Rethinking Agentic RAG: Toward LLM-Driven Logical Retrieval Beyond Embeddings

Anonymous ACL submission

Abstract

Recent advances in RAG have shifted toward an agentic paradigm, where LLMs interact with retrieval systems over multiple turns and iteratively refine queries based on intermediate results. At the same time, LLMs have demonstrated a strong ability to construct structured queries that precisely express their information needs. However, contemporary RAG systems remain heavily focused on engineering complex retrieval backends, including dense, hybrid, and graph-based retrieval architectures. In this study, we argue that agentic RAG should delegate greater control to the LLM to steer the retrieval process, while relying on a lightweight retrieval interface that provides fine-grained control and faithfully executes the LLM's structured intent. Guided by this principle, we propose an agentic RAG framework that enables LLMs to formulate retrieval intents using logical expressions while simplifying the retrieval backend to an inverted-index-based system. Extensive experiments show that our framework matches a strong agentic hybrid baseline, while substantially reducing construction and serving cost. Moreover, we show that anchoring the retrieval process in logical queries substantially reduces hallucinations in generated responses.

1 Introduction

Retrieval-augmented generation (RAG) has become a fundamental framework for grounding the outputs of large language models (LLMs) in external knowledge (Lewis et al., 2020; Izacard and Grave, 2021). More recently, advancements in RAG have shifted toward an agentic paradigm, wherein LLMs interact with retrieval backends over multiple turns (Trivedi et al., 2023; Asai et al., 2024; Li et al., 2025a). This allows the model to progressively refine its queries to fulfill complex information needs until the required information is successfully retrieved or the model determines that the requested knowledge is unavailable. Consequently, the effectiveness of modern RAG systems no longer depends solely on the quality of a single query and its retrieved results, but on the quality of multi-turn interactions between LLMs and retrieval backends. In particular, successful agentic RAG requires LLMs to accurately express retrieval intents, retrieval backends to faithfully execute those intents, and LLMs to adaptively refine subsequent queries based on intermediate results.

Retrieval backends are commonly constructed using either graph-based or embedding-based approaches. Graph-based backends are typically motivated by the observation that complex queries often require decomposition into several successive sub-queries, particularly when the target information is scattered across disparate sources (Edge et al., 2024; Gutiérrez et al., 2024; Zhu et al., 2025). While graph-based retrieval remains valuable for highly complex tasks such as autonomous research and deep research scenarios, its advantages in many multi-hop question-answering scenarios are diminished in agentic RAG systems, where iterative multi-turn querying is already a native capability.

Embedding-based retrieval methods were primarily developed to retrieve information relevant to natural-language queries by computing similarity within a dense semantic embedding space (Karpukhin et al., 2020). Although embeddings are highly effective at capturing abstract semantic relationships, they may overlook critical lexical details, such as proper names or rare entities appearing in queries. To mitigate this issue, embedding-based retrieval is often augmented with sparse retrieval techniques, such as TF-IDF or other vector space models (Arivazhagan et al., 2023; Wang et al., 2024). However, these methods still face a fundamental challenge in multi-turn agentic RAG settings: the retrieved results are often relatively insensitive to slightly refined queries. Because refined queries tend to remain semantically similar to previous ones, the retrieved documents frequently exhibit substantial overlap with earlier retrieval re-

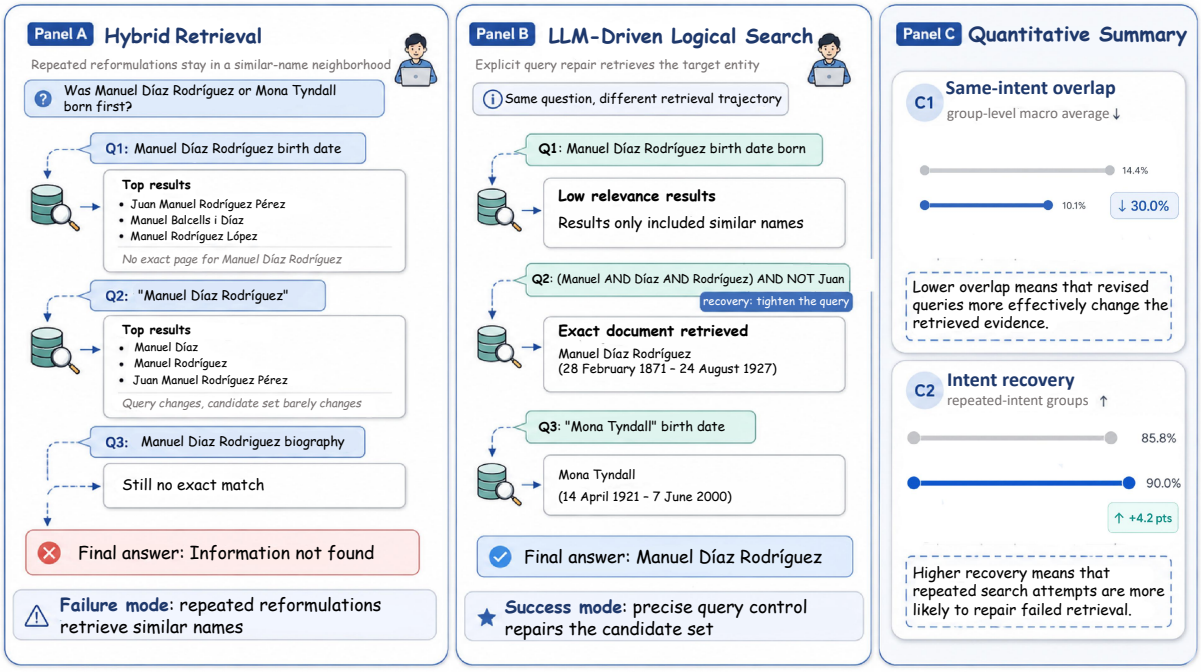

Figure 1: Motivating example and trajectory statistics for LLM-driven logical retrieval. Agentic Hybrid repeatedly retrieves similar-name distractors, while LogicalRAG repairs the search with explicit lexical constraints. The right panel reports same-intent result overlap and intent recovery across trajectories.

sults (Figure 1). This behavior not only reduces retrieval efficiency by repeatedly returning redundant content, but can also make it difficult for LLMs to repair failed searches through subsequent query refinement.

