SemHippo: Improving GraphRAG with a Semantic Backbone

Anonymous ACL submission

Abstract

Retrieval-Augmented Generation (RAG) is widely used to inject external knowledge into large language models, but it struggles with multi-hop reasoning and cannot surface information that is not explicitly stated in the corpus. GraphRAG systems improve multi-hop performance by introducing associative or hierarchical graph structures, yet they still lack the semantic machinery required for formal inference and aggregation. We present SemHippo, a GraphRAG framework that augments retrieval with a *semantic backbone*: an automatically induced ontology and an ontology-compliant knowledge graph that can be queried directly via SPARQL. At inference time, an LLM agent issues SPARQL queries against the knowledge graph while a parallel passage-retrieval pipeline returns supporting text; the two evidence streams are fused before answer generation. Instantiated on top of HippoRAG 2, SemHippo consistently outperforms both HippoRAG 2 and RAG on MuSiQue and the GraphRAG-Benchmark, recovering 97.18% of a full-context oracle's accuracy on 2-hop MuSiQue. Its margin over HippoRAG 2 widens from 6 to 8 points as the evaluation broadens from 2-hop to a mix of 2-, 3-, and 4-hop questions, suggesting that explicit symbolic structure becomes increasingly valuable as reasoning depth grows. Beyond raw accuracy, the semantic backbone is designed to support exact aggregation and ontology-driven inference, capabilities we discuss but leave to future evaluation. that are structurally out of reach for retrieval-only systems. [1]

1 Introduction

Retrieval-Augmented Generation (RAG) (Gao et al., 2024; Lewis et al., 2021) has become one of the most popular approaches for injecting new information into large language models (LLMs), largely because it requires neither labeled training data nor parameter updates. Despite these advantages, RAG has two key limitations. First, because it relies on simple semantic similarity matching, it struggles with complex multi-hop reasoning tasks (Gutiérrez et al., 2025b). Second, it can only surface information explicitly present in the corpus; it cannot derive new knowledge from the underlying data.

To address the multi-hop limitation, GraphRAG systems (Edge et al., 2025a) have emerged and shown consistent improvements over standard RAG on reasoning benchmarks (Gutiérrez et al., 2025a,b). However, the most prominent systems rely on either associative (Gutiérrez et al., 2025a,b) or hierarchical (Edge et al., 2025a; Sarthi et al., 2024) graph structures, none of which possess the semantic machinery needed to infer new knowledge; they retrieve information but cannot derive it.

We propose a GraphRAG system that addresses both limitations: in addition to retrieving information, it can infer new knowledge that is not explicitly stated in the corpus. Our approach automatically constructs an ontology together with a knowledge graph (KG), enabling semantic inference through an agent that queries the KG via SPARQL. A KG is a graph-structured data model paired with an ontology that defines entity types and the relations among them. The ontology supports automatic inference through formal relation properties: for example, if the *"located in"* relation is declared transitive, and the KG contains the facts *"The Eiffel Tower is located in Paris"* and *"Paris is located in Europe,"* the system can automatically infer that the Eiffel Tower is located in Europe.

A KG further enables aggregation queries that are effectively impossible for purely retrieval-based systems. Consider the question *"Which drug affects the most symptoms related to ALS?"* A standard RAG system has no principled way to determine

[1] Code is included in the supplementary materials; a public repository link will be released upon acceptance.

how many documents to retrieve: too few risks missing relevant evidence, while too many risks exceeding the model's context window. Even when all relevant documents fit, the system must rely on the LLM to count occurrences correctly. The same question, expressed as a short SPARQL query, returns a single exact number.

We were unable to find existing RAG benchmarks that require aggregation, likely because such queries are intractable for both RAG and current GraphRAG systems. We therefore evaluate our system on standard reasoning and multi-hop benchmarks. We integrate our semantic backbone with HippoRAG 2 (Gutiérrez et al., 2025b), the strongest published GraphRAG system on our chosen benchmarks, and compare against both HippoRAG 2 and RAG as a non-graph baseline. Our contributions are as follows:

- We propose SemHippo, a GraphRAG framework that augments retrieval with an automatically induced ontology and ontology-compliant knowledge graph queryable via SPARQL.
- We introduce a two-phase ontology induction procedure (relation ranking followed by incremental schema construction with validator feedback) and an LLM-agent-based KG retrieval pipeline primed by a triple vector store.
- We show that SemHippo consistently outperforms HippoRAG 2 and RAG on MuSiQue and GraphRAG-Benchmark, with the margin over HippoRAG 2 growing as reasoning depth increases.
- We provide an ablation study isolating the contributions of priming and of HippoRAG 2's document retrieval, and an error analysis identifying label-noise patterns in MuSiQue.

2 Related Work

A growing body of research on GraphRAG aims to address the shortcomings of standard RAG when applied to large and complex reasoning tasks, primarily by incorporating structured knowledge representations and more sophisticated retrieval mechanisms. RAPTOR (Sarthi et al., 2024) and Microsoft's GraphRAG (Edge et al., 2025a), for instance, both rely on hierarchical clustering: the former builds a recursive tree with summaries at each level, while the latter applies community detection paired with LLM-produced synopses, both enabling coarse-to-fine retrieval and broad, high-coverage answers. Other approaches, such as GFM-RAG (Luo et al., 2025), G-Retriever (He et al., 2024), and LightRAG (Guo et al., 2025) pair graph neural encoders with tailored retrieval objectives.

