiMultihopQA	HippoRAG2 (BGE)	0.5698	0.6559
	Adaptive Graph-Path	**0.5700**	**0.6637**
HotpotQA	HippoRAG2 (BGE)	0.5580	0.7085
	Adaptive Graph-Path	**0.5610**	**0.7154**

Adaptive Graph-Path consistently improves end-to-end QA performance across all evaluated datasets. Compared with HippoRAG2 (BGE), F1 increases from 0.4498 to 0.4577 on MuSiQue, from 0.6559 to 0.6637 on 2WikiMultihopQA, and from 0.7085 to 0.7154 on HotpotQA. Similar trends are observed for Exact Match (EM), indicating that the proposed framework provides consistent improvements across different multi-hop QA benchmarks.

The magnitude of improvement varies only slightly across datasets. The largest absolute F1 gain is observed on MuSiQue (+0.0079), followed closely by 2WikiMultihopQA (+0.0078), while HotpotQA shows a smaller gain of +0.0069. These differences suggest that the proposed framework provides consistent benefits across benchmarks, with the strongest effect on the dataset that requires more compositional multi-hop reasoning.

MuSiQue requires compositional reasoning across multiple supporting documents with limited lexical overlap, whereas HotpotQA generally contains more densely connected supporting evidence. Although the degree of improvement differs, Adaptive Graph-Path consistently outperforms the HippoRAG2 (BGE) baseline on all three benchmarks. Overall, the results indicate that combining adaptive graph construction with Graph-Path retrieval provides a consistent benefit for downstream multi-hop question answering.

4.4 Comparison with Retrieval Baselines

Table 4 compares Adaptive Graph-Path with conventional MuSiQue retrievers using the same corpus and GPT-4o-mini reader. RAPTOR is included as a hierarchical retrieval baseline because it recursively clusters and summarizes passages to support retrieval at multiple abstraction levels [28].

Table 4: MuSiQue retrieval baselines

Method	Retrieval				Answer Generation	
	R@1	R@2	R@5	R@10	EM	F1
BM25	0.2520	0.3346	0.4326	0.5107	0.2110	0.3152
ColBERTv2	0.2348	0.3089	0.4207	0.4967	0.2160	0.3253
Contriever	0.2416	0.3482	0.4662	0.5512	0.2180	0.3140
RAPTOR	0.2753	0.3838	0.5200	0.5955	0.2590	0.3714
Adaptive Graph-Path	**0.2793**	**0.4243**	**0.5678**	**0.6423**	**0.3220**	**0.4577**

Adaptive Graph-Path obtains higher scores than the conventional retrievers in this MuSiQue setting across both retrieval and answer generation metrics. The clearest difference appears at higher recall cutoffs, where Recall@10 increases from 0.5955 with RAPTOR to 0.6423. In contrast, the gain at Recall@1 is relatively modest (0.2793 versus 0.2753). This pattern suggests that graph expansion is most useful for recovering additional supporting evidence,

rather than for substantially changing the top-ranked passage.

The downstream QA results follow a similar trend. Although retrieval improvements are observed across all recall levels, the corresponding F1 difference relative to RAPTOR is larger (0.4577 versus 0.3714), suggesting that the additional passages retrieved through graph traversal can provide useful reasoning evidence rather than only redundant context. This observation is consistent with the requirements of multi-hop QA, where successful reasoning often depends on assembling multiple supporting facts rather than identifying a single highly relevant passage. The results therefore suggest that Adaptive Graph-Path mainly helps by improving evidence coverage in this experimental setting.

4.5 Ablation Studies

We conduct three ablation analyses to isolate the contribution of key components: adaptive schema expansion, context selection strategies, and traversal radius.

4.5.1 Effect of Adaptive Schema Expansion

Table 5 isolates adaptive schema expansion across datasets.