These observations motivate us to rethink the design of agentic RAG systems. We argue that greater control over the retrieval process should be delegated to the LLM during iterative multi-turn interactions. This paradigm is enabled by the rapidly improving ability of LLMs to formulate structured and precise queries that explicitly capture their retrieval intent. Correspondingly, retrieval backends should faithfully execute the search actions issued by the LLM, while exposing a high-control interface that lets the model broaden recall, tighten precision, exclude distractors, and adjust matching granularity. When the requested information is unavailable, such an interface should also make retrieval failure more explicit, rather than forcing approximate semantic matches that may encourage unsupported answers.

Guided by these design principles for agentic RAG systems, we enable LLMs to construct complex queries using logical operators and compositional constraints (Section 3). To support such logical query expressions, the underlying retrieval backend is implemented using an inverted-index-based architecture. Compared with embedding-based or graph-based retrieval backends, inverted-index systems are substantially easier to build and maintain. More importantly, they can faithfully execute explicit lexical constraints and provide a transparent feedback signal for subsequent query refinement. We found that combining logical query expressions with inverted-index retrieval achieved performance comparable to a strong agentic hybrid baseline while substantially reducing construction and serving cost. Furthermore, our experiments demonstrate that grounding the retrieval process in explicit logical queries improves abstention behavior and reduces unsupported answers when evidence is unavailable.

Our contributions are threefold:

- We propose a design paradigm for agentic RAG centered on LLM-driven retrieval control. This paradigm frames the retrieval process as an interface-control problem. It emphasizes intent-faithful execution and fine-grained control over the search space, enabling agents to deliberately broaden, constrain, and repair retrieval across multi-turn reasoning.
- We introduce LogicalRAG, an agentic RAG framework that executes LLM-formulated logical expressions via a lightweight inverted-index backend. It achieves performance com-

parable to complex hybrid systems while substantially reducing construction and serving cost, with fewer unsupported answers when evidence is unavailable.

- We identify an emerging capability threshold for LLM-driven logical retrieval. Model scaling experiments show that logical retrieval lags behind hybrid retrieval with weaker agents, but reaches parity with the strongest agent we evaluate. This suggests that high-control retrieval interfaces are becoming increasingly viable for modern LLM agents.

2 Related Work

2.1 Dense and Hybrid Retrieval in RAG

A widely adopted method is dense retrieval, which maps queries and documents into a continuous embedding space to measure semantic similarity. To further improve retrieval recall and precision, dense retrieval is often combined with sparse lexical matching methods, giving rise to hybrid retrieval approaches that jointly capture semantic and lexical relevance (Karpukhin et al., 2020; Gao et al., 2023; Arivazhagan et al., 2023). Although hybrid retrieval has become a popular choice for retrieval backends (Wang et al., 2024), its behavior can be difficult for LLMs to anticipate or control during multi-turn interactions. In particular, refined conversational queries often remain close to earlier queries in the shared embedding space, leading to substantial overlap in retrieved results. Such redundancy may confuse the LLM and make it more difficult to formulate effective follow-up queries.

2.2 Graph- and Tree-based Retrieval

Another line of research introduces explicit structure into the retrieval space. Graph-based systems such as GRAPHRAG and HIPPORAG organize documents around entities and relations, while tree- or page-based methods structure evidence hierarchically to support navigation and aggregation (Edge et al., 2024; Gutiérrez et al., 2024; Zhang et al., 2025). Such structured representations can facilitate multi-hop reasoning and global information synthesis. However, constructing and maintaining graph- or tree-based retrieval structures is often time-consuming and labor-intensive (Zhuang et al., 2025). These approaches are further complicated by persistent engineering challenges, including entity disambiguation, cascading errors in relation extraction, and the need to continuously update graph structures as the underlying corpus evolves.

2.3 Agentic Workflows

Recent RAG systems have shifted from single-step retrieval toward agentic workflows, in which the LLM iteratively plans, retrieves evidence, observes intermediate results, and revises subsequent actions. Prior work such as ReAct (Yao et al., 2022) and IRCoT (Trivedi et al., 2023) demonstrates that interleaving reasoning and retrieval enables models to recover from incomplete evidence and better decompose multi-hop questions. This shift also changes the role of the retriever: beyond returning relevant documents, it must provide feedback signals that the model can interpret and act upon during reasoning (Li et al., 2025b). Inspired by agent tools such as `grep`-style search in code environments and structure-aware document navigation (Anthropic, 2024; Zhang et al., 2025), LOGICALRAG maintains a simple yet reliable retrieval backend while exposing logical expressions and compositional constraints as an LLM-driven action space.

3 Method

We propose LOGICALRAG, an agentic RAG system built around an LLM-driven logical search interface. Bypassing architectural overhauls, LOGICALRAG exposes a retrieval action space through which the model can express and revise precise search intent. We first define this interface-level principle, then describe its realization with Boolean query logic, adjustable matching granularity, and BM25 ranking (Robertson and Zaragoza, 2009).

3.1 LLM-driven Retrieval

We formalize LLM-driven retrieval as an interface-level design principle for agentic RAG. The language model is responsible for deciding what evidence to seek, while the retriever is responsible for executing the model's stated search intent. Beyond standalone matching strength, this view prioritizes whether the interface lets the agent deliberately change the search space as its reasoning evolves.

Two properties are central to this interface. First, retrieval should be *intent-faithful*: when the model specifies an entity, phrase, term combination, or exclusion, the retriever should reflect that constraint directly in the candidate set. Second, the interface should provide sufficient *control* for query repair. As the model inspects intermediate evidence, it should be able to broaden recall, tighten precision,

remove distractors, or adjust matching granularity without relying on the backend to infer these changes implicitly.

This formulation separates the role of the retrieval interface from the complexity of the retrieval backend. A strong backend may improve standalone retrieval quality, but an agentic system also depends on whether the model can deliberately change what the retriever returns. LogicalRAG instantiates this principle with a lightweight lexical interface that exposes logical constraints and adjustable matching granularity while using BM25 only to rank matched candidates.