Drawing inspiration from hippocampal memory mechanisms, HippoRAG (Gutiérrez et al., 2025a) and HippoRAG 2 (Gutiérrez et al., 2025b) use Personalized PageRank to perform multi-hop retrieval in a single step on an associative graph, attaining state-of-the-art results and efficiency on both path-following and path-finding QA benchmarks. DALK (Li et al., 2024), OG-RAG (Sharma et al., 2024), and KGP (Wang et al., 2023) take a different route by dynamically constructing knowledge graphs and employing traversal agents, where LLMs assemble domain-specific graphs and apply self-aware retrieval strategies to inject structural signals but use graph traversal instead of SPARQL querying. Xiao et al. (2025) compare RAPTOR, KGP, LightRAG, GraphRAG (Edge et al., 2025a), G-Retriever, HippoRAG, DALK, ToG (Sun et al., 2024) and GFM-RAG across five tasks and ranked HippoRAG as second best after RAPTOR. In HippoRAG 2 (Gutiérrez et al., 2025b), HippoRAG 2 is compared to LightRAG, GraphRAG, and RAPTOR across seven datasets and find that HippoRAG 2 is the best system. Xiang et al. (2026) compared HippoRAG, HippoRAG 2, GraphRAG (Edge et al., 2025a), LightRAG, RAPTOR, Lazy-GraphRAG (Edge et al., 2025b), and Fast-GraphRAG(circlemind-ai, 2024) across two domains and four tasks and ranked HippoRAG 2 as the best system. Because HippoRAG 2 performs best across the benchmarks, we apply our system to it, although the system can be applied to any RAG or GraphRAG system.

Prior work has shown that LLMs are effective at writing SPARQL queries (Perevalov and Both, 2025; Emonet et al., 2025; Pan et al., 2025; Smeros et al., 2025). This further motivates the use of SPARQL in a GraphRAG approach.

3 Method

3.1 Background: HippoRAG 2

HippoRAG 2 (Gutiérrez et al., 2025b) is a neurobiologically inspired RAG framework that extends HippoRAG (Gutiérrez et al., 2025a) to better emulate human long-term memory. It couples an LLM

(acting as the neocortex) with a schemaless triple store serving as a hippocampal index and uses a retrieval encoder to bridge the two. The pipeline consists of two stages: *offline indexing* and *online retrieval*.

Offline indexing. Given a corpus, an LLM performs open information extraction on each passage to extract (subject, relation, object) triples, whose arguments become *phrase nodes* connected by *relation edges* in the schemaless KG. The retrieval encoder then adds *synonym edges* between phrase pairs whose embedding similarity exceeds a threshold, linking equivalent concepts across passages. Each passage is also inserted as a *passage node* and connected to its constituent phrases via *context edges* labeled "*contains*". This *dense-sparse integration* unifies sparse conceptual representations (phrases) with dense contextual representations (passages) in a single graph, mitigating the context loss of the entity-centric design of the original HippoRAG.

Online retrieval. Given a query q, HippoRAG 2 adopts a *query-to-triple* linking strategy: the encoder scores all KG triples against q and returns the top-k candidates $\mathcal{T}$. A *recognition memory* step then prompts an LLM to filter $\mathcal{T}$ down to a relevant subset $\mathcal{T}' \subseteq \mathcal{T}$, mirroring the recall/recognition duality of human memory. The phrase nodes appearing in $\mathcal{T}'$ are selected as *seed nodes*, with reset probabilities proportional to their ranking scores; all passage nodes additionally receive a reset mass proportional to their embedding similarity with q, scaled by a weight factor w that balances phrase-driven and passage-driven activation. Personalized PageRank (PPR) (Haveliwala, 2002) is then run over the KG from these seeds, and passages are re-ranked by their resulting PPR scores. The top-ranked passages are passed to the LLM reader for the final QA step. If no triples survive filtering, the system falls back to dense passage retrieval.

3.2 SemHippo

SemHippo augments HippoRAG 2 with a *semantic backbone*: an ontology-aligned knowledge graph (KG) that can be queried directly via SPARQL. Like HippoRAG 2, the system operates in two stages: *offline indexing* and *online retrieval*.

3.2.1 Offline indexing

Offline indexing begins by running HippoRAG 2's standard indexing procedure and then leverages the resulting set of extracted relations to induce an ontology, which is in turn used to construct an ontology-compliant KG.

Ontology creation. Ontology generation proceeds in two phases.

Relation Ranking. All extracted relations are partitioned into batches, and an LLM is prompted to order each batch by placing relations that should appear higher in the ontology first. For instance, *hasParent* should precede *hasFamilyMember*, since the former is a sub-property of the latter. The per-batch orderings are then combined to produce a global ordering: given k batches $\{s_{i,1}, s_{i,2}, \ldots, s_{i,n}\}_{i=1}^{k}$, the merged sequence is

$$s_{1,1}, s_{2,1}, \ldots, s_{k,1},\ s_{1,2}, s_{2,2}, \ldots, s_{k,2},\ \ldots,\ s_{1,n}, s_{2,n}, \ldots, s_{k',n}.$$

i.e., the first element of every batch, followed by the second element of every batch, and so on.

Ontology Schema Creation. The ordered relations are then presented to an LLM in batches. At each step, the model receives the ontology constructed so far (initially empty) together with the current batch of relations, and decides which new classes and properties, if any, should be introduced. Every proposed addition is checked by an ontology validation tool; if syntactic or structural issues are detected, the LLM is prompted with the validator's feedback and asked to repair them.

