AgenticRAG: Agentic Retrieval for Enterprise Knowledge Bases

Susheel Suresh[*†], **Hazel Mak**[*‡], **Shangpo Chou, Fred Kroon, Sahil Bhatnagar**

Microsoft Corporation

Abstract

We present **AgenticRAG**, a practical agentic harness for retrieval and analysis over enterprise knowledge bases. Standard RAG pipelines place significant burden of grounding on the search stack, constraining the language model to a fixed candidate set chosen deep in the retrieval process. Our approach reduces this overdependence by layering a lightweight harness on top of existing enterprise search infrastructure, equipping a reasoning LLM with search, find, open, and summarize tools enabling the model to iteratively retrieve information, navigate within documents, and analyze evidence autonomously. On three open benchmarks we observe substantial gains: 49.6% recall@1 on BRIGHT (+21.8 pp over the best embedding baseline), 0.96 factuality on WixQA (+13% relative improvement), and 92% answer correctness on FinanceBench–within 2 pp of oracle access to true evidence. Ablation studies show that the most significant factor is the shift from single-shot retrieval to agentic tool use (5.9× improvement), while multi-query search and in-document navigation contribute to both quality and efficiency. We present various design choices in our agentic harness that were informed by pre-production deployments. Our results demonstrate its suitability for real-world enterprise production environments.

1 Introduction

Standard retrieval-augmented generation (RAG) pipelines follow a static retrieve-then-generate paradigm (Lewis et al., 2020). In this design, the search stack effectively determines the final candidate set the large language model (LLM) will see and the model's reasoning is constrained to that set. Modern enterprise-grade search stacks are highly optimized for scalability, latency, and multi-stage ranking pipelines built on inverted indexes, probabilistic retrieval, and learned ranking models (Liu et al., 2009; Nogueira and Cho, 2019; Thakur et al., 2021). These systems excel at keyword and short semantic queries and are strong for high-recall candidate generation. However, they are not designed to resolve situational, multi-document, or analytically complex information needs—the kinds of queries knowledge workers issue against dense corpora such as technical manuals, compliance documents, and financial reports.

Real-world RAG systems (AzureAISearch) attempt to compensate for these limitations through retrieval enhancement techniques such as HyDE (Gao et al., 2023), multi-query reformulation (Wang et al., 2023), and adaptive or iterative retrieval strategies (Trivedi et al., 2023; Jeong et al., 2024). While these methods provide robustness to query phrasing and higher retrieval coverage, they largely preserve the same architectural assumption: retrieval decisions are finalized before substantive reasoning begins. The LLM still operates over a fixed candidate set selected deep in the search stack, without the ability to iteratively navigate documents, synthesize evidence across sources, or reassess results from a higher-level vantage point.

Recent advances in reasoning-capable language models have demonstrated strong performance on planning and iterative external tool use (Yao et al., 2023; Schick et al., 2023). Rather than hard-coding retrieval steps, we can empower the model itself to drive the process—deciding what to search for, which documents warrant deeper investigation, and when sufficient evidence has been gathered. This relaxes the pressure on the search stack: it only needs to achieve good recall, while the model handles the final precision from its broader context. We present *AgenticRAG*, a practical harness that equips a reasoning LLM with four tools—**search**, **find**, **open**, and **summarize**—layered on top of existing enterprise search infrastructure. The search tool

[*]Equal contribution.
[†]sussuresh@microsoft.com
[‡]hazelmak@microsoft.com

delegates to the underlying search stack for broad candidate discovery, while find and open serve as precision instruments that let the model drill into candidate documents via in-document search and full-content retrieval (with rolling window access). To manage the growing context during long reasoning chains, the harness monitors token usage and triggers the summarize tool when a threshold is reached, allowing the model to consolidate its findings while preserving key references. Our contribution is system-level: a lightweight inference-time tool harness that requires no model fine-tuning, custom embedding model, graph construction, or corpus-specific preprocessing beyond indexing documents into the existing enterprise search backend.

