TraceRAG: Targeted Graph Traversal via Adaptive Query Resolution for Multi-Hop Retrieval

1st Giuseppe Trimigno
Dept. of Engineering and Architecture
University of Parma
Parma, Italy
giuseppe.trimigno@unipr.it

2nd Gianfranco Lombardo
Dept. of Engineering and Architecture
University of Parma
Parma, Italy
gianfranco.lombardo@unipr.it

3rd Stefano Cagnoni
Dept. of Engineering and Architecture
University of Parma
Parma, Italy
stefano.cagnoni@unipr.it

***Abstract*—Retrieval-Augmented Generation (RAG) has become the standard for extending Large Language Models (LLMs) with non-parametric, up-to-date knowledge. Standard RAG systems face challenges with multi-hop reasoning, which involves combining information from separate and unconnected passages to derive answers. While recent graph-based approaches leverage Knowledge Graphs (KGs) to bridge these gaps, they typically rely on single-shot global query embeddings to initialize retrieval, which often leads to semantic drift and weak modeling of sequential dependencies. We introduce TraceRAG, a novel framework that reframes multi-hop retrieval as an adaptive and sequential resolution problem. Given a complex query, we decompose it into atomic sub-questions, which are resolved sequentially by grounding them in factual triples extracted from the knowledge graph. These grounded triples serve as high-precision structural seeds for a Personalized PageRank (PPR) traversal, enabling controlled and interpretable graph exploration across hops. By explicitly binding intermediate resolution steps to verified structure, our framework mitigates semantic drift and preserves the query intent throughout retrieval. Extensive experiments on MuSiQue, HotpotQA, and 2WikiMultiHopQA demonstrate that TraceRAG consistently outperforms the strongest baselines, achieving gains of up to 3.5 for Recall@5 and 4.6 for F1 score. Moreover, it remains robust even when the underlying graph is derived from smaller, open-weights models, highlighting its effectiveness and enabling scalable, structured RAG systems.**

***Index Terms*—RAG, Knowledge Graphs, KG-RAG, Large Language Models, Multi-hop QA, Multi-hop Retrieval**

I. Introduction

Large Language Models (LLMs) have demonstrated remarkable capabilities in natural language understanding and generation. However, they are fundamentally constrained by their static parametric memory, which limits their ability to access real-time information and makes them prone to factual inconsistencies [1]–[3]. Traditional parametric approaches, such as Continual Fine-Tuning or Model Editing, often struggle with catastrophic forgetting and incur prohibitive computational costs when updates are frequent [4], [5]. Retrieval-Augmented Generation (RAG) has emerged as the leading paradigm to address these limitations [6], [7]. By decoupling reasoning from memory, and grounding generation in external, up-to-date knowledge, RAG significantly mitigates hallucinations. This approach further enables models to adapt to evolving domains without the prohibitive cost or risks associated with retraining [8]–[10].

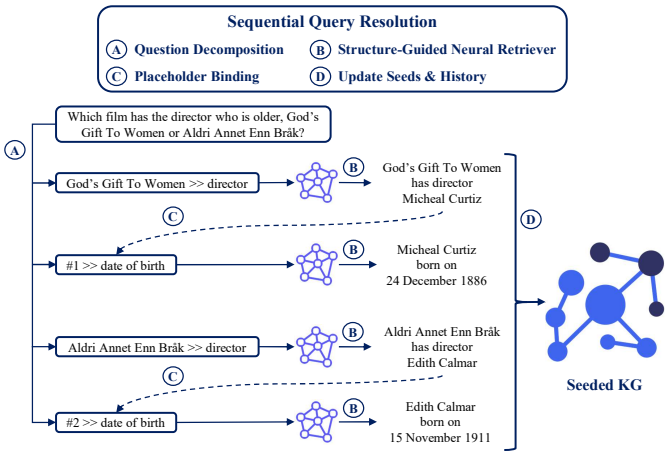

Fig. 1. Instead of relying on a single global query embedding, **TraceRAG** employs **Sequential Query Resolution** to decompose the complex query Q into dependent sub-tasks. Each sub-question is grounded in the Knowledge Graph using **Structure-Guided Neural Retrieval**, which selects high-precision triples to bind placeholders for subsequent steps. These verified triples act as specific seeds for the Personalized PageRank (PPR) algorithm, controlling the graph traversal to the correct answer, effectively mitigating semantic drift.

While standard RAG excels at *single-hop* retrieval, where the answer resides explicitly within a single passage, it faces significant challenges with *multi-hop reasoning*. In these complex scenarios, the required information is often distributed over disjoint passages that share no lexical overlap, requiring the retrieval system to synthesize intermediate evidence to reach the final answer. Standard dense retrieval methods, which rely on semantic similarity between a query and individual documents, often fail to bridge these disconnected pieces of evidence, resulting in incomplete or irrelevant context.