Knowledge graph construction. Once the ontology is fixed, the KG is built by iterating over the corpus. For each document, an LLM agent receives the ontology together with the document text and is prompted to extract triples that conform to the ontology, using structured outputs.

To support provenance, every triple and every newly introduced entity is paired with a *sourceText* relation pointing back to its originating document. For example, a triple $\langle \textit{John}, \textit{knows}, \textit{Jane} \rangle$ is accompanied by $\langle \langle \textit{John}, \textit{knows}, \textit{Jane} \rangle, \textit{sourceText}, \textit{DocID} \rangle$, where *DocID* identifies the source passage. This allows downstream queries such as "*How did John meet Jane?*" to retrieve the original text, e.g., a passage stating they met in a bar by following the *sourceText* link.

The extracted triples are translated into SPARQL `INSERT` queries automatically. Any syntax errors encountered during insertion are forwarded to a

repair agent, which is tasked with producing a corrected query.

Another advantage of using HippoRAG 2 is that our system requires a relation extraction step and reuses the triple-extraction step from HippoRAG 2. A dedicated relation-extraction step would likely work better, since we only need relations rather than full triples, but reusing HippoRAG 2's extraction simplifies the pipeline.

KG Triple vector store. After the ontology-compliant KG has been built, all of its triples are indexed in a dedicated vector store. At query time, this store is used to *prime* the SPARQL agent (Section 3.2.2) with semantically relevant triples.

An overview of the offline indexing pipeline is shown in Figure 1 (Appendix A).

3.2.2 Online retrieval

Online retrieval consists of two parallel pipelines whose outputs are fused before answer generation.

Pipeline 1: HippoRAG 2 document retrieval. The first pipeline is the standard HippoRAG 2 retrieval procedure, which returns a ranked list of passages relevant to the query.

Pipeline 2: KG retrieval. The second pipeline is driven by an LLM agent equipped with two tools: (i) an *autocomplete* tool that discovers relevant entities in the KG via IRI autocomplete, and (ii) a *SPARQL* tool that executes arbitrary SPARQL queries against the KG. Given the query, we first perform a vector search over the triple store to retrieve the top semantically relevant triples; these serve as a priming context for the SPARQL agent, with the aim of finding the entities needed for the query directly and reducing the number of autocomplete calls the agent must issue. The agent then receives the ontology, the priming triples, and the two tools and is prompted to produce a report containing all evidence relevant to the query.

Fusion and answer generation. The outputs of the two pipelines, the retrieved passages and the SPARQL agent's reports, are concatenated and passed to a final LLM, which generates the answer.

An overview of the online retrieval pipeline is shown in Figure 2 (Appendix A).

3.2.3 Implementation details

Ontology validation is performed with `rdflib` (RDFLib Team, 2026). The SPARQL agent is implemented as a ReAct agent in LangGraph (LangChain Inc., 2026), and the autocomplete and SPARQL tools are adapted from the ttyg-langgraph library (Graphwise, 2026b). The KG is stored in Graphwise GraphDB (Graphwise, 2025). We also fixed a division-by-zero error in HippoRAG 2 that is present in the provided code.

4 Experiments

4.1 Experimental Setup

In every experiment, the LLM we use is OpenAI's GPT-5.1 (OpenAI, 2025); unless stated otherwise, the embedding model we use is OpenAI's text-embedding-3-large (OpenAI, 2024b), which is used in the RAG, HippoRAG, and SemHippo systems. For every system, we set the number of documents retrieved to 5, and for SemHippo, we set the number of priming triples to 5. For every experiment, the results shown are the average of 3 runs. The temperature for each model is set to 0. Any text longer than 20,480 characters is split recursively with an overlap of 2,560 characters. For GPT-5.1 models, we set the max context to 1,000,000 tokens and the max completion to 128,000; the relation batch size is 1,024 relations. For the GPT-4.1 model, we decreased the max completion to 32,000, and for the GPT-4.1-mini, we decreased the relations batch size to 32.

4.1.1 Datasets

The datasets we used were one of the most popular multi-hop datasets, MuSiQue (Trivedi et al., 2022), and the newer GraphRAG-Benchmark (Xiang et al., 2026). The GraphRAG-Benchmark contains two main domains, novel and medical, and for each domain, it contains four types of questions: fact retrieval, complex reasoning, contextual summarization, and creative generation. This gives us a total of 10 datasets across different domains and tasks. For each dataset, we took a 100-question subset to limit the experiment's cost. For the novel domain, we used 25 questions per subtask because some subtasks contained fewer than 100 examples, and we wanted equal representation for each subtask. For MuSiQue, we used two subsets: a random subset containing equal parts of two-, three-, and four-hop questions and a separate random subset containing only two-hop questions to evaluate how the system's performance changes when the number of hops exceeds two.

4.1.2 Metrics

For an evaluation metric, we used LLM as a judge. To be more specific, we used the recall from the answer correctness metric found in the graphrag-eval (Graphwise, 2026a) library. The way that the answer correctness metric works is that it accepts as input the question, the true answer, and the predicted answer, which are given to an LLM. The LLM extracts the claims made in the true answer and the claims in the predicted answer. After that, recall, precision, and F1 can be calculated based on the claims. We used only the recall on claims, as we argue that making additional claims that are not in the true answer does not mean that they are wrong. The benefit of this metric compared to a regular LLM as a judge metric is that it allows for judging partially correct answers, where if the true answer is two separate claims and the model answers one of them, we still get 50% accuracy on that question. We believe that this metric is superior to exact match and F1 on word tokens, as it measures the meaning of the answer and not whether the answer is formatted exactly as expected.