Table 5: Adaptive schema ablation

Dataset	Configuration	Injected Triples	New Relations	EM	F1	ΔF1
MuSiQue	Seed-only	–	–	0.3020	0.4352	–
	+ Adaptive	1,691	93	0.3220	0.4577	+0.0225
2WikiMultihopQA	Seed-only	–	–	0.5640	0.6561	–
	+ Adaptive	1,874	109	0.5700	0.6637	+0.0076
HotpotQA	Seed-only	–	–	0.5609	0.7152	–
	+ Adaptive	874	43	0.5610	0.7154	+0.0002

The effect of adaptive schema expansion differs across datasets. On MuSiQue, adaptive schema discovery identifies 93 additional relation types and injects 1,691 triples, improving F1 by 0.0225. On 2WikiMultihopQA, 109 discovered relation types and 1,874 injected triples improve F1 by 0.0076. On HotpotQA, 43 additional relation types and 874 injected triples result in a marginal F1 improvement of 0.0002.

The contribution of adaptive schema expansion varies across datasets. The largest improvement is observed on MuSiQue, suggesting that corpus-specific relation discovery provides additional reasoning connections beyond the predefined schema. In contrast, the gains on 2WikiMultihopQA and especially HotpotQA are relatively small, indicating that the seed schema already captures much of the useful relational structure. Nevertheless, adaptive schema expansion does not degrade performance on any dataset and consistently provides non-negative improvements over the seed schema.

4.5.2 Context Selection Ablation

To isolate the effect of the context selection strategy, all experiments in this section are conducted using the same GLiNER2-based adaptive graph. Only the context selector is varied while all other components remain unchanged.

Table 6: Ablation of context selection strategies on the proposed GLiNER2-based adaptive graph (MuSiQue).

Context Selector	F1	EM
raw BFS	0.3932	0.2710
RRF + CE	0.4280	0.2990
Bridging + CE	0.4366	0.2990
Graph-Path (H=4)	**0.4577**	**0.3220**

Graph-Path obtains the highest score among the evaluated context selection strategies in this setting, reaching an F1 score of 0.4577 and an EM score of 0.3220. Compared with RRF + CE and Bridg-

ing + CE, the results suggest that the constructed graph contains useful topological signal for selecting supporting contexts. This should be interpreted as evidence of graph coherence rather than as a claim that BFS-based routing is generally superior to relevance-oriented selection methods.

The comparison with raw BFS further illustrates the role of routing constraints. While simple graph traversal achieves 0.3932 F1, Graph-Path reaches 0.4577 F1 by using path-guided exploration of the adaptive graph. This result suggests that the graph contains structural information that can be useful for topology-guided context selection, although the method should be viewed primarily as a diagnostic probe of graph quality rather than as a standalone retrieval advance.

4.5.3 Traversal Radius Sensitivity

Table 7 reports sensitivity on MuSiQue to the BFS hop limit H.

Table 7: Sensitivity of BFS traversal radius for Graph-Path on MuSiQue.

Radius (H)	**F1**	**EM**
$H = 2$	0.4125	0.2800
$H = 3$	0.4455	0.3080
$H = 4$	**0.4577**	**0.3220**
$H = 5$	0.4528	0.3150

Performance peaks at $H = 4$; the slight drop at $H = 5$ is consistent with semantic drift from expanding to more distant nodes. This pattern reveals a coverage–noise trade-off in graph retrieval: useful reasoning paths require enough hops to reach bridge entities, but excessive expansion dilutes the context with weakly related evidence. Traversal depth should therefore be treated as a reasoning-control parameter, not merely a search-radius hyperparameter.

5 Discussion

The results suggest that graph construction quality is more important than extraction volume for multi-hop retrieval. Although GPT-4o-mini OpenIE produces many more relation types, the local GLiNER2-based pipeline yields a more compact relation space, which is better suited for traversal. In graph-based RAG, useful structure depends on stable, reusable edges rather than a large number of noisy predicate variants.

The experiments also indicate that local, schema-guided extraction can be a practical alternative to LLM-based OpenIE in the evaluated setting. The final answer generator still uses GPT-4o-mini, but extraction, graph construction, retrieval, and reranking are executed locally. This finding indicates that retrieval graphs do not necessarily require open-ended generative extraction when relation labels are controlled and adaptively extended.

Adaptive schema expansion is most useful when the seed schema misses corpus-specific relations. This explains the gains on MuSiQue and 2WikiMultihopQA, as well as the near-zero change on HotpotQA, where only a small number of new relations are discovered. Therefore, adaptive schema discovery should be viewed as a mechanism for closing schema gaps, not as an automatic improvement for every dataset.