3.2 LogicalRAG: Controllable and Faithful Keyword Search

LogicalRAG realizes LLM-driven search with a lightweight logical retrieval interface. The agent issues structured lexical queries, while the retriever executes these queries over a standard inverted index and ranks matched passages with BM25. This design keeps the backend simple and efficient, while giving the LLM direct control over how its search intent is translated into corpus access.

The interface supports two kinds of search control. First, it supports Boolean query logic, allowing the agent to express required terms, alternatives, and exclusions. For example, the agent can require an entity to co-occur with a relation phrase, expand recall with aliases, or remove a known distractor. Second, it supports adjustable matching granularity. The agent can begin with broad keyword matching for exploration, and later switch to exact phrase or entity-level matching when a specific surface form becomes important. This is especially useful in multi-hop questions, where early retrieval may identify an intermediate entity and later retrieval must fetch evidence about that entity precisely.

Concretely, we instantiate the interface with a Lucene-style query-string syntax over the `title` and `content` fields. The agent can combine terms with `AND`, `OR`, and `NOT`, group constraints with parentheses, use quoted phrases for exact matching, and optionally target fields such as `title:"entity name"`.

Given a logical query, the inverted index first determines the candidate documents that satisfy the lexical constraints. BM25 is then used to rank documents within this candidate set. Thus, the logical query controls *what can be retrieved*, while BM25 controls *which matched documents are ranked highest*. This separation makes retrieval behavior more intent-faithful than latent semantic matching: when the agent changes a constraint or matching granularity, the eligible candidate set changes accordingly.

LogicalRAG does not introduce a new scoring function or retrieval architecture. Instead, it provides a lightweight interface tailored for LLM-driven search, combining lexical matching for faithful query execution, logical operators for recovering from failed searches, and BM25 ranking for evidence selection among matched documents. This design enables the LLM to iteratively refine retrieval across interaction turns while avoiding the cost of online query embedding generation, corpus-scale vector indexing, and graph construction.

4 Experimental Setup

To assess its effectiveness–efficiency trade-off in multi-hop QA, we compare LogicalRAG against no-retrieval, hybrid, and graph-based agentic baselines. Our analysis spans answer accuracy, trajectory-level query repair, system efficiency, model scaling, and robustness.

4.1 Datasets and Scales

We conducted evaluations under two corpus-scale settings.

Medium-scale corpora. Following the dataset construction method of HippoRAG (Gutiérrez et al., 2024), we evaluated on HotpotQA (Yang et al., 2018), 2WikiMultiHopQA (Ho et al., 2020), and MuSiQue (Trivedi et al., 2022). For each dataset, we used 1,000 questions and the corresponding dataset-specific Wikipedia subset, containing approximately 9K, 6K, and 11K passages for HotpotQA, 2WikiMultiHopQA, and MuSiQue, respectively. This setting allows for a fair comparison with graph-based and agentic RAG baselines under controlled corpus sizes.

Large-scale corpus. We further evaluated LogicalRAG on the KILT-scale Wikipedia corpus (Petroni et al., 2021). Unlike the medium-scale setting, queries are matched against a substantially larger open-domain knowledge base. This setting places greater demands on indexing efficiency, serving latency, and the ability of each retrieval interface to support multi-step agentic search. Graph-based systems are not evaluated at this scale, as their released pipelines require corpus-level graph construction, which is prohibitively expensive in terms of both time and computational resources.

4.2 Baseline Methods and Systems

We compared LogicalRAG with four baselines. **Direct Answer** produces answers without retrieval and serves as a no-evidence reference point. **Agentic Hybrid** is our main controlled baseline: it uses the same multi-step search procedure as LogicalRAG, but replaces logical lexical search with hybrid sparse-dense retrieval merged by reciprocal rank fusion. **HippoRAG2** (Gutiérrez et al., 2025) represents graph-based RAG and is evaluated only in the medium-scale setting, where graph construction is feasible. **MA-RAG** (Nguyen et al., 2025) is an embedding-based agentic RAG system with a more engineered multi-agent harness, including sub-agents and parallel retrieval calls, and is evaluated as an end-to-end baseline.

For the controlled comparison between LogicalRAG and Agentic Hybrid, both systems use the same lightweight search procedure. Given a question, the model first generates a tentative evidence-seeking plan, then iteratively observes retrieved passages and either issues another retrieval action or produces a final answer. The two systems differ only in retrieval action space: LogicalRAG issues structured lexical queries with logical constraints and adjustable matching granularity, while Agentic Hybrid issues natural-language queries to a hybrid retriever. We intentionally kept the search procedure simple by appending previous searches and retrieved passages to the running context, without specialized memory compression, reranking, or additional controllers. Complete prompts, retrieval limits, RRF parameters, and generation settings are provided in Appendix B.

4.3 Efficiency Evaluation

We measured offline construction cost and online retrieval efficiency on the KILT-scale corpus. Offline cost includes all preprocessing before serving: inverted-index construction for both systems, plus corpus-wide embedding generation and FAISS indexing for Agentic Hybrid (Douze et al., 2025). All indexing and serving measurements were conducted on the same local server; hardware and serving details are provided in Appendix B.

For online efficiency, we replayed retrieval queries from KILT-scale trajectories after warming up the retrieval services. We reported throughput and per-request wall-clock latency under different client-side concurrency levels. LogicalRAG was measured as a full OpenSearch retrieval call (OpenSearch Project, 2026). Agentic Hybrid was measured as a full hybrid retrieval call, including sparse retrieval, query embedding, dense retrieval, document fetching, and rank fusion. Mean and P95 latencies were computed over individual request wall-clock times at each concurrency level. Additional phase-level measurements are provided in Appendix B.

4.4 Evaluation Metrics

Answer accuracy. We reported exact match (EM), word-level F1, and LLM-as-a-Judge accuracy. EM and F1 evaluate surface-form correctness, while LLM-as-a-Judge additionally captures semantically equivalent answers that may differ lexically.