To overcome this reasoning gap, recent research has turned to Knowledge Graphs (KGs) as a structured intermediary. By modeling the explicit relationships between entities, KGs provide a "reasoning substrate" that dense vectors lack. State-of-the-art methods, such as HippoRAG [11] and HippoRAG2 [12], leverage this structure by employing Personalized PageRank (PPR) [13] to propagate relevance signals across the graph, effectively linking disparate passages via shared entities.

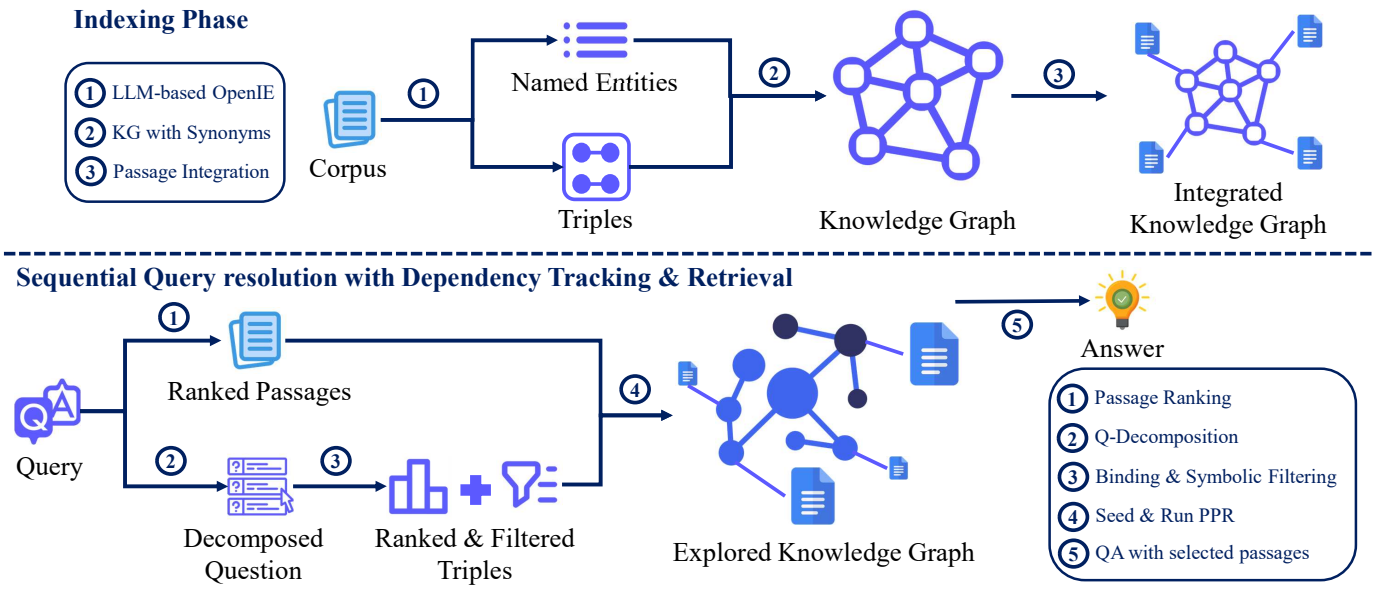

Fig. 2. **The TraceRAG Framework.** The pipeline consists of two phases: **Indexing**, where an LLM-based OpenIE module extracts triples from the corpus to construct a schemaless KG integrated with source passages. **Sequential Query Resolution with Dependency Tracking & Retrieval**, where the complex query is decomposed into dependent sub-questions. We employ a sequential process to bind placeholders and verify triples using a symbolic structure filter. These triples serve as structural seeds for the PPR algorithm, which propagates relevance to identify the final context for generating the final answer.

However, these graph-based approaches rely on *global semantic matching*, encoding complex multi-hop queries into a single vector for graph traversal. This one-shot representation is prone to semantic drift, favoring dominant keywords over the sequential reasoning needed for multi-hop inference. To mitigate retrieval noise, existing methods often depend on 70B+ LLMs for post-filtering, limiting real-world deployment.

TraceRAG. In this paper, we propose **TraceRAG** (**T**argeted **R**esolution via **A**daptive **C**hain **E**xploration) a novel framework that shifts the paradigm from global matching to *sequential query resolution with dependency tracking* (Figure 1). We posit that multi-hop retrieval is effectively a decomposition-and-binding problem. We decompose complex queries into atomic sub-questions, sequentially resolving dependencies by grounding them in factual triples extracted from the KG. Crucially, these grounded triples serve as highly specific seed nodes for the PPR algorithm, ensuring that the graph traversal starts from verified evidence rather than noisy semantic matches. Figure 2 shows how our framework is structured.