Finally, the Graph-Path results should be interpreted as a diagnostic view of graph coherence. Compared with raw BFS, Graph-Path achieves higher QA performance in the reported ablation, suggesting that the adaptive graph places some relevant passages within reachable neighborhoods of the query entities. This does not imply that BFS-style routing is inherently superior; rather, it indicates that preserving coherent relational structure can make graph topology useful as a lightweight signal for evidence selection.

6 Conclusion

This paper has presented a graph-enhanced QA framework for Wikipedia-based multi-hop question answering. The framework combines local GLiNER2-based knowledge graph construction, Adaptive Schema Discovery with Post-hoc Triple Injection, cross-encoder reranking, and Graph-Path retrieval within a unified RAG pipeline.

The evaluation demonstrates that structured graph retrieval can improve evidence selection for questions whose supporting passages are distributed across multiple entities. Rather than relying on extraction volume alone, the results highlight the importance of building a navigable graph with coherent relation labels and sufficient schema coverage. Adaptive expansion is useful when the seed schema misses corpus-specific relations, while topology-based routing provides a lightweight way to test whether the constructed graph contains meaningful reasoning paths.

The current system still relies on GPT-4o-mini for final answer generation, so it is not yet a fully local QA pipeline. Future work will replace the generator with a local LLM, explore dynamic hop-radius selection based on query complexity, and evaluate the framework on larger and more diverse knowledge-intensive QA settings.

Data Availability Statement

No new primary data were collected for this study. The secondary datasets analyzed are openly available from their official sources: MuSiQue (`https://github.com/StonyBrookNLP/musique`), 2WikiMultihopQA (`https://github.com/Alab-NII/2wikimultihop`), and HotpotQA (`https://hotpotqa.github.io/`). The 1,000-question evaluation subsets reported in this manuscript were drawn from these publicly released datasets.

References

[1] Asai, A., Hashimoto, K., Hajishirzi, H., Socher, R., Xiong, C.: Learning to retrieve reasoning paths over wikipedia graph for question answering. In: International Conference on Learning Representations (2020)

[2] Bai, J., Fan, W., Hu, Q., Zong, Q., Li, C., Tsang, H.T., Luo, H., Yim, Y., Huang, H., Zhou, X., Qin, F., Zheng, T., Peng, X., Yao, X., Yang, H., Wu, L., Ji, Y., Zhang, G., Chen, R., Song, Y.: AutoSchemaKG: Autonomous knowledge graph construction through dynamic schema induction from web-scale corpora. arXiv preprint arXiv:2505.23628 (2025)

[3] Chen, S., Zhou, C., Yuan, Z., Zhang, Q., Cui, Z., Chen, H., Xiao, Y., Cao, J., Huang, X.: You don't need pre-built graphs for RAG: Retrieval augmented generation with adaptive reasoning structures. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 40, pp. 30,270–30,278 (2026). DOI 10.1609/aaai.v40i36.40278

[4] Cormack, G.V., Clarke, C.L.A., Buettcher, S.: Reciprocal rank fusion outperforms condorcet and individual rank learning methods. In: Proceedings of the 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 758–759 (2009). DOI 10.1145/1571941.1572114

[5] Edge, D., Trinh, H., Cheng, N., Bradley, J., Chao, A., Mody, A., Truitt, S., Metropolitansky, D., Ness, R.O., Larson, J.: From local to global: A graph RAG approach to

query-focused summarization. arXiv preprint arXiv:2404.16130 (2024)

[6] Ehrlinger, L., Wöß, W.: Towards a definition of knowledge graphs. In: Joint Proceedings of the Posters and Demos Track of the 12th International Conference on Semantic Systems, *CEUR Workshop Proceedings*, vol. 1695 (2016)

[7] Fan, T., Wang, J., Ren, X., Huang, C.: MiniRAG: Towards extremely simple retrieval-augmented generation. arXiv preprint arXiv:2501.06713 (2025)

[8] Gao, Y., Xiong, Y., Gao, X., Jia, K., Pan, J., Bi, Y., Dai, Y., Sun, J., Wang, H.: Retrieval-augmented generation for large language models: A survey. arXiv preprint arXiv:2312.10997 (2023)