Trajectory-level query repair. We grouped retrieval actions by their underlying search intent and report same-intent result overlap and intent recovery rate. These metrics measure whether query revisions change the retrieved evidence and recover from earlier failed searches. Additional details are provided in Appendix F.

System efficiency. We reported offline index construction cost, online throughput, and per-query retrieval latency, following the efficiency setup described in Section 4.3.

Answer-unavailable robustness. We removed gold supporting passages to create answer-unavailable examples and report refusal, hallucination, and gold leak rates. Hallucination denotes an incorrect, non-refusal answer unsupported by the remaining corpus, while gold leak denotes a correct answer despite evidence removal.

Model scaling. We evaluated Agentic Hybrid and LogicalRAG using multiple Qwen3.5 models (Team, 2026) on the KILT-scale corpus. This experiment investigates how retrieval performance scales with agent capability and examines whether stronger agents can more effectively exploit the high-control action space enabled by LLM-driven logical search.

5 Results

We evaluated LogicalRAG from five perspectives: (i) answer accuracy under medium-scale and KILT-scale corpora, (ii) trajectory-level query refinement, (iii) system efficiency, (iv) how performance changes with agent capability, and (v) robustness when the answer is absent from the

Method	2WikiMultiHopQA			HotpotQA			MuSiQue			Avg. Judge
	EM	F1	Judge	EM	F1	Judge	EM	F1	Judge	
Direct Answer	0.383	0.508	0.596	0.467	0.546	0.597	0.200	0.324	0.340	0.511
Medium-scale corpus										
HippoRAG2	0.575	0.655	0.732	0.560	0.689	0.802	0.366	0.489	0.515	0.683
MA-RAG	0.663	0.766	0.860	0.536	0.669	0.767	0.335	0.485	0.545	0.724
Agentic Hybrid	0.706	0.785	0.867	**0.623**	**0.762**	**0.893**	**0.461**	**0.573**	**0.662**	**0.807**
LogicalRAG	**0.708**	**0.789**	**0.874**	0.622	0.761	0.884	0.410	0.520	0.595	0.784
Large-scale corpus (KILT)										
HippoRAG2		N/A			N/A			N/A		N/A
MA-RAG	0.429	0.501	0.543	0.337	0.457	0.523	0.137	0.252	0.273	0.446
Agentic Hybrid	**0.713**	**0.797**	0.882	**0.534**	**0.668**	**0.794**	**0.302**	0.420	0.471	0.716
LogicalRAG	0.701	0.787	**0.886**	0.524	0.664	0.784	**0.302**	**0.422**	**0.481**	**0.717**

Table 1: Main experimental results on medium-scale and KILT-scale corpora. "Judge" denotes LLM-as-a-Judge accuracy. Avg. Judge is the average Judge score across the three datasets within each corpus setting.

knowledge base. Overall, results showed that a lightweight lexical backend could be competitive with more complex retrieval systems when the interface faithfully executes the LLM's search intent and provides fine-grained retrieval control.

5.1 Main Results: Matching Hybrid Retrieval at Scale

Table 1 reports the main results on medium-scale and KILT-scale corpora. Across both settings, LogicalRAG was competitive with Agentic Hybrid, despite using only lexical matching, logical query control, and BM25 ranking. On the medium-scale corpora, LogicalRAG slightly outperformed Agentic Hybrid on 2WikiMultiHopQA, while Agentic Hybrid remained stronger on HotpotQA and MuSiQue. On the KILT-scale corpus, the two systems reached near parity: the average LLM-as-a-Judge accuracy is 0.717 for LogicalRAG and 0.716 for Agentic Hybrid.

This result is notable because LogicalRAG does not add dense retrieval to the backend. Instead, it relies on the LLM to drive search through explicit lexical constraints. The trajectory analysis in Figure 1 helps explain why this simple interface can be effective. When the agent repeatedly searched for the same information need, LogicalRAG retrieved result sets with lower overlap than Agentic Hybrid and achieved higher recovery from failed searches. This suggests that logical query control helps the agent repair retrieval failures by changing the candidate set more directly, rather than repeatedly retrieving similar evidence.

These results highlight a key observation: in agentic RAG, strong end-to-end performance does not necessarily require increasingly complex retrievers. When the retrieval interface can faithfully execute the LLM's explicit search intent while providing sufficient control for iterative refinement across turns, a lightweight lexical backend can achieve performance comparable to that of a substantially more expensive hybrid retriever. Having established comparable answer accuracy, we next examined the system cost required to achieve this performance.

5.2 Efficiency: Matching Accuracy with Lower Retrieval Cost

Table 2 shows that the comparable accuracy in Table 1 comes with substantially different construction costs. LogicalRAG only builds an inverted index, whereas Agentic Hybrid additionally requires corpus-wide embedding generation and FAISS indexing, resulting in about $41\times$ higher offline construction time. Graph-based retrieval is even less aligned with large-scale deployment, requiring LLM-based entity and relation extraction over the corpus, which amounts to about 37B input tokens and 18B output tokens in our estimate.

The efficiency gain is not attributable to a shorter agent loop, as the two controlled systems use a comparable number of search turns: 3.77 per question for LogicalRAG versus 3.55 for Agentic Hybrid. Instead, the improvement arises from the lighter-weight retrieval backend during online serving. At a concurrency level of 16, LogicalRAG achieved 152.5 QPS with a mean latency of 74.9 ms, whereas Agentic Hybrid reached 66.6 QPS with a mean latency of 230.5 ms (Table 3). This corresponds to $2.3\times$ higher throughput and $3.1\times$

Backend	Main preprocessing	Construction Cost	Relative
LogicalRAG	Inverted index	1.27 h	1.0×
Agentic Hybrid	Inverted index + Embedding + FAISS	52.02 h	41.0×
Graph-based	LLM graph construction	∼37B input + 18B output tokens	Outside budget

Table 2: Offline construction cost on the KILT-scale corpus. Graph-based construction cost is measured in terms of LLM preprocessing tokens.