Contributions. Our contributions are summarized as follows. (1) **Sequential Query Resolution:** We introduce a structure-guided mechanism that dynamically updates sub-tasks by binding "placeholders" to verified entities, ensuring graph traversal follows a coherent logical chain; (2) **Structure-Guided Neural Retrieval:** We replace the computationally prohibitive 70B-parameter LLM filters with a lighter Cross-Encoder Reranker. This design choice enables high-precision triple selection without relying on massive general-purpose models. (3) **SOTA Performance:** We conduct evaluation on three benchmark datasets (MuSiQue, HotpotQA, 2WikiMultiHopQA). TraceRAG achieves state-of-the-art results, surpassing the strongest baseline by up to **3.5%** Recall@5, and outperforming the best Large Embedding Model in downstream QA by up to **14.1%** F1 score.

II. Related Work

A. Retrieval-Augmented Generation and Embedding Models

Standard RAG systems rely heavily on the quality of the dense retriever. Early approaches utilized sparse methods like BM25 or dual-encoders like Contriever and GTR [14]–[16] to match queries to passages. While effective for simple lookups, these models struggle with the semantic gap in complex reasoning tasks. Recent advancements in "Large Embedding Models", such as NV-Embed-v2 and GritLM [17], [18], leverage LLMs to generate text embeddings, significantly improving representation quality. However, even with SOTA embeddings, standard RAG lacks a mechanism to perform multi-hop reasoning over disjoint passages, as it relies on single-step semantic similarity rather than structured traversal.

B. Iterative Retrieval and Reasoning

To address the limitations of single-step retrieval, methods like IRCoT [19] interleave Chain-of-Thought (CoT) reasoning with retrieval. By using the LLM to generate reasoning steps that guide subsequent queries, IRCoT dynamically expands the context for multi-hop questions. However, this iterative "retrieve-and-read" paradigm incurs high computational latency, as it requires repeated LLM inference calls for every reasoning step and need models supporting long contexts to maintain history. Furthermore, its performance is strictly bound by the LLM's inherent reasoning capabilities. In contrast, our framework iteratively refines the PPR seeds to control the retrieval, but performs a *single* passage-level retrieval after

PPR, avoiding the cost of multiple read-retrieve cycles while preserving the benefits of query decomposition.

C. Structure-Enhanced RAG

To enable multi-hop reasoning without the overhead of iterative reading, recent works integrate structural priors into the retrieval process.

Tree-Based Methods. RAPTOR [20] recursively clusters text chunks into a hierarchical tree, allowing retrieval at varying levels of abstraction. While effective for tasks requiring structured abstraction, it lacks explicit entity-level reasoning required for granular fact-checking.

Summarization-Based Methods. Approaches like GraphRAG [21] extend RAG by building explicit knowledge graphs and applying community detection to identify relevant subgraphs for multi-hop retrieval. Similarly, LightRAG [22] integrates graph indexing with dual-level retrieval for efficient entity-relationship lookup. However, GraphRAG can be computationally expensive, and static in its hierarchical structures. Nevertheless, both methods struggle with flexibly integrating widely separated evidence or adaptively compose reasoning chains for particularly complex factoid QA.

PPR-Based Methods. HippoRAG [11] and its extension HippoRAG2 [12] are the most relevant to our work. They treat retrieval as a graph traversal problem, using PPR to propagate signals across the KG. HippoRAG seeds PPR by linking query entities to graph nodes, while HippoRAG2 matches the query embedding against graph triples, using a large-scale LLM to filter triples according to relevance. While powerful, both rely on a single, global representation for PPR initialization, limiting their ability to model sequential dependencies. In contrast, TraceRAG decomposes complex queries and employs efficient **structure-guided neural retrieval** for dependency-aware graph seeding, combining the precision of **sequential resolution** with the speed of graph propagation.

III. Methodology

We model the Knowledge Graph (KG) as an *undirected* graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ contains both entity nodes $\mathcal{V}_E$ and passage nodes $\mathcal{V}_D$, and $\mathcal{E}$ are the edges. Treating edges as undirected is crucial for multi-hop reasoning, allowing the retrieval process to traverse relationships in reverse (e.g., *starred_in* $\leftrightarrow$ *cast_member*) without explicit inverse labeling. Our goal is to retrieve a set of relevant passages $\mathcal{D}^* \subset \mathcal{V}_D$ given a complex multi-hop query Q.

A. Schemaless Graph Construction

Following the protocol of HippoRAG2 [12], we construct $\mathcal{G}$ via an OpenIE process tailored for retrieval.

Indexing Phase. We use an instruction-tuned LLM to extract named entities and factual triples (h, r, t) from each passage in the corpus, where h is a head entity, r is a relation, and t is a tail entity. This results in a "schemaless" graph where edge labels are open-text descriptions, preserving the nuance of the source text.