We evaluate on three benchmarks spanning retrieval, enterprise QA, and financial document reasoning. Our approach achieves 49.6% recall@1 on BRIGHT (+21.8 pp over the best embedding baseline), 0.96 factuality on WixQA (+13% relative), and 92.00% answer correctness on FinanceBench–within 2 pp of oracle access. Our method is deployed for pre-production evaluation, and learnings from these deployments directly inform our design choices. We provide detailed ablations analyzing the contribution of each tool, the effect of multi-query search, and model-level differences in retrieval strategy.

2 Related Work

Retrieval-Augmented Generation (RAG) grounds LLM generation in external corpora to mitigate parametric memory limitations (Lewis et al., 2020; Guu et al., 2020). Early approaches focused on identifying relevant documents using sparse or dense vector retrieval (Khattab and Zaharia, 2020; Izacard and Grave, 2021) to enhance performance on knowledge-intensive NLP tasks. As context windows expanded, research shifted toward scaling retrieval to trillions of tokens (Borgeaud et al., 2022) and optimizing in-context learning (Ram et al., 2023; Shi et al., 2023). Despite these advancements, standard RAG pipelines often struggle with "long-tail" knowledge and can suffer from hallucinations when retrieval fails (Mallen et al., 2023; Gao et al., 2024). Furthermore, static "retrieve-then-generate" paradigms lack the flexibility to handle complex, multi-hop queries that require iterative information gathering (Jiang et al., 2023; Press et al., 2023).

To address the brittleness of static pipelines, the field has evolved toward *Agentic* patterns, where autonomous agents (LLMs) dynamically orchestrate the retrieval process (Singh et al., 2025; Oche et al., 2025). Foundational work in agentic behaviors, such as ReAct (Yao et al., 2023) and Toolformer (Schick et al., 2023), demonstrated that LLMs could effectively wield external tools to solve reasoning problems. This paradigm has been formalized in systems like Self-RAG (Asai et al., 2024) and Corrective RAG (Yan et al., 2024), which employ self-reflection mechanisms to critique retrieved content and trigger fallbacks (e.g., web search) when necessary. Recent approaches propose to integrate retrieval into planning: Plan-RAG (Lee et al., 2024) and Search-o1 (Li et al., 2025) separate high-level planning from low-level execution, allowing agents to decompose complex queries into sub-tasks. Similarly, Search-R1 (Jin et al., 2025) uses reinforcement learning to train LLMs for autonomous search decisions. While effective, many of these systems are designed for open-domain search or require fine-tuning, reinforcement learning, or dedicated retrieval policies, which makes them less directly applicable to proprietary enterprise corpora that cannot be exported for training. They can also incur high latency and token costs due to recursive reasoning loops (Trivedi et al., 2023).

Another critical limitation in standard RAG is the "flattening" of documents into disjointed chunks, which discards valuable structural priors like headings and document boundaries. RAPTOR (Sarthi et al., 2024) addresses this by recursively clustering and summarizing text chunks into a tree structure, enabling retrieval at varying levels of abstraction. Similarly, HiQA (Chen et al., 2024) constructs multi-document hierarchical contexts. Graph RAG (Edge et al., 2024; Scaffidi et al., 2025) approaches seek to build knowledge graphs from documents to support query-focused summarization. While powerful for unifying knowledge (Pan et al., 2024; Wang et al., 2024), graph construction is often computationally prohibitive for dynamic enterprise environments. In contrast, our Agentic RAG harness is an inference-time system that leverages a reasoning model with a "search" tool (using a fast enterprise grade search stack) alongside "find" and "open" tools for deeper information gathering and reasoning. This positions the contribution as a deployable system integration for enterprise file systems: it works with existing search infrastructure, preserves document access controls,

and avoids extensive pre-computation or retraining.

3 Method

3.1 System Overview

We present an agentic RAG system for enterprise document search and question answering over large file systems. Unlike traditional single-pass RAG pipelines, our system employs an iterative reasoning loop where a large language model (LLM) autonomously decides when to search for documents, drill into specific passages, and retrieve full content before producing a final answer.