Backend	QPS ↑	Mean ms ↓	P95 ms ↓
Concurrency = 1			
LogicalRAG	11.8	84.6	232.9
Hybrid	5.4	184.5	279.1
Concurrency = 16			
LogicalRAG	152.5	74.9	234.1
Hybrid	66.6	230.5	294.2

Table 3: Online retrieval efficiency on KILT-scale trajectory queries. Latencies are per-request wall-clock times.

lower mean latency for LogicalRAG.

Together, these results showed that LogicalRAG reached near-parity in answer accuracy while avoiding both corpus-wide embedding construction and online query embedding. By executing the LLM's logical query directly over an inverted index, it retains ranked top-k retrieval with BM25 at substantially lower construction and serving cost. Additional serving details and phase-level measurements are provided in Appendix B.

5.3 Model Scaling: Stronger Agents Make LLM-Driven Search Viable

A central question for LLM-driven search is why such a lightweight retrieval design becomes competitive. Our hypothesis is that logical search becomes more effective as the agent becomes better at decomposing questions, inspecting evidence, and issuing precise query revisions. We therefore evaluated Agentic Hybrid and LogicalRAG on the KILT-scale corpus with multiple Qwen3.5 models.

Figure 2 reports the average LLM-as-a-Judge accuracy across 2WikiMultiHopQA, HotpotQA, and MuSiQue. The overall trend supports this hypothesis. With Qwen3.5-4B, LogicalRAG trailed Agentic Hybrid by about three points, indicating that smaller agents still benefit from the additional semantic smoothing of hybrid retrieval. As the agent became stronger, LogicalRAG improved rapidly and the gap to Agentic Hybrid largely closed. With Qwen3.5-Plus, the two methods reached near parity. The results suggest that LLM-driven search becomes more viable when the agent is capable enough to operate a high-control retrieval interface, turning reasoning intent into explicit lexical constraints and query repairs.

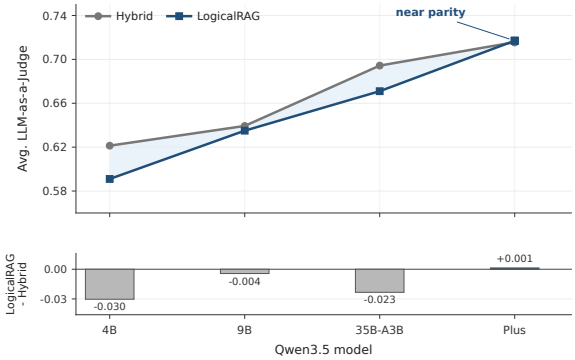

Figure 2: Model scaling on the KILT-scale corpus. As model capability increases, LogicalRAG progressively closes the gap with Agentic Hybrid, reaching near parity on Qwen3.5-Plus.

5.4 Answer-Unavailable Questions: Reducing Hallucination

We evaluated robustness when the corpus lacks sufficient evidence to answer the question. For each dataset, we constructed 400 answer-unavailable examples by removing gold supporting passages. The desired behavior is to refuse rather than produce an unsupported answer. We report refusal, hallucination, and gold leak rates, where gold leak denotes a correct answer despite evidence removal, and hallucination denotes an incorrect non-refusal answer unsupported by the remaining corpus.

Table 4 shows that LogicalRAG improved abstention behavior across all three datasets. On average, its refusal rate increased from 0.767 to 0.828, while the hallucination rate decreased from 0.128 to 0.083.

We attribute this improvement to the legibility of failed logical search. Repeated failures under explicit lexical constraints provide a clearer signal that required evidence may be absent, whereas Agentic Hybrid may still return semantically related but unsupported passages. Thus, LLM-driven logical search helps the agent distinguish missing evidence from merely incomplete retrieval, leading to more appropriate abstention.

Dataset	Method	Refusal ↑	Hallucination ↓	Gold Leak
2Wiki	Hybrid	0.913	0.050	0.038
	LogicalRAG	**0.943**	**0.028**	0.030
HotpotQA	Hybrid	0.660	0.160	0.180
	LogicalRAG	**0.733**	**0.110**	0.158
MuSiQue	Hybrid	0.728	0.175	0.098
	LogicalRAG	**0.808**	**0.110**	0.083
Avg.	Hybrid	0.767	0.128	0.105
	LogicalRAG	**0.828**	**0.083**	0.090

Table 4: Answer-unavailable evaluation after removing gold supporting passages. Hallucination denotes an incorrect non-refusal answer unsupported by the remaining corpus.

6 Discussion

Logical search and `grep`-style access. Recent agent systems have explored direct text access through `grep`-style tools (Subramanian et al., 2025; Wang et al., 2026; Li et al., 2026). LogicalRAG shares the same motivation of giving the agent explicit lexical control, but differs in the retrieval substrate. A literal `grep` interface is effective for exact string matching, yet it provides limited ranking and can be inefficient for large-scale top-k evidence retrieval. LogicalRAG implements lexical control with a standard inverted-index engine, supporting Boolean logic, phrase matching, and BM25 ranking while retaining efficient retrieval over large corpora. Thus, `grep`-style search can be viewed as a useful special case of direct lexical access, whereas LogicalRAG targets ranked, large-scale RAG over document collections.

Why not combine dense and logical tools? A natural extension is to expose both dense retrieval and logical search to the agent (Du et al., 2026). Such a system may improve accuracy when relevant evidence is difficult to express lexically. We did not study this combined setting as our main system because it answers a different question. Our goal was to isolate how far LLM-driven search can go with a lightweight lexical backend under a controlled agentic loop. Adding dense retrieval would reintroduce corpus-wide embedding construction and online query encoding, reducing the efficiency advantage that motivates our design. Hybrid tool use is therefore a promising extension, but not the focus of this work.

Interface ablations. Additional ablations in Appendix C suggest that LogicalRAG is not highly sensitive to prompt engineering. Removing Boolean operators moderately reduces performance, while replacing the full tool description with a minimal syntax-only description causes little degradation. These results support our interpretation that an intent-faithful lexical search interface drives the main performance benefit, independent of prompt engineering.