Graph Connectivity. To enable multi-hop reasoning across documents, we augment $\mathcal{G}$ with two types of edges: (1) *Synonym Edges*, linking entity nodes with cosine similarity $> \tau$ (computed via the retrieval encoder), and (2) *Passage-Context Edges*, linking every passage node d to the entities extracted from it.

B. Sequential Query Resolution

Unlike standard approaches that embed Q into a single global vector, we employ a *sequential resolution* strategy to identify the seed nodes relevant to graph traversal. Figure 1 presents an example of the procedure.

Step 1: Decomposition. We first decompose Q into a sequence of atomic sub-questions $\mathcal{S} = \{q_1, \ldots, q_m\}$ using an LLM. Dependencies are explicitly modeled via placeholders; if q_i relies on the answer to q_k ($k < i$), it contains a symbolic token η_k.

Step 2: Sequential Resolution and Verification. We process each sub-question q_i sequentially. For the i-th step:

1) **Binding:** We resolve the sub-query into $\hat{q}_i$ by replacing any placeholder η_k with the answer entity e_k^* identified in the previous iterations: $\hat{q}_i = \text{Replace}(q_i, \eta_k, e_k^*)$.
2) **Structure-Guided Neural Retrieval:** We identify relevant triples using a cascade filter. First, we retrieve the top-K candidates based on cosine similarity with $\mathbf{e}_{\hat{q}_i}$. Second, we apply a Cross-Encoder Reranker $\mathcal{R}$ to verify factual relevance. We retain a set $\mathcal{T}_i^*$ containing at most N_{max} triples that satisfy a confidence threshold δ.
3) **Update Seeds & History:** The entities within $\mathcal{T}_i^*$ are added to the global seed set $\mathcal{V}_{seed}$, and their reranking scores are accumulated into the PPR weight vector $\mathbf{w}$. Finally, the tail entity of the highest-scoring triple is designated as the answer e_i^*, used to resolve the placeholders in the subsequent steps (q_{i+1}).

If placeholder resolution fails entirely, we default to dense retrieval using the original query embedding to ensure coverage.

C. Context-Aware Graph Search

Once the seed set $\mathcal{V}_{seed}$ is fully constructed from all sub-questions, we execute the search using the PPR algorithm. The restart vector $\mathbf{p}^{(0)}$ is initialized to distribute the probability mass across the verified evidence. For a node v:

$$\mathbf{p}_{v_{(0)}} \propto \begin{cases} \sum_{\hat{q} \in \mathcal{S}} \mathcal{R}(\hat{q}, \tau_v) & \text{if } v \in \mathcal{V}_{seed} \\ \beta \cdot \cos(\mathbf{e}_Q, \mathbf{e}_v) & \text{if } v \in \mathcal{V}_D \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

Here, entity seeds are weighted by their reranker confidence, and passage nodes are weighted by their semantic similarity to the query. We balance these structural and semantic signals via β. The PPR algorithm iterates $\mathbf{p}_{(t+1)} = (1-\alpha)\mathbf{A}\mathbf{p}_{(t)} + \alpha\mathbf{p}_{(0)}$ until convergence. The final retrieval consists of the passage nodes with the highest probability mass in $\mathbf{p}_{(\infty)}$.

IV. Experimental Setup

To evaluate TraceRAG, we design our experiments to answer two primary questions: (1) Does sequential query resolution effectively mitigate the semantic drift of the global matching approaches? (2) What is the specific contribution of

TABLE I
QA PERFORMANCE (F1 SCORES) ON RAG BENCHMARKS USING LLAMA-3.3-70B-INSTRUCT AS THE QA READER. THIS TABLE, AND THE FOLLOWING ONES, HIGHLIGHT THE **BEST** AND SECOND-BEST RESULTS.

	MuSiQue	HotPotQA	2Wiki	Avg
Standard Retrievers				
BM25	31.3	62.3	41.9	45.2
Contriever	28.8	63.4	51.2	47.8
GTR (T5-base)	34.6	62.8	52.8	50.1
Large Embedding Models				
GTE-Qwen2-7B-Instruct	40.9	71.0	60.0	57.3
GritLM-7B	44.8	73.3	60.6	59.6
NV-Embed-v2 (7B)	45.7	75.3	61.5	60.8
Structure-Enhanced RAG				
RAPTOR	28.9	69.5	52.1	50.2
GraphRAG	38.5	68.6	58.6	55.2
LightRAG	1.6	2.4	11.6	5.2
HippoRAG	35.1	63.5	71.8	56.8
HippoRAG2	48.6	**75.5**	71.0	65.0
TraceRAG	**50.9**	**75.5**	**75.6**	**67.4**

TABLE II
DATASET STATISTICS.