The system addresses several challenges in enterprise RAG: (1) multi-step reasoning: complex queries require information from multiple documents, (2) context window constraints: accumulated retrieval results must fit within LLM limits, (3) grounded responses: answers must include traceable citations to source documents, and (4) multi-turn efficiency: follow-up queries should reuse previously retrieved content rather than re-executing redundant searches. Our architecture supports multiple model families and reuses existing search infrastructure for the backend implementation of the retrieval tools. By lightweight, we mean that the harness consists of four tools, requires no model fine-tuning, no graph construction, and no custom embedding index beyond the enterprise search stack already deployed for document discovery. Overall the system comprises three main components:

1. **Agentic Loop**: Orchestrates LLM-tool interactions, bounded by maximum iterations.

2. **Retrieval Tools**: Three tools (*search*, *find*, *open*) provide hierarchical access to enterprise documents. A *summarize* tool for context management during long reasoning chains.

3. **Conversation State**: Maintains message history, token accounting, and reference ID mappings that track documents across iterations.

3.2 Agentic Loop

The agent processes each query through a bounded iteration loop (Figure 1). Each iteration, upon receiving the current conversation, the agent either selects a tool to call and appends to the conversation, or returns the final answer with citations.

The loop terminates under two conditions: (1) the model produces a text response, or (2) the iteration count reaches maximum iterations (default: 15). When maximum iterations are reached without a final answer, the agent issues a forced completion request, requiring the model to respond using available information. If the token budget is exceeded during execution, the agent triggers context management (Sec. 3.4) to free space and continues the loop. For detailed algorithm, see Appendix A.1.

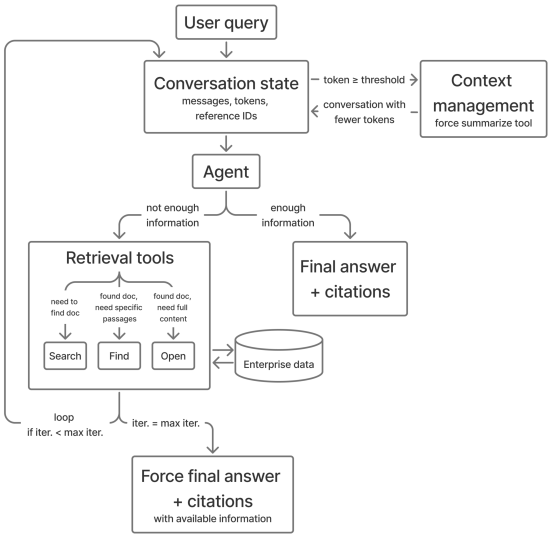

Figure 1: Agentic loop

3.3 Retrieval Tools

The system provides three retrieval tools enabling hierarchical document exploration (Table 1). The agent decides which to invoke based on current information needs.

Search performs enterprise-wide document discovery by delegating to the existing enterprise search stack. In the default configuration, the model may issue up to five query reformulations in one tool call. The tool returns up to 10 results per query, each containing a snippet, title, filename, file type, and other available metadata. Results from multiple queries are combined and deduplicated. Each result receives a unique reference ID (format: turn$m$search$n$) using a globally incrementing counter, enabling subsequent find and open operations.

Find performs targeted in-document search within a single document identified by its reference ID. Given a list of keyword patterns, lexical matching is case-insensitive substring matching; an optional semantic find mode can also be enabled. The tool returns up to 2 matching passages per pattern. Results are deduplicated by content and truncated at a bounded token limit ($\sim$11k tokens).

Find is most useful when the model knows *what* to look for, such as a revenue metric or a named concept inside a long filing.