4.2 Results

We compare SemHippo against four reference points: a *no-context* baseline where the model answers without any retrieved passages; a *full-context* oracle that provides all gold passages, which we treat as an upper bound on what a retrieval-augmented system can achieve; regular RAG; and HippoRAG. Results on MuSiQue 2-hop questions are reported in Table 1, on the mixed 2-, 3-, and 4-hop MuSiQue subset in Table 2, and across the four task types of GraphRAG-Benchmark in Table 3. All numbers are averages over three runs unless stated otherwise. In addition to raw accuracy, we report *normalized accuracy*, which expresses each method's score as a fraction of the full-context oracle and isolates the fraction of achievable retrieval performance each method recovers. We do not report the normalized accuracy for the novel and medical datasets, as for those datasets, the gold context is often very long; for example, the medical subtasks contain a single context of 200,000 tokens, and it has been consistently shown that LLMs' performance significantly drops on long context tasks (Hsieh et al., 2024; Bai et al., 2024, 2025).

MuSiQue 2-hop. On 2-hop MuSiQue questions (Table 1), SemHippo reaches 69% accuracy, an 11-point improvement over RAG (58%) and a 6-point improvement over HippoRAG 2 (63%). In normalized terms, SemHippo recovers 97.18% of the full-context upper bound compared to 88.73% for HippoRAG 2 and 81.69% for RAG. The remaining gap to the oracle is only about three points of normalized accuracy, suggesting that on 2-hop questions, retrieval is no longer the dominant source of error for SemHippo and that further gains will have to come from the reader rather than the retriever.

MuSiQue with longer reasoning chains. When the evaluation set is broadened to a mix of 2-, 3-, and 4-hop questions (Table 2), accuracy drops for every method, including the full-context oracle (67%). This indicates that part of the added difficulty is intrinsic to longer-chain reasoning rather than attributable to retrieval. Even so, SemHippo retains a clear margin, reaching 60% accuracy versus 52% for HippoRAG 2 and 51% for RAG, corresponding to 89.55%, 77.61%, and 76.12% normalized accuracy, respectively. Notably, the absolute gap between SemHippo and HippoRAG 2 widens from 6 points on the 2-hop set to 8 points here, while the gap between HippoRAG 2 and RAG shrinks from 5 points to a single point. This is consistent with our hypothesis that semantic-based retrieval generalizes better than entity-graph traversal as the number of hops grows: when more intermediate links are required, dense lexical or entity-level matches become increasingly sparse, whereas SemHippo's semantic associations remain informative.

GraphRAG-Benchmark. Table 3 reports per-task scores on the novel and medical domains. On the *novel* domain, SemHippo obtains the best average score (65.00), ahead of HippoRAG 2 (57.00), RAG (50.00), and the no-context baseline (52.00). The improvements are largest on *fact retrieval* (87.00 vs. 76.00 for HippoRAG 2) and *complex reasoning* (76.00 vs. 67.00), the two task types that most directly reward precise multi-hop retrieval. On *contextual summarization* and *creative generation*, all retrieval methods are closer together, and the no-context baseline remains competitive; we read this as evidence that these task types depend less on retrieving specific facts and more on the model's parametric knowledge and generation ability.

On the *medical* domain, SemHippo again achieves the highest average (73.00), with HippoRAG 2 a close second (71.00). The largest gain

Method	Accuracy	Acc. norm.
No context	52.00	73.24
Full context	**71.00**	**100.00**
RAG	58.00	81.69
HippoRAG 2	63.00	88.73
SemHippo	**69.00**	**97.18**

Table 1: Results on MuSiQue 2-hop questions. SemHippo outperforms both HippoRAG 2 and RAG. The dataset used is MuSiQue 2-hop questions.

Method	Accuracy	Acc. norm.
No context	37.00	55.22
Full context	**67.00**	**100.00**
RAG	51.00	76.12
HippoRAG 2	52.00	77.61
SemHippo	**60.00**	**89.55**

Table 2: Results on the mixed-hop MuSiQue subset. SemHippo outperforms both HippoRAG 2 and RAG. The dataset used is a MuSiQue random subset.

over HippoRAG 2 is on *contextual summarization* (71.00 vs. 61.00), where broader semantic associations appear to surface more relevant supporting context. On *complex reasoning*, HippoRAG 2 outperforms SemHippo (75.00 vs. 70.00).

5 Discussion

Across all benchmarks, SemHippo outperforms HippoRAG 2 and RAG on average, with the largest gains on multi-hop, fact-oriented questions and the smallest differences on open-ended generation tasks where retrieval contributes less. The MuSiQue results further indicate that SemHippo's advantage over HippoRAG 2 widens with the number of reasoning hops, supporting our central claim that semantic-association retrieval scales better with reasoning depth than purely entity-driven graph methods.

5.1 Ablation Study

We also evaluated SemHippo without using HippoRAG 2 document retrieval in the inference; instead, we used RAG retrieval to retrieve documents. The results shown in Table 4 demonstrate that using RAG document retrieval instead of HippoRAG 2 decreases the performance by only 1 percentage point, showing that the semantic backbone can find most of the relationships that HippoRAG's associative graph can find. Additionally, we checked the performance effect of the priming shown in Table 4, and we see that the priming gives a 4-point performance improvement.