	MuSiQue	HotPotQA	2WikiMultihopQA
# queries	1,000	1,000	1,000
# passages	11,656	9,811	6,119

TABLE III
RETRIEVAL PERFORMANCE (RECALL@5) ON RAG BENCHMARKS. ALL STRUCTURED RAG METHODS USE LLAMA-3.3-70B-INSTRUCT AS THE BACKBONE LLM AND NV-EMBED-V2 AS THE RETRIEVER. GRAPHRAG AND LIGHTRAG ARE NOT CONSIDERED BECAUSE THEY DO NOT DIRECTLY PRODUCE PASSAGE RETRIEVAL RESULTS.

	MuSiQue	HotPotQA	2Wiki	Avg
Standard Retrievers				
BM25	43.5	74.8	65.3	61.2
Contriever	46.6	75.3	57.5	59.8
GTR (T5-base)	49.1	73.9	67.9	63.6
Large Embedding Models				
GTE-Qwen2-7B-Instruct	63.6	89.1	74.8	75.8
GritLM-7B	65.9	92.4	75.3	77.9
NV-Embed-v2 (7B)	69.7	94.5	73.9	79.4
Structure-Enhanced RAG				
RAPTOR	57.8	86.9	66.2	70.3
HippoRAG	53.2	77.3	90.4	72.9
HippoRAG2	74.7	96.3	90.4	87.1
TraceRAG	**78.2**	**96.4**	**92.7**	**89.1**

TABLE IV
IMPACT EVALUATION OF SEQUENTIAL RESOLUTION AND STRUCTURE-GUIDED NEURAL RETRIEVAL ON RECALL@5.

	MuSiQue	HotPotQA	2Wiki	Avg
TraceRAG	**78.2**	**96.4**	**92.7**	**89.1**
w/o Structure-Guided Neural Retrieval	68.4	92.4	86.6	82.4
w/o Sequential Resolution	70.4	95.5	81.6	82.5
Global Cosine Matching	73.0	95.4	90.0	86.1

the structure-guided retrieval components, and is the framework robust to variations in model scale and hyperparameters?

Baselines. We compare TraceRAG against a comprehensive set of methods, grouped into three distinct paradigms:

1) *Standard Retrievers:* We include the lexical baseline **BM25**, as well as **Contriever** and **GTR**, two established dual-encoder models that serve as standard dense retrieval benchmarks.
2) *Large Embedding Models:* To verify if increasing model scale compensates for the lack of explicit reasoning structure, we evaluate **NV-Embed-v2**, **GritLM-7B**, and **GTE-Qwen2-7B-Instruct**. These models currently hold top positions on the BEIR leaderboard [23].
3) *Structure-Enhanced RAG:* We benchmark against the current state-of-the-art in structured retrieval, including **RAPTOR**, **GraphRAG**, **LightRAG**, **HippoRAG**, and **HippoRAG2**.

Datasets. Evaluation is conducted using three challenging QA benchmarks that test multi-hop reasoning across disjoint passages: **MuSiQue**, **HotpotQA**, and **2WikiMultihopQA**. These datasets are characterized by queries that require aggregating evidence from multiple documents which often share no lexical overlap. Following the protocol of prior works [11], [12], we perform evaluations of all methods on the same randomly sampled subset of $1,000$ queries per dataset (table II).

Metrics. We evaluate retrieval quality using Recall@5, measuring the proportion of gold supporting passages in the top-5 retrieved results. This is critical for multi-hop RAG, where generation requires dense relevant context in a limited window. For downstream QA, we report the F1 score of the generated answers, following the standard evaluation protocols.

Implementation Details. We evaluate TraceRAG using LLaMA-3.3-70B-Instruct as the backbone for OpenIE, query decomposition, and QA. For dense retrieval and similarity computations, we use NV-Embed-v2. Regarding prompting strategies, we employ one-shot prompting for OpenIE and a ten-shot strategy for the decomposition module to ensure robust resolution. For graph hyperparameters, we follow the HippoRAG configuration [11], setting the synonym threshold to 0.8, the PPR damping factor $\alpha = 0.5$, and the passage seed weighting $\beta = 0.05$. For the Structure-Guided Neural Retrieval (Sec. III-B), we deploy the Qwen3-4B-Reranker [24]. The candidate pool K, triple cap N_{max}, and confidence threshold δ are set based on sensitivity analysis (Sec. V-D).

V. RESULTS AND ANALYSIS

In this section, we evaluate the empirical performance of our framework. We first present the main results for passage retrieval and QA, compared to SOTA baselines. Subsequently, we conduct an ablation study to isolate the impact of our sequential query resolution and structure-guided neural retrieval components. Finally, we analyze the robustness of TraceRAG across different hyperparameters and model backbones.