[9] Gui, H., Yuan, L., Ye, H., Zhang, N., Sun, M., Liang, L., Chen, H.: IEPile: Unearthing large-scale schema-based information extraction corpus. In: Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (2024). DOI 10.48550/arXiv.2402.14710

[10] Guo, Z., Xia, L., Yu, Y., Ao, T., Huang, C.: LightRAG: Simple and fast retrieval-augmented generation. arXiv preprint arXiv:2410.05779 (2024)

[11] Gutiérrez, B.J., Shu, Y., Gu, Y., Yasunaga, M., Su, Y.: HippoRAG: Neurobiologically inspired long-term memory for large language models. In: Advances in Neural Information Processing Systems (2024)

[12] He, X., Tian, Y., Sun, Y., Chawla, N.V., Laurent, T., LeCun, Y., Bresson, X., Hooi, B.: G-Retriever: Retrieval-augmented generation for textual graph understanding and question answering. arXiv preprint arXiv:2402.07630 (2024)

[13] Ho, X., Nguyen, A.K.D., Sugawara, S., Aizawa, A.: Constructing a multi-hop qa dataset for comprehensive evaluation of reasoning steps. In: Proceedings of the 28th International Conference on Computational Linguistics, pp. 6609–6625 (2020). DOI 10.18653/v1/2020.coling-main.580

[14] Izacard, G., Caron, M., Hosseini, L., Riedel, S., Bojanowski, P., Joulin, A., Grave, E.: Unsupervised dense information retrieval with contrastive learning. arXiv preprint arXiv:2112.09118 (2021)

[15] Jeong, S., Baek, J., Cho, S., Hwang, S.J., Park, J.C.: Adaptive-RAG: Learning to adapt retrieval-augmented large language models through question complexity. In: Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pp. 7036–7050 (2024). DOI 10.18653/v1/2024.naacl-long.389

[16] Ji, S., Pan, S., Cambria, E., Marttinen, P., Philip, S.Y.: A survey on knowledge graphs: Representation, acquisition, and applications. IEEE Transactions on Neural Networks and Learning Systems **33**(2), 494–514 (2021). DOI 10.1109/TNNLS.2021.3070843

[17] Karpukhin, V., Oguz, B., Min, S., Lewis, P., Wu, L., Edunov, S., Chen, D., Yih, W.t.: Dense passage retrieval for open-domain question answering. In: Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, pp. 6769–6781 (2020). DOI 10.18653/v1/2020.emnlp-main.550

[18] Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., Küttler, H., Lewis, M., Yih, W.t., Rocktäschel, T., Riedel, S., Kiela, D.: Retrieval-augmented generation for knowledge-intensive NLP tasks. In: Advances in Neural Information Processing Systems, vol. 33, pp. 9459–9474 (2020)

[19] Liu, N.F., Lin, K., Hewitt, J., Paranjape, A., Bevilacqua, M., Petroni, F., Liang, P.: Lost in the middle: How language models use long contexts. Transactions of the Association for Computational Linguistics **12**, 157–173 (2024). DOI 10.1162/tacl_a_00638

[20] Ma, S., Xu, C., Jiang, X., Li, M., Qu, H., Yang, C., Mao, J., Guo, J.: Think-on-graph 2.0: Deep and faithful large language model reasoning with knowledge-guided retrieval augmented generation. arXiv preprint arXiv:2407.10805 (2024). DOI 10.48550/arXiv.2407.10805

[21] Mavromatis, C., Karypis, G.: GNN-RAG: Graph neural retrieval for large language model reasoning. arXiv preprint arXiv:2405.20139 (2024)

[22] Müllner, D.: Modern hierarchical, agglomerative clustering algorithms. arXiv preprint arXiv:1109.2378 (2011)

[23] Nogueira, R., Cho, K.: Passage re-ranking with BERT. arXiv preprint arXiv:1901.04085 (2019)

[24] Page, L., Brin, S., Motwani, R., Winograd, T.: The PageRank citation ranking: Bringing order to the web. Tech. rep., Stanford InfoLab (1999)

[25] Paulheim, H.: Knowledge graph refinement: A survey of approaches and evaluation methods. Semantic Web **8**(3), 489–508 (2017). DOI 10.3233/SW-160218