7 Conclusion

In this study, we revisited the design of agentic RAG and challenged the prevailing trend of building increasingly complex retrieval backends. We proposed a shift in perspective toward a "heavy brain, precise hands" paradigm, where the complex reasoning and query planning are centralized within the LLM (the "heavy brain"), while the retrieval backend is simplified into a lightweight, deterministic execution layer (the "precise hands"). Implementing this philosophy, we introduced LogicalRAG, a framework that empowers LLMs to formulate precise retrieval intents using compositional logical expressions, executed over a lightweight inverted-index backend. Our evaluations demonstrate that this framework matches the performance of dense, hybrid, and graph-based baselines, while achieving significant reductions in downstream hallucinations and system overhead. More broadly, our findings suggest that agentic RAG may benefit from a "heavy brain, precise hands" paradigm, in which increasingly capable LLMs perform sophisticated retrieval planning and reasoning, while retrieval systems serve as faithful executors of the model's structured intent.

Limitations

Although LogicalRAG demonstrates strong performance across a wide range of tasks, several limitations remain. First, our framework depends on the ability of LLMs to formulate accurate logical queries. When the model fails to identify the appropriate constraints, entities, or query structure,

retrieval quality may degrade correspondingly. Second, while logical retrieval provides strong precision and interpretability, embedding-based retrieval may still offer advantages in scenarios requiring highly abstract semantic matching or fuzzy associations that are difficult to express explicitly through logical constraints. To alleviate this, our retrieval backend also supports natural language access, although such retrieval is generally less accurate than embedding-based approaches. In addition, our current implementation primarily focuses on textual retrieval settings, and extending the framework to multimodal or continuously evolving knowledge environments remains an important direction for future work. Nevertheless, we believe these limitations do not diminish the broader promise of the proposed paradigm for building simpler, more controllable, and more reliable agentic RAG systems.

References

Anthropic. 2024. Claude code: Tools reference. Accessed: 2026-04-21.

Manoj Ghuhan Arivazhagan, Lan Liu, Peng Qi, Xinchi Chen, William Yang Wang, and Zhiheng Huang. 2023. Hybrid hierarchical retrieval for open-domain question answering. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 10680–10689, Toronto, Canada. Association for Computational Linguistics.

Akari Asai, Zeqiu Wu, Yizhong Wang, Avi Sil, and Hannaneh Hajishirzi. 2024. Self-rag: Learning to retrieve, generate, and critique through self-reflection. In *International Conference on Learning Representations*, volume 2024, pages 9112–9141.

Matthijs Douze, Alexandr Guzhva, Chengqi Deng, Jeff Johnson, Gergely Szilvasy, Pierre-Emmanuel Mazaré, Maria Lomeli, Lucas Hosseini, and Hervé Jégou. 2025. The faiss library. *IEEE Transactions on Big Data*.

Mingxuan Du, Benfeng Xu, Chiwei Zhu, Shaohan Wang, Pengyu Wang, Xiaorui Wang, and Zhendong Mao. 2026. A-RAG: Scaling agentic retrieval-augmented generation via hierarchical retrieval interfaces. *arXiv preprint arXiv:2602.03442*.

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

Luyu Gao, Xueguang Ma, Jimmy Lin, and Jamie Callan. 2023. Precise zero-shot dense retrieval without relevance labels. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1762–1777.

Bernal J Gutiérrez, Yiheng Shu, Yu Gu, Michihiro Yasunaga, and Yu Su. 2024. Hipporag: Neurobiologically inspired long-term memory for large language models. *Advances in neural information processing systems*, 37:59532–59569.

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

Xanh Ho, Anh-Khoa Duong Nguyen, Saku Sugawara, and Akiko Aizawa. 2020. Constructing a multi-hop qa dataset for comprehensive evaluation of reasoning steps. In *Proceedings of the 28th International Conference on Computational Linguistics*, pages 6609–6625.

Gautier Izacard and Edouard Grave. 2021. Leveraging passage retrieval with generative models for open domain question answering. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 874–880, Online. Association for Computational Linguistics.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6769–6781, Online. Association for Computational Linguistics.

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. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. In *Advances in Neural Information Processing Systems*, volume 33, pages 9459–9474. Curran Associates, Inc.

Xiaoxi Li, Guanting Dong, Jiajie Jin, Yuyao Zhang, Yujia Zhou, Yutao Zhu, Peitian Zhang, and Zhicheng Dou. 2025a. Search-o1: Agentic search-enhanced large reasoning models. In *Proceedings of the 2025 Conference on Empirical Methods in Natural Language Processing*, pages 5420–5438.

Yangning Li, Weizhi Zhang, Yuyao Yang, Wei-Chieh Huang, Yaozu Wu, Junyu Luo, Yuanchen Bei, Henry Peng Zou, Xiao Luo, Yusheng Zhao, Chunkit Chan, Yankai Chen, Zhongfen Deng, Yinghui Li, Hai-Tao Zheng, Dongyuan Li, Renhe Jiang, Ming Zhang, Yangqiu Song, and Philip S. Yu. 2025b. Towards agentic rag with deep reasoning: A survey of rag-reasoning systems in llms. *Preprint*, arXiv:2507.09477.

Zhuofeng Li, Haoxiang Zhang, Cong Wei, Pan Lu, Ping Nie, Yi Lu, Yuyang Bai, Shangbin Feng, Hangxiao Zhu, Ming Zhong, Yuyu Zhang, Jianwen Xie, Yejin

Choi, James Zou, Jiawei Han, Wenhu Chen, Jimmy Lin, Dongfu Jiang, and Yu Zhang. 2026. Beyond semantic similarity: Rethinking retrieval for agentic search via direct corpus interaction. *Preprint*, arXiv:2605.05242.

Thang Nguyen, Peter Chin, and Yu-Wing Tai. 2025. Ma-rag: Multi-agent retrieval-augmented generation via collaborative chain-of-thought reasoning. *arXiv preprint arXiv:2505.20096*.

OpenSearch Project. 2026. OpenSearch. https://github.com/opensearch-project/OpenSearch.

Fabio Petroni, Aleksandra Piktus, Angela Fan, Patrick Lewis, Majid Yazdani, Nicola De Cao, James Thorne, Yacine Jernite, Vladimir Karpukhin, Jean Maillard, Vassilis Plachouras, Tim Rocktäschel, and Sebastian Riedel. 2021. KILT: a benchmark for knowledge intensive language tasks. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 2523–2544, Online. Association for Computational Linguistics.