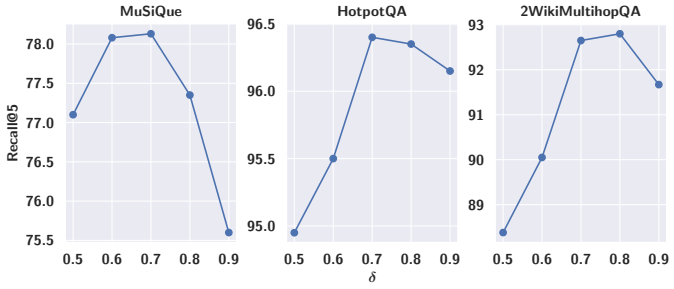

Fig. 3. Impact of reranker threshold δ on Recall@5 across RAG benchmarks. Optimal stability occurs in $[0.6, 0.8]$, balancing seed precision and evidence availability.

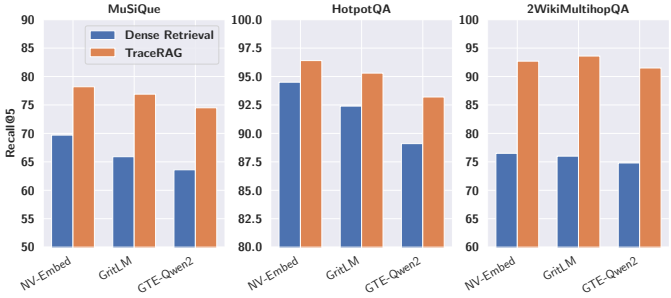

Fig. 4. Recall@5 comparison between Dense Retrieval and TraceRAG across three large embedding models (NV-Embed-v2, GritLM-7B, GTE-Qwen2-7B-Instruct). TraceRAG consistently improves retrieval.

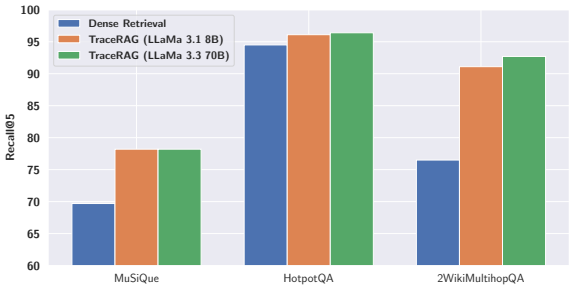

Fig. 5. TraceRAG performance with 8B vs 70B LLaMa OpenIE backbones; high retrieval recall is maintained with the smaller model.

A. Retrieval Performance

Table III summarizes retrieval performance (Recall@5). Our framework achieves SOTA results across all benchmarks (avg. **89.1%**), consistently outperforming both large-scale embeddings and structured baselines. Notably, TraceRAG exceeds NV-Embed-v2 by nearly **10%** on average (up to **+18.8%** on 2Wiki), demonstrating that model scale cannot substitute for structural reasoning in disjoint contexts. Furthermore, TraceRAG surpasses the previous best, **HippoRAG2**, on the complex MuSiQue dataset by **+3.5%**. This validates that the *sequential query resolution* generates optimized PPR seeds compared to global matching, effectively mitigating semantic drift.

B. Question Answering Performance

Table I reports the downstream QA performance (F1 scores). Consistent with retrieval results, TraceRAG achieves the highest average F1 score of **67.4%**, surpassing the strongest baseline, HippoRAG2, by **2.4%**. The performance gains are most relevant on the datasets requiring rigorous multi-hop synthesis, namely MuSiQue (**+2.3%**) and 2WikiMultihopQA (**+4.6%**). This indicates that our *sequential query resolution* module ensures that the generator is fed with a coherent chain of evidence, rather than a bag of noisy related facts.

C. Ablation Study

To isolate the contribution of our key components, we compare the full TraceRAG framework against three variants. Table IV confirms the critical synergy between resolution and retrieval. Removing decomposition (*w/o Sequential Resolution*) causes sharp drops (e.g., **-11.1%** on 2WikiMultihopQA), often underperforming simple baselines. This stems from a "Granularity Mismatch": complex queries fail to semantically align with atomic triples, causing the reranker to reject valid intermediate facts. Conversely, removing the reranker (*w/o Structure-Guided Neural Retrieval*) leads to severe degradation (e.g., **-9.8%** on MuSiQue). Without precise filtering, the expanded search surface from decomposition results in error cascading, polluting the PPR seeds. Thus, both components are necessary: Resolution provides the traversal *path*, while Retrieval ensures step-wise *precision*.