Open retrieves full document content in a fixed line window. Each call returns a window of lines (default: 1,800) starting from either the beginning (line 0) or a specific line number chosen by the agent, and a response header indicating the viewing range and total document length (e.g., "Viewing lines [0–1799] of 3000 lines"). To access more than one portion from a file, the model makes subsequent calls with an explicit line number value. This enables navigation through documents exceeding the window size while keeping each response bounded. Open is most useful when the model knows *where* to read, such as context around a table, section heading, or line-numbered preview. The system prompt guides effective tool usage. See Appendix A.2 for details.

3.4 Context Management

Since retrieval tools can load ∼11k tokens from files each time, the context window can be used up quickly. To manage that, the harness monitors token usage against a 128K-token threshold: it emits an internal warning when the conversation reaches 90% of the budget and forces summarization at the threshold. The summarize tool enables the model to record current reasoning and designate which references to preserve. The system then scans tool messages and removes content not associated with preserved reference IDs, freeing tokens while retaining cited evidence. This approach extends effective context capacity. See Figure 2 for an example conversation history before and after context management.

Figure 2: Example conversation history with context management via forced summarize tool call.

4 Experiment Setup

Our goal is to evaluate AgenticRAG in realistic enterprise settings where knowledge workers issue complex situational queries requiring multi-step reasoning over large corpora of long, domain-specific documents. To this end we adopt the **BRIGHT benchmark** (Su et al., 2024) which contains StackExchange questions spanning eight domains. We choose to evaluate on the long-context setting of BRIGHT, where documents correspond to entire web pages rather than snippets and the task is to retrieve the full relevant document(s) for a given query. For our agentic setting, the full BRIGHT web pages are converted to document files and indexed into the same enterprise search backend used by the search tool. Search returns snippet previews with metadata and reference IDs, and the find and open tools then access full document content through those IDs. Standard protocol of using Recall@1 to measure relevance of retrieved documents is adopted (Su et al., 2024). We instruct the model in AgenticRAG to provide relevancy scores for the citations it uses when producing the answer, which induces a ranking over documents for evaluation. We also test our method on **WixQA** (Cohen et al., 2025), which targets real-world support and troubleshooting enterprise scenarios that require multi-document and multi-step reasoning for procedural answers. We adopt the same LLM based factuality metric defined in WixQA. Finally, we run evaluations on the popular **FinanceBench** (Islam et al., 2023) dataset, which contains financial questions that require deep reasoning over large company financial documents. Our metric here is answer correctness as a proxy for accurate information retrieval, since questions pertain to single documents. Detailed benchmark descriptions, query set and corpus statistics are presented in Appendix B.

5 Results

5.1 Long-Context Retrieval on BRIGHT

Table 2 presents retrieval performance on the BRIGHT benchmark, showing the best baseline per category. Full results with all models are in Appendix Table 9, including reasoning-

Table 1: Retrieval Tool Specifications

Tool	Definition	Input	Output
SEARCH	Discover relevant documents from entire corpus	queries	Snippets with reference ID and file metadata ($\leq$10 per query)
FIND	Locate specific information from single document	reference id, patterns	Passages ($\leq$2 per pattern, $\leq$11k tokens total)
OPEN	Retrieve windowed full content from single document	reference id, line number (optional)	Line-numbered content ($\leq$1800 lines)

Table 2: Long-context retrieval performance on unsplit web pages of StackExchange data from BRIGHT benchmark. Scores are reported in recall@1. Best baseline per category shown; full results in Table 9.

Category	Model	Bio.	Earth.	Econ.	Psy.	Rob.	Stack.	Sus.	Pony	Avg.
Sparse	BM25	10.7	15.4	10.7	8.4	7.4	22.2	10.7	5.4	11.4
Open-source Emb.	Qwen	39.2	36.1	25.7	42.3	21.3	23.5	33.1	1.3	27.8
Proprietary Emb.	Voyage	34.4	35.4	26.7	41.6	12.9	12.8	31.1	1.3	24.5
Reasoning Enhanced	ReDI	28.4	22.4	21.2	32.0	19.8	36.3	21.7	–	26.0
Ours (AgenticRAG)	GPT-5-mini	61.7	48.1	41.4	65.3	39.4	**40.6**	46.6	4.8	43.5
⌞search, find, open, summ.	**Claude Sonnet 4.5**	**62.3**	**60.0**	**58.7**	**67.9**	**55.0**	34.1	**51.7**	**7.1**	**49.6**