We also tried different models on the novel and medical datasets to further test how model size affects the performance. The results shown in Table 5 demonstrate that the gap between the systems becomes smaller with weaker models. The systems keep their relative rankings on average, and there is no distinct pattern found.

5.2 RAG vs GraphRAG

Xiang et al. (2026) report that GraphRAG methods underperform RAG on several NLP tasks. On the same benchmarks, we observe the opposite ordering: both HippoRAG 2 and SemHippo outperform RAG on average. Several factors plausibly contribute to this discrepancy. First, Xiang et al. (2026) evaluates with GPT-4o-mini (OpenAI, 2024a), whereas our experiments use GPT-5.1 (OpenAI, 2025), stronger models may benefit disproportionately from structured retrieval. Second, our HippoRAG 2 numbers reflect a fix to a division-by-zero bug present in the released implementation, which may affect the ranking of methods. Third, we use a claim-level recall metric rather than the metric employed in prior work. We cannot isolate the contribution of each factor from our current experiments, but the gap suggests that conclusions about when GraphRAG helps may be sensitive to reader strength and implementation details, and we encourage future comparisons to control for these.

5.3 MuSiQue Error Analysis

To better understand where SemHippo's remaining errors come from, we manually inspected a sample of incorrectly answered MuSiQue questions.

A non-trivial fraction of the questions marked as incorrect by the automatic evaluator, on inspection, are either ill-posed or rely on facts that are never actually stated in the supporting passages. Because MuSiQue is constructed by composing single-hop questions, the composition can introduce assumptions that the underlying paragraphs do not support.

Consider the question, *"What record label does the spouse of Heart Like Mine's performer belong to?"*, whose gold answer is *Warner Bros. Records*. The intended reasoning chain is *Heart Like Mine* → Miranda Lambert → spouse (Blake Shelton) → Warner Bros. Records. However, none of the supporting paragraphs in the MuSiQue instance states that Blake Shelton is Miranda Lambert's spouse;

Method	Fact Retrieval	Complex Reasoning	Contextual Summarize	Creative Generation	Average
Novel Dataset					
No context	41.00	67.00	**56.00**	**42.00**	52.00
RAG	72.00	55.00	42.00	32.00	50.00
HippoRAG 2	76.00	67.00	53.00	33.00	57.00
SemHippo	**87.00**	**76.00**	**56.00**	**42.00**	**65.00**
Medical Dataset					
No context	73.00	66.00	58.00	50.00	62.00
RAG	87.00	64.00	59.00	58.00	67.00
HippoRAG 2	**90.00**	**75.00**	61.00	58.00	71.00
SemHippo	**90.00**	70.00	**71.00**	**59.00**	**73.00**

Table 3: SemHippo consistently outperforms both HippoRAG 2 and RAG. Combined results across both datasets, novel and medical.

Method	Accuracy
SemHippo	**69.00**
SemHippo + dense retrieval	68.00
SemHippo w/o priming	65.00

Table 4: SemHippo + dense retrieval is SemHippo where, instead of HippoRAG 2 retrieval at inference, RAG is used for document retrieval.

the spousal link is assumed by the question generator but never grounded in the retrieved context. Both the full-context oracle and SemHippo correctly report that the spouse relation is not specified, and SemHippo additionally surfaces the relevant record-label fact for Blake Shelton's album. The evaluator nevertheless marks both systems as wrong. This pattern, a multi-hop question whose middle hop is not actually supported by the gold paragraphs, appeared in several of the cases we inspected.

We also observed cases where the gold answer is a label rather than the entity the question literally asks about. For example, *"What is the administrative territorial entity that contains the location of Eric Marcus Municipal Airport PBS station?"* has the gold answer *KUAT-TV 6*, which is a PBS station, not an administrative territorial entity. Both the oracle and our system answer *Pima County, Arizona, United States*, which is a faithful reading of the question as written. Such cases penalize systems for producing a literally correct answer.

These observations are consistent with prior reports of label noise in older compositional multi-hop benchmarks (Trivedi et al., 2022) and suggest that MuSiQue also suffers from such issues, and scores can be interpreted as a lower bound on true system accuracy. They also imply that the gap between SemHippo and the full-context oracle in Tables 1 and 2 is a conservative estimate of the headroom that remains.

5.4 Dataset Contamination

We considered two other widely used multi-hop benchmarks, 2WikiMultiHopQA (Ho et al., 2020) and HotpotQA (Yang et al., 2018), but found them unsuitable for evaluating modern LLMs. As shown in Table 6, the no-context baseline substantially outperforms RAG on both datasets, by 10 points on 2WikiMultiHopQA and 30 points on HotpotQA. Since retrieval cannot help when the model already knows the answer and can be hurt by introducing distractors, gaps of this magnitude are most plausibly explained by training-data contamination. We therefore exclude both datasets from our main evaluation.