D. Sensitivity and Robustness Analysis

Sensitivity to the Hyperparameters. We analyze the impact of the reranker confidence threshold δ on the retrieval performance. Figure 3 shows that retrieval performance follows an inverted U-shape. Lower thresholds ($\delta \leq 0.5$) introduce noise that distracts the PPR, while overly strict thresholds ($\delta \geq 0.9$) filters out valid intermediate steps, breaking the multi-hop chain. Performance is robust in the interval $[0.6, 0.8]$, leading us to set $\delta = 0.7$. We find that variations in the candidate pool size K and triple cap $N_{\max}$ have negligible impact on Recall@5. As δ typically selects only 1-2 high-confidence triples, the system remains largely unaffected by these upper limits. Accordingly, we set $K = 10$ and $N_{\max} = 3$.

Sensitivity to the Embedding Model. Figure 4 demonstrates that TraceRAG consistently outperforms Dense Retrieval across three backbones (NV-Embed-v2, GritLM-7B, GTE-Qwen2-7B-Instruct). Notably, it improves GTE performance by over **+15%** on 2WikiMultihopQA. This confirms that our structural resolution mechanism induces significant performance gains independent of the encoder's quality.

Sensitivity to the OpenIE Backbone. Figure 5 reveals negligible performance gaps between graphs constructed by smaller models (**LLaMA-3.1-8B-Instruct**) versus larger ones (**LLaMA-3.3-70B-Instruct**). On MuSiQue, the 8B model matches the 70B results almost exactly. This proves the TraceRAG's robustness to extraction noise, as the iterative resolution mechanism can effectively leverage even imperfect graph structures, enabling effective deployment with lower-cost, open-weights models.

VI. Conclusion

In this paper, we introduced **TraceRAG**, a novel retrieval framework that addresses the limitations of global semantic matching in multi-hop Question Answering. By shifting the paradigm from "one-shot" embedding to **sequential query resolution**, TraceRAG explicitly models the sequential dependencies inherent in complex queries. We demonstrated that decomposing questions and verifying intermediate steps with a **structure-guided neural retrieval** mechanism generates highly precise seeds for graph traversal, effectively mitigating the semantic drift affecting existing approaches.

Empirical results over three benchmarks validate TraceRAG's state-of-the-art performance, outperforming strong baselines like HippoRAG2 and large embedding models. Crucially, our robustness analysis reveals that these gains are achievable even when using smaller, open-weights models for graph construction, democratizing access to high-performance structured retrieval. However, we acknowledge that our framework relies on the quality of the initial query decomposition; as with all multi-step pipelines, errors in generating sub-questions can propagate through the sequential resolution chain. Future work will explore extending this resolution mechanism to dynamic knowledge graphs, allowing the system not just to retrieve, but also to update the graph structure during the reasoning process.

References

[1] P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Küttler, M. Lewis, W.-t. Yih, T. Rocktäschel, S. Riedel, and D. Kiela, "Retrieval-augmented generation for knowledge-intensive nlp tasks," in *Proceedings of the 34th International Conference on Neural Information Processing Systems*, ser. NIPS '20, 2020.

[2] G. Lombardo, G. Trimigno, M. Pellegrino, and S. Cagnoni, "Language models fine-tuning for automatic format reconstruction of sec financial filings," *IEEE Access*, vol. 12, pp. 31 249–31 261, 2024.

[3] W. Liu, J. Chen, K. Ji, L. Zhou, W. Chen, and B. Wang, "RAG-instruct: Boosting LLMs with diverse retrieval-augmented instructions," in *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 2025. [Online]. Available: https://aclanthology.org/2025.emnlp-main.192/

[4] R. Cohen, E. Biran, O. Yoran, A. Globerson, and M. Geva, "Evaluating the ripple effects of knowledge editing in language models," *Transactions of the Association for Computational Linguistics*, vol. 12, 2024. [Online]. Available: https://aclanthology.org/2024.tacl-1.16/

[5] J.-C. Gu, H.-X. Xu, J.-Y. Ma, P. Lu, Z.-H. Ling, K.-W. Chang, and N. Peng, "Model editing harms general abilities of large language models: Regularization to the rescue," in *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 2024. [Online]. Available: https://aclanthology.org/2024.emnlp-main.934/

[6] Z. Zhong, Z. Wu, C. Manning, C. Potts, and D. Chen, "MQuAKE: Assessing knowledge editing in language models via multi-hop questions," in *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 2023. [Online]. Available: https://aclanthology.org/2023.emnlp-main.971/

[7] J. Xie, K. Zhang, J. Chen, R. Lou, and Y. Su, "Adaptive chameleon or stubborn sloth: Revealing the behavior of large language models in knowledge conflicts," in *The Twelfth International Conference on Learning Representations*, 2024. [Online]. Available: https://openreview.net/forum?id=auKAUJZMO6