Stephen Robertson and Hugo Zaragoza. 2009. *The probabilistic relevance framework: BM25 and beyond*, volume 4. Now Publishers Inc.

Shreyas Subramanian, Adewale Akinfaderin, Yanyan Zhang, Ishan Singh, Mani Khanuja, Sandeep Singh, and Maira Ladeira Tanke. 2025. Keyword search is all you need: Achieving rag-level performance without vector databases using agentic tool use. *arXiv preprint arXiv:2602.23368*.

Qwen Team. 2026. Qwen3.5: Accelerating productivity with native multimodal agents.

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.

Harsh Trivedi, Niranjan Balasubramanian, Tushar Khot, and Ashish Sabharwal. 2023. 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 (volume 1: long papers)*, pages 10014–10037.

Baoyi Wang, Xingliang Wang, Guochang Li, Chen Zhi, Junxiao Han, Xinkui Zhao, Nan Wang, Shuiguang Deng, and Jianwei Yin. 2026. GrepRAG: An empirical study and optimization of grep-like retrieval for code completion. *arXiv preprint arXiv:2601.23254*.

Xiaohua Wang, Zhenghua Wang, Xuan Gao, Feiran Zhang, Yixin Wu, Zhibo Xu, Tianyuan Shi, Zhengyuan Wang, Shizheng Li, Qi Qian, Ruicheng Yin, Changze Lv, Xiaoqing Zheng, and Xuanjing Huang. 2024. Searching for best practices in retrieval-augmented generation. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pages 17716–17736, Miami, Florida, USA. Association for Computational Linguistics.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D Manning. 2018. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 conference on empirical methods in natural language processing*, pages 2369–2380.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik R Narasimhan, and Yuan Cao. 2022. React: Synergizing reasoning and acting in language models. In *The eleventh international conference on learning representations*.

Mingtian Zhang, Yu Tang, and PageIndex Team. 2025. PageIndex: Next-generation vectorless, reasoning-based rag. *PageIndex Blog*. https://pageindex.ai/blog/pageindex-intro.

Xiangrong Zhu, Yuexiang Xie, Yi Liu, Yaliang Li, and Wei Hu. 2025. Knowledge graph-guided retrieval augmented generation. In *Proceedings of the 2025 Conference of the Nations of the Americas Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 8912–8924.

Luyao Zhuang, Shengyuan Chen, Yilin Xiao, Huachi Zhou, Yujing Zhang, Hao Chen, Qinggang Zhang, and Xiao Huang. 2025. Linearrag: Linear graph retrieval augmented generation on large-scale corpora. *arXiv preprint arXiv:2510.10114*.

A Prompts and Tool Descriptions

A.1 LogicalRAG Search Tool

LogicalRAG Search Tool Description (summary)

```
Backend: BM25 sparse retrieval with Lucene query-string
      syntax.

Supported query controls:
  - Boolean logic: AND, OR, NOT, parentheses
  - Phrase matching: "exact phrase"
  - Field targeting: title:..., content:...
  - Term boosting: ^N
  - default_operator: OR or AND

Search strategy:
  - Start broad with distinctive keywords.
  - Narrow with AND, phrase matching, fields, or NOT when
      results are too broad.
  - Simplify or relax constraints when there are no hits.
  - For multi-hop questions, search one hop at a time.
  - Avoid repeated paraphrases that retrieve the same
      documents.

Examples:
  - broad keyword search
  - broad-to-narrow query refinement
  - distractor exclusion
  - exact title / phrase search
```

A.2 Agentic Hybrid Search Tool

Agentic Hybrid Search Tool Description (summary)

```
Backend: hybrid retrieval with BM25 keyword matching and
      FAISS dense retrieval,
merged by Reciprocal Rank Fusion (RRF).

Use case:
  - Robust retrieval when exact lexical overlap is
      incomplete.
  - Useful for paraphrases, aliases, indirect wording,
      and semantic matches.
  - Still benefits from explicit entity names, titles,
      years, locations, and rare relation terms.

Query format:
  - Use a short natural-language search query.
  - Do not use Boolean syntax such as AND, OR, NOT.
  - Prefer plausible semantic phrasing over loose keyword
      bags.
  - Unlike LogicalRAG, this tool does not expose logical
      query control;
    the agent controls retrieval through natural-language
      reformulation.

Search strategy:
  - Start with the most distinctive entity names and
      relation words.
  - For multi-hop questions, search one hop at a time.
  - If results are noisy, rewrite the query into a
      cleaner query for the current hop.
  - If direct wording fails, try aliases, paraphrases,
      adjectival forms,
    nominalizations, or related surface forms.
  - Increase max_results when more candidates around a
      relevant topic are needed.

Examples:
  - book title -> author
  - author -> education
  - direct wording -> paraphrased retry
  - focused topic -> increase max_results
```

B Experimental Settings

Table 5 summarizes the main retrieval and generation settings. We use publicly available datasets, models, and software under their released terms and do not redistribute these artifacts.

Component	Setting
Sparse engine	OpenSearch
Indexed fields	`title`, `content`
Analyzer	OpenSearch default
BM25	$k_1 = 1.2,\ b = 0.75$
Hybrid fusion	RRF, $K = 60$
Embedding model	All embedding-based components use Qwen3-Embedding-0.6B.
Query instruction	`Given a web search query, retrieve relevant passages that answer the query`
Medium FAISS	Flat index
KILT FAISS	IVFPQ
KILT FAISS params	`nprobe=64`, `hnsw_m=32`, `pq_m=64`, `pq_nbits=8`
Max search turns	8
Agent model	Qwen3.5-Plus unless specified
Agent decoding	temp $= 0.6$, top-$p = 0.95$
Judge model	Qwen3.5-Plus
Judge decoding	temp $= 0.3$
Hardware	Intel Xeon Gold 6226R CPU, 4 RTX 3090 GPUs with 24GB memory each, 128GB RAM

Table 5: Experimental settings.