6 Conclusion

We introduced SemHippo, which augments retrieval-based QA systems with a *semantic backbone*: an automatically induced ontology and an ontology-compliant knowledge graph queryable via SPARQL. Instantiated on top of HippoRAG 2, SemHippo consistently outperforms both HippoRAG 2 and RAG across MuSiQue and GraphRAG-Benchmark, recovering 97.18% of the full-context upper bound on 2-hop MuSiQue. Notably, its margin over HippoRAG 2 widens from 6 to 8 points as the evaluation broadens from 2-hop to a mix of 2-, 3-, and 4-hop questions, suggesting that explicit symbolic structure becomes increasingly

Method	Model	Fact Retrieval	Complex Reasoning	Contextual Summarize	Creative Generation	Average
Novel Dataset						
RAG	GPT-5.1	72.00	55.00	42.00	32.00	50.00
HippoRAG 2	GPT-5.1	76.00	67.00	53.00	33.00	57.00
SemHippo	GPT-5.1	**87.00**	**76.00**	**56.00**	**42.00**	**65.00**
RAG	GPT-4.1	72.00	**71.00**	44.00	41.00	57.00
HippoRAG 2	GPT-4.1	**84.00**	67.00	50.00	38.00	59.75
SemHippo	GPT-4.1	81.00	70.00	**53.00**	**43.00**	**62.00**
RAG	GPT-4.1-mini	65.00	70.00	50.00	37.00	55.50
HippoRAG 2	GPT-4.1-mini	76.00	**72.00**	48.00	**43.00**	59.75
SemHippo	GPT-4.1-mini	**81.00**	70.00	**53.00**	**43.00**	**61.75**
Medical Dataset						
RAG	GPT-5.1	87.00	64.00	59.00	58.00	67.00
HippoRAG 2	GPT-5.1	**90.00**	**75.00**	61.00	58.00	71.00
SemHippo	GPT-5.1	**90.00**	70.00	**71.00**	**59.00**	**73.00**
RAG	GPT-4.1	84.00	74.00	71.00	**70.00**	74.75
HippoRAG 2	GPT-4.1	**92.00**	76.00	74.00	**70.00**	78.00
SemHippo	GPT-4.1	91.00	**77.00**	**77.00**	69.00	**78.50**
RAG	GPT-4.1-mini	85.00	72.00	67.00	**69.00**	73.25
HippoRAG 2	GPT-4.1-mini	84.00	**75.00**	71.00	**69.00**	74.75
SemHippo	GPT-4.1-mini	**89.00**	74.00	**73.00**	68.00	**76.00**

Table 5: Model comparison across methods on the novel and medical datasets. Within each model group, the best score per metric is bolded.

Method	2WikiMHQA	HotpotQA
No context	52.00	70.00
RAG	42.00	40.00

Table 6: Accuracy of the no-context baseline versus RAG with GPT-5.1. The no-context system outperforms RAG on both datasets, suggesting training-data contamination.

valuable as reasoning depth grows. Beyond raw accuracy, the semantic backbone enables query types that are structurally out of reach for retrieval-only systems, most notably exact aggregation and quantitative queries expressible in SPARQL, as well as inferences derivable from the ontology rather than stated explicitly in the source data. We are not aware of an existing benchmark in which such queries are posed against a document corpus rather than only a pre-built KG, and we view the construction of one as a natural next step toward evaluating the full capabilities of neuro-symbolic retrieval systems.

Ethical Considerations

The development of SemHippo, while aimed at improving the accuracy and reasoning capabilities of RAG systems, carries several ethical considerations:

Potential for Misinformation: Like all RAG systems, SemHippo relies on the quality of the underlying corpus. While the semantic backbone and SPARQL-based reasoning aim to increase the faithfulness of the model to the source text, the system may still propagate biases or inaccuracies present in the original documents. Furthermore, the use of an LLM to induce ontologies and extract triples introduces a risk of "structured hallucinations," where the knowledge graph itself may contain factually incorrect relations not present in the text.

Environmental Impact: SemHippo requires a more computationally intensive indexing phase than standard RAG, involving multiple LLM passes for ontology induction and triple extraction. Additionally, the online retrieval phase utilizes an LLM agent that may issue multiple tool calls (SPARQL and autocomplete), increasing the per-query carbon footprint and energy consumption compared to single-pass dense retrieval.

Transparency and Interpretability: SemHippo's intermediate SPARQL queries and retrieved reports may aid interpretability, though they do not guarantee faithfulness. By providing the SPARQL queries and retrieved reports alongside the final answer, the system offers a more transparent reasoning trace than end-to-end "black box" models. We encourage users to inspect these intermediate steps to verify the system's claims.

Dual-Use and Biased Knowledge Graphs: If deployed on sensitive domains (e.g., medical or legal), the automated induction of an ontology might inadvertently codify societal biases. We acknowl-

edge that the ontologies generated by the model are subject to the LLM's training data and may not represent a neutral or universal categorization of knowledge.

Limitations

A central capability of our system is its ability to answer aggregation and quantitative questions (e.g., *"How many. . . ?"*, *"What is the total. . . ?"*, *"List all X where Y"*), yet we do not evaluate it on a benchmark targeting such questions. To our knowledge, no suitable RAG benchmark currently exists, likely because prior RAG systems have been unable to handle these queries in the first place. A growing body of work demonstrates that LLMs can effectively generate SPARQL queries, as mentioned in Section 2. However, the associated datasets pair questions and answers with a predefined ontology, whereas our setting requires an ontology induced over a document collection, together with questions answerable from those documents. We leave the construction of such a benchmark to future work.

The performance gains we report come at the cost of increased offline indexing and online retrieval. The offline overhead can be largely amortized, or avoided entirely, when a suitable ontology and KG already exist. The online retrieval overhead, however, is intrinsic to the approach.