[8] S. Siriwardhana, R. Weerasekera, E. Wen, T. Kaluarachchi, R. Rana, and S. Nanayakkara, "Improving the domain adaptation of retrieval augmented generation (RAG) models for open domain question answering," *Transactions of the Association for Computational Linguistics*, vol. 11, 2023. [Online]. Available: https://aclanthology.org/2023.tacl-1.1/

[9] G. Trimigno, G. Lombardo, M. Tomaiuolo, S. Cagnoni, and A. Poggi, "Llms in staging: An orchestrated llm workflow for structured augmentation with fact scoring," *Future Internet*, vol. 17, no. 12, 2025. [Online]. Available: https://www.mdpi.com/1999-5903/17/12/535

[10] R. Xu, H. Liu, S. Nag, Z. Dai, Y. Xie, X. Tang, C. Luo, Y. Li, J. C. Ho, C. Yang, and Q. He, "SimRAG: Self-improving retrieval-augmented generation for adapting large language models to specialized domains," in *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2025. [Online]. Available: https://aclanthology.org/2025.naacl-long.575/

[11] B. J. Gutiérrez, Y. Shu, Y. Gu, M. Yasunaga, and Y. Su, "Hipporag: neurobiologically inspired long-term memory for large language models," in *Proceedings of the 38th International Conference on Neural Information Processing Systems*, ser. NIPS '24, 2024.

[12] B. J. Gutiérrez, Y. Shu, W. Qi, S. Zhou, and Y. Su, "From RAG to memory: Non-parametric continual learning for large language models," in *Proceedings of the 42nd International Conference on Machine Learning*, ser. Proceedings of Machine Learning Research, vol. 267. PMLR, 2025. [Online]. Available: https://proceedings.mlr.press/v267/gutierrez25a.html

[13] T. H. Haveliwala, "Topic-sensitive pagerank," in *Proceedings of the 11th International Conference on World Wide Web*, ser. WWW '02, 2002. [Online]. Available: https://doi.org/10.1145/511446.511513

[14] S. E. Robertson and S. Walker, "Some simple effective approximations to the 2-poisson model for probabilistic weighted retrieval," in *Proceedings of the 17th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, ser. SIGIR '94.

[15] G. Izacard, M. Caron, L. Hosseini, S. Riedel, P. Bojanowski, A. Joulin, and E. Grave, "Towards unsupervised dense information retrieval with contrastive learning," *CoRR*, vol. abs/2112.09118, 2021. [Online]. Available: https://arxiv.org/abs/2112.09118

[16] R. Google and Apple, "Large dual encoders are generalizable retrievers," in *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 2022. [Online]. Available: https://aclanthology.org/2022.emnlp-main.669/

[17] T. NVIDIA, "Nv-embed: Improved techniques for training llms as generalist embedding models," 2025. [Online]. Available: https://arxiv.org/abs/2405.17428

[18] N. Muennighoff, H. Su, L. Wang, N. Yang, F. Wei, T. Yu, A. Singh, and D. Kiela, "Generative representational instruction tuning." in *ICLR*. OpenReview.net, 2025. [Online]. Available: http://dblp.uni-trier.de/db/conf/iclr/iclr2025.htmlMuennighoffS00W25

[19] H. Trivedi, N. Balasubramanian, T. Khot, and A. Sabharwal, "Interleaving retrieval with chain-of-thought reasoning for knowledge-intensive multi-step questions," in *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics*. Toronto, Canada: Association for Computational Linguistics, 2023. [Online]. Available: https://aclanthology.org/2023.acl-long.557/

[20] P. Sarthi, S. Abdullah, A. Tuli, S. Khanna, A. Goldie, and C. D. Manning, "RAPTOR: recursive abstractive processing for tree-organized retrieval," in *The Twelfth International Conference on Learning Representations, ICLR 2024, Vienna, Austria, May 7-11, 2024*. OpenReview.net, 2024. [Online]. Available: https://openreview.net/forum?id=GN921JHCRw

[21] T. Microsoft, "From local to global: A graph rag approach to query-focused summarization," 2025. [Online]. Available: https://arxiv.org/abs/2404.16130

[22] Z. Guo, L. Xia, Y. Yu, T. Ao, and C. Huang, "LightRAG: Simple and fast retrieval-augmented generation," in *Findings of the Association for Computational Linguistics: EMNLP 2025*. Association for Computational Linguistics, 2025. [Online]. Available: https://aclanthology.org/2025.findings-emnlp.568/

[23] N. Thakur, N. Reimers, A. Rücklé, A. Srivastava, and I. Gurevych, "BEIR: A heterogeneous benchmark for zero-shot evaluation of information retrieval models," in *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*, 2021. [Online]. Available: https://openreview.net/forum?id=wCu6T5xFjeJ

[24] T. Qwen, "Qwen3 embedding: Advancing text embedding and reranking through foundation models," *arXiv preprint arXiv:2506.05176*, 2025.