[26] Peng, B., Zhu, Y., Liu, Y., Bo, X., Shi, H., Hong, C., Zhang, Y., Tang, S.: Graph retrieval-augmented generation: A survey. arXiv preprint arXiv:2408.08921 (2024)

[27] Reimers, N., Gurevych, I.: Sentence-BERT: Sentence embeddings using siamese BERT-networks. In: Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, pp. 3982–3992 (2019). DOI 10.18653/v1/D19-1410

[28] Sarthi, P., Abdullah, S., Tuli, A., Khanna, S., Goldie, A., Manning, C.D.: RAPTOR: Recursive abstractive processing for tree-organized retrieval. In: International Conference on Learning Representations (2024)

[29] Shen, Z., Diao, C., Vougiouklis, P., Merita, P., Piramanayagam, S., Chen, E., Graux, D., Melo, A., Lai, R., Jiang, Z., Li, Z., Qi, Y., Ren, Y., Tu, D., Pan, J.Z.: GEAR: Graph-enhanced agent for retrieval-augmented generation. In: Findings of the Association for Computational Linguistics: ACL 2025, pp. 12,049–12,072. Association for Computational Linguistics (2025)

[30] Trivedi, H., Balasubramanian, N., Khot, T., Sabharwal, A.: MuSiQue: Multihop questions via single-hop question composition. Transactions of the Association for Computational Linguistics **10**, 539–554 (2022). DOI 10.1162/tacl_a_00475

[31] Wang, C., Deng, W., Guan, W., Lu, Q., Jiang, N.: Cross-granularity hypergraph retrieval-augmented generation for multi-hop question answering. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 40,

pp. 33,368–33,376 (2026). DOI 10.1609/aaai.v40i39.40623

[32] Xiang, Z., Wu, C., Zhang, Q., Chen, S., Hong, Z., Huang, X., Su, J.: When to use graphs in RAG: A comprehensive analysis for graph retrieval-augmented generation. arXiv preprint arXiv:2506.05690 (2025)

[33] Xiao, S., Liu, Z., Zhang, P., Muennighoff, N.: C-Pack: Packaged resources to advance general chinese embedding. arXiv preprint arXiv:2309.07597 (2023)

[34] Yan, S.Q., Gu, J.C., Zhu, Y., Ling, Z.H.: Corrective retrieval augmented generation. arXiv preprint arXiv:2401.15884 (2024)

[35] Yang, Z., Qi, P., Zhang, S., Bengio, Y., Cohen, W.W., Salakhutdinov, R., Manning, C.D.: HotpotQA: A dataset for diverse, explainable multi-hop question answering. In: Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, pp. 2369–2380 (2018). DOI 10.18653/v1/D18-1259

[36] Zaratiana, U., Pasternak, G., Boyd, O., Hurn-Maloney, G., Lewis, A.: GLiNER2: Schema-driven multi-task learning for structured information extraction. In: Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing: System Demonstrations, pp. 130–140. Association for Computational Linguistics (2025)

[37] Zaratiana, U., Tomeh, N., Holat, P., Charnois, T.: GLiNER: Generalist model for named entity recognition using bidirectional transformer. In: Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, pp. 5364–5376 (2024). DOI 10.18653/v1/2024.naacl-long.300

[38] Zhang, Q., Chen, S., Bei, Y., Yuan, Z., Zhou, H., Hong, Z., Chen, H., Xiao, Y., Zhou, C., Dong, J., Chang, Y., Huang, X.: A survey of graph retrieval-augmented generation for customized large language models. arXiv preprint arXiv:2501.13958 (2025). DOI 10.48550/arXiv.2501.13958

[39] Zhang, S.H., Lin, J.W.: Tracert-retrieval-augmented generation: Boosting multi-hop retrieval-augmented generation with direction-aware graph traversal. Engineering Proceedings **120**(1), 47 (2025). DOI 10.3390/engproc2025120047

[40] Zhou, Y., Su, Y., Sun, Y., Wang, S., Wang, T., He, R., Zhang, Y., Liang, S., Liu, X., Ma, Y., Fang, Y.: In-depth analysis of graph-based RAG in a unified framework. Proceedings of the VLDB Endowment **18**(11) (2025). DOI 10.14778/3773731.3773738