References

Yushi Bai, Xin Lv, Jiajie Zhang, Hongchang Lyu, Jiankai Tang, Zhidian Huang, Zhengxiao Du, Xiao Liu, Aohan Zeng, Lei Hou, Yuxiao Dong, Jie Tang, and Juanzi Li. 2024. Longbench: A bilingual, multitask benchmark for long context understanding. *Preprint*, arXiv:2308.14508.

Yushi Bai, Shangqing Tu, Jiajie Zhang, Hao Peng, Xiaozhi Wang, Xin Lv, Shulin Cao, Jiazheng Xu, Lei Hou, Yuxiao Dong, Jie Tang, and Juanzi Li. 2025. Longbench v2: Towards deeper understanding and reasoning on realistic long-context multitasks. *Preprint*, arXiv:2412.15204.

circlemind-ai. 2024. fast-graphrag: Rag that intelligently adapts to your use case, data, and queries. https://github.com/circlemind-ai/fast-graphrag. GitHub repository, accessed 2026-05-21.

Darren Edge, Ha Trinh, Newman Cheng, Joshua Bradley, Alex Chao, Apurva Mody, Steven Truitt, Dasha Metropolitansky, Robert Osazuwa Ness, and Jonathan Larson. 2025a. From local to global: A graph rag approach to query-focused summarization. *Preprint*, arXiv:2404.16130.

Darren Edge, Ha Trinh, Newman Cheng, Joshua Bradley, Alex Chao, Apurva Mody, Steven Truitt, Dasha Metropolitansky, Robert Osazuwa Ness, and Jonathan Larson. 2025b. From local to global: A graph rag approach to query-focused summarization. *Preprint*, arXiv:2404.16130.

Vincent Emonet, Jerven Bolleman, Severine Duvaud, Tarcisio Mendes de Farias, and Ana Claudia Sima. 2025. Llm-based sparql query generation from natural language over federated knowledge graphs. *Preprint*, arXiv:2410.06062.

Yunfan Gao, Yun Xiong, Xinyu Gao, Kangxiang Jia, Jinliu Pan, Yuxi Bi, Yi Dai, Jiawei Sun, Meng Wang, and Haofen Wang. 2024. Retrieval-augmented generation for large language models: A survey. *Preprint*, arXiv:2312.10997.

Graphwise. 2025. Graphwise GraphDB. RDF graph database with SPARQL and reasoning support.

Graphwise. 2026a. graphrag-eval: A Python Library for Evaluating Graph RAG Question-Answering Systems. Apache-2.0 License.

Graphwise. 2026b. TTYG: Talk to Your Graph – Natural Language Querying with GraphDB and LangChain Agents. Apache-2.0 License.

Zirui Guo, Lianghao Xia, Yanhua Yu, Tu Ao, and Chao Huang. 2025. Lightrag: Simple and fast retrieval-augmented generation. *Preprint*, arXiv:2410.05779.

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

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

Taher H. Haveliwala. 2002. Topic-sensitive pagerank. In *Proceedings of the 11th International Conference on World Wide Web*, WWW '02, page 517–526, New York, NY, USA. Association for Computing Machinery.

Xiaoxin He, Yijun Tian, Yifei Sun, Nitesh V. Chawla, Thomas Laurent, Yann LeCun, Xavier Bresson, and Bryan Hooi. 2024. G-retriever: Retrieval-augmented generation for textual graph understanding and question answering. *Preprint*, arXiv:2402.07630.

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

Cheng-Ping Hsieh, Simeng Sun, Samuel Kriman, Shantanu Acharya, Dima Rekesh, Fei Jia, Yang Zhang, and Boris Ginsburg. 2024. Ruler: What's the real context size of your long-context language models? *Preprint*, arXiv:2404.06654.

LangChain Inc. 2026. LangGraph: A Low-Level Orchestration Framework for Building Stateful, Multi-Actor Agents with LLMs. MIT License.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2021. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Preprint*, arXiv:2005.11401.

Dawei Li, Shu Yang, Zhen Tan, Jae Young Baik, Sukwon Yun, Joseph Lee, Aaron Chacko, Bojian Hou, Duy Duong-Tran, Ying Ding, Huan Liu, Li Shen, and Tianlong Chen. 2024. Dalk: Dynamic co-augmentation of llms and kg to answer alzheimer's disease questions with scientific literature. *Preprint*, arXiv:2405.04819.

Linhao Luo, Zicheng Zhao, Gholamreza Haffari, Dinh Phung, Chen Gong, and Shirui Pan. 2025. Gfm-rag: Graph foundation model for retrieval augmented generation. *Preprint*, arXiv:2502.01113.

OpenAI. 2024a. GPT-4o mini: Advancing cost-efficient intelligence. https://openai.com/index/gpt-4o-mini-advancing-cost-efficient-intelligence/. Accessed: 2026-05-12.

OpenAI. 2024b. New embedding models and API updates. https://openai.com/index/new-embedding-models-and-api-updates/. Accessed: 2026-05-12.

OpenAI. 2025. GPT-5.1. https://openai.com/index/gpt-5-1/. Accessed: 2026-05-12.

Xueli Pan, Victor de Boer, and Jacco van Ossenbruggen. 2025. Firesparql: A llm-based framework for sparql query generation over scholarly knowledge graphs. *Preprint*, arXiv:2508.10467.

Aleksandr Perevalov and Andreas Both. 2025. Text-to-sparql goes beyond english: Multilingual question answering over knowledge graphs through human-inspired reasoning. *Preprint*, arXiv:2507.16971.

RDFLib Team. 2026. RDFLib: A Python Library for Working with RDF.