C Ablation Study

We conducted two lightweight ablations on the first 400 examples of each KILT-scale dataset.

No logical operators. This variant disables `AND`, `OR`, and `NOT`, while still allowing keyword and phrase queries.

Syntax-only tool description. This variant keeps the query grammar but removes BM25-specific explanations and search strategies.

Table 6 reports EM, F1, and Judge results; the overall trend is consistent across metrics.

Dataset	Variant	EM	F1	Judge
2Wiki	No logical ops	0.690	0.782	0.878
	Syntax-only	0.708	0.799	0.900
	LogicalRAG	**0.710**	**0.803**	**0.907**
HotpotQA	No logical ops	0.548	0.669	0.783
	Syntax-only	0.548	**0.683**	0.797
	LogicalRAG	**0.555**	0.679	**0.800**
MuSiQue	No logical ops	**0.320**	0.423	0.476
	Syntax-only	0.308	**0.435**	0.499
	LogicalRAG	0.315	0.427	**0.501**
Avg.	No logical ops	0.519	0.625	0.712
	Syntax-only	0.521	0.639	0.732
	LogicalRAG	**0.527**	**0.636**	**0.736**

Table 6: Full ablation results on the first 400 examples of each KILT-scale dataset.

D Example Query Trajectory

Example LogicalRAG Trajectory

Question. What composer with an Italian libretto was born 4 March 1678?

Plan. Identify the person born on 4 March 1678, verify that the person is a composer, and confirm the Italian-libretto constraint.

Search 1. `born 4 March 1678`, `default_operator=AND`.
Key evidence. A timeline passage states that "Italian Baroque composer Antonio Vivaldi is born on 4 March." The *Antonio Vivaldi* page further states that "Antonio Lucio Vivaldi ... 4 March 1678–28 July 1741" was an Italian Baroque composer.

Search 2. `Antonio Vivaldi operas Italian libretto`, `default_operator=OR`.
Key evidence. Retrieved passages include *Griselda (Vivaldi)*, which states that the opera uses an Italian libretto, along with other Vivaldi opera pages involving librettos.

Answer. *Antonio Vivaldi*. The gold answer is *Antonio Lucio Vivaldi*; LLM-as-a-Judge marks the answer as correct.

E Model Scaling Details

Model	Method	2Wiki	Hotpot	MuSiQue	Avg.	Gap
4B	Hybrid	0.823	0.707	0.334	0.621	–
	Logical	0.788	0.677	0.308	0.591	-0.030
9B	Hybrid	0.834	0.724	0.360	0.639	–
	Logical	0.816	0.728	0.361	0.635	-0.004
35B-A3B	Hybrid	0.867	0.777	0.439	0.694	–
	Logical	0.848	0.761	0.404	0.671	-0.023
Plus	Hybrid	0.882	0.794	0.471	0.716	–
	Logical	0.886	0.784	0.481	0.717	+0.001

Table 7: Model scaling results with LLM-as-a-Judge accuracy. Hybrid denotes Agentic Hybrid, Logical denotes LOGICALRAG, and Gap is Logical minus Hybrid on Avg.

Model	Method	Metric	2Wiki	Hotpot	MuSiQue	Avg.	Gap
4B	Hybrid	EM	0.604	0.427	0.205	0.412	–
	Hybrid	F1	0.707	0.559	0.306	0.524	–
	Logical	EM	0.586	0.425	0.194	0.402	-0.010
	Logical	F1	0.682	0.544	0.286	0.504	-0.020
9B	Hybrid	EM	0.592	0.431	0.224	0.416	–
	Hybrid	F1	0.693	0.574	0.334	0.534	–
	Logical	EM	0.578	0.436	0.221	0.412	-0.004
	Logical	F1	0.682	0.570	0.326	0.526	-0.008
35B-A3B	Hybrid	EM	0.659	0.479	0.263	0.467	–
	Hybrid	F1	0.754	0.619	0.379	0.584	–
	Logical	EM	0.629	0.489	0.249	0.456	-0.011
	Logical	F1	0.728	0.622	0.359	0.570	-0.014
Plus	Hybrid	EM	0.713	0.534	0.302	0.516	–
	Hybrid	F1	0.797	0.668	0.420	0.628	–
	Logical	EM	0.701	0.524	0.302	0.509	-0.007
	Logical	F1	0.787	0.664	0.422	0.624	-0.004

Table 8: EM and F1 model scaling results on the KILT-scale datasets. Gap is Logical minus Hybrid on Avg.

F Trajectory Metric Details

We computed the trajectory metrics in Figure 1 from retrieval actions in agent trajectories. For each trajectory, we first grouped retrieval turns by their underlying search intent. The grouping was produced by an LLM judge using only the issued retrieval queries, not the final answer or gold evidence. Queries were assigned to the same intent if they attempted to retrieve the same missing fact or evidence, even when their surface forms differed. We then applied deterministic post-processing to ensure that every retrieval turn was assigned to exactly one intent group.

We also judged whether each retrieval turn succeeded. A turn was considered successful if at least one of its top-K retrieved passages was judged relevant to the issued query. Only this binary success label was used when computing recovery.

Same-intent overlap. For each intent group with at least two retrieval turns, we computed the overlap between adjacent top-K retrieval results as the fraction of shared retrieved passage identifiers. We then averaged these adjacent overlaps within the group. The reported same-intent overlap is the macro average over all repeated-intent groups. Lower overlap indicates that query revisions more effectively change the retrieved evidence for the same underlying information need.

Intent recovery. Intent recovery measures whether later query revisions can repair an initially failed search intent. For each intent group, we checked whether the first retrieval attempt succeeded and whether any retrieval attempt in the same group eventually succeeded. A group was considered recoverable only when it contained at least two retrieval turns and its first attempt failed. It was considered recovered if any later attempt in the same group succeeded.

The intent recovery rate is the fraction of recoverable groups that were recovered. Thus, singleton intents and intents that succeeded on the first attempt were excluded from the denominator. This metric captures whether the agent can recover from an initial retrieval failure for the same search intent, rather than simply rewarding trajectories that issue more retrieval calls.