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

Kartik Sharma, Peeyush Kumar, and Yunqing Li. 2024. Og-rag: Ontology-grounded retrieval-augmented generation for large language models. *Preprint*, arXiv:2412.15235.

Panayiotis Smeros, Vincent Emonet, Ruijie Wang, Ana-Claudia Sima, and Tarcisio Mendes de Farias. 2025. Sparql-llm: Real-time sparql query generation from natural language questions. *Preprint*, arXiv:2512.14277.

Jiashuo Sun, Chengjin Xu, Lumingyuan Tang, Saizhuo Wang, Chen Lin, Yeyun Gong, Lionel M. Ni, Heung-Yeung Shum, and Jian Guo. 2024. Think-on-graph: Deep and responsible reasoning of large language model on knowledge graph. *Preprint*, arXiv:2307.07697.

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

Yu Wang, Nedim Lipka, Ryan A. Rossi, Alexa Siu, Ruiyi Zhang, and Tyler Derr. 2023. Knowledge graph prompting for multi-document question answering. *Preprint*, arXiv:2308.11730.

Zhishang Xiang, Chuanjie Wu, Qinggang Zhang, Shengyuan Chen, Zijin Hong, Xiao Huang, and Jinsong Su. 2026. When to use graphs in rag: A comprehensive analysis for graph retrieval-augmented generation. *Preprint*, arXiv:2506.05690.

Yilin Xiao, Junnan Dong, Chuang Zhou, Su Dong, Qian wen Zhang, Di Yin, Xing Sun, and Xiao Huang. 2025. Graphrag-bench: Challenging domain-specific reasoning for evaluating graph retrieval-augmented generation. *Preprint*, arXiv:2506.02404.

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

A Pipeline Diagrams

An overview of the offline indexing pipeline is shown in Figure 1. An overview of the online retrieval pipeline is shown in Figure 2.

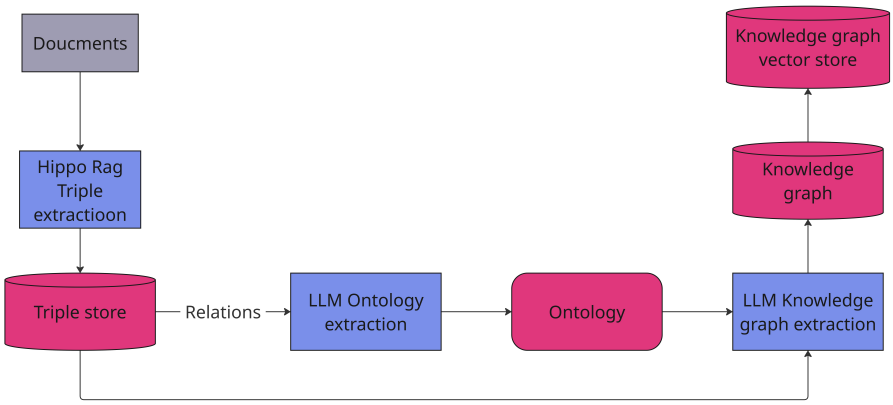

Figure 1: Offline indexing pipeline of SemHippo.

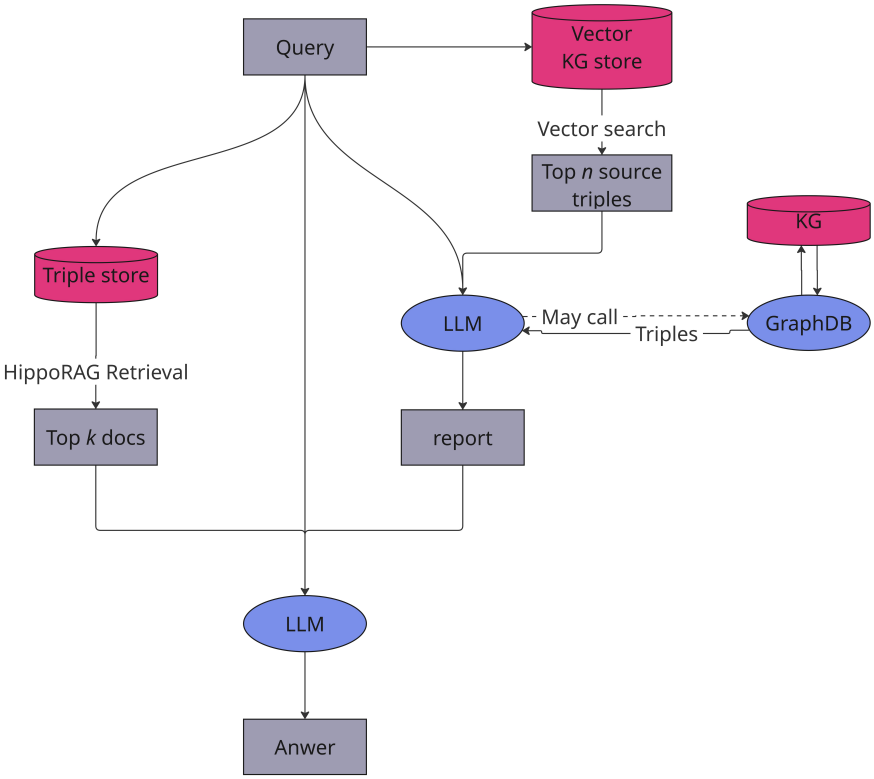

Figure 2: Online retrieval pipeline of SemHippo.