enhanced baselines such as LLM re-rankers over BM25/SBERT and ReDI. Our agentic harness equipped with search, find, open, and summarize tools enables both Claude Sonnet 4.5 and GPT-5-mini to achieve the highest recall@1 across all eight benchmark splits compared to all baselines. With our AgenticRAG retrieval harness, Claude Sonnet 4.5 achieves **49.6%** average recall@1 (+21.8 pp over Qwen, the best embedding model at 27.8%) and GPT-5-mini reaches 43.5% (+15.7 pp). Gains are consistent across domains, with the largest improvements in Economics (+33.0 pp), Earth Science (+24.6 pp), Robotics (+33.7 pp), and Psychology (+25.6 pp). Even with a vast corpus of 5.65K long documents averaging 16K tokens each, our method scales by leveraging traditional retrieval via the search tool, deeper reasoning enabled by the open / find tools and effective context window management by the summarize tool. The best-performing reasoning-enhanced baseline, ReDI, uses a fine-tuned Qwen3-8B decomposition and retrieval-fusion model and achieves 26.0% recall@1 in the BRIGHT long-document setting. Our method outperforms it by +23.6 pp (via Claude) and +17.5 pp (via GPT-5-mini). Static approaches such as one-time query rewriting or LLM-based reranking cannot match the iterative reasoning that our harness provides, and the gap is evident across all splits.

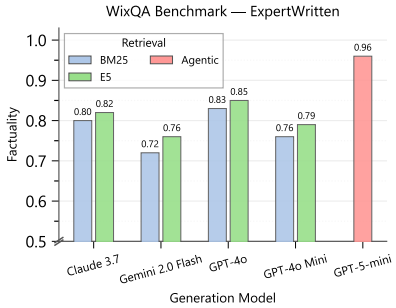

Figure 3: Factuality performance on the WixQA Expert Written dataset (N=200). Our agentic approach (red) substantially outperforms BM25 (blue) and E5 (green) retrieval baselines across all generation models.

5.2 Enterprise QA on WixQA

Figure 3 presents factuality results on the WixQA benchmark, which requires multi-document analysis to answer enterprise support questions. Semantic embeddings alone fail to capture the cross-document reasoning needed for these queries, whereas the iterative search and reasoning enabled by our harness excels. On the *Expert Written* split, our method with GPT-5-mini achieves a factuality score of **0.96**, compared to 0.85 for E5 retrieval and 0.83 for BM25—a **13%** relative improvement over the best baseline. We observe similar gains on the *Simulated* split; see Appendix C.3 for details.

5.3 Financial Document QA on FinanceBench

Table 3 presents answer correctness on the FinanceBench dataset, which evaluates question an-

Table 3: Evaluation results on FinanceBench (N=150).

Method	Ans. Correct (%)
Traditional RAG	24.24
Agentic w. keyword search tools ⌞pdfgrep,rga,linux cmd	32.71
Golden Evidence + GPT-5-mini ⌞oracle retrieval (full page)	94.00
Ours (AgenticRAG) GPT-5-mini ⌞search, find, open, summ.	**92.00**
Ours (AgenticRAG) Claude Sonnet 4.5 ⌞search, find, open, summ.	**91.78**

swering over real-world financial filings. The retrieval corpus consists of 84 long financial documents averaging ∼116K tokens each (∼140 pages per PDF). Our agentic approach with GPT-5-mini achieves **92.00%** correctness, substantially outperforming both traditional RAG (by **3.8**×) and moreover, is more general than the agentic tool use baseline of (Subramanian et al., 2026) (by **2.8**×), which relies on keyword search tools like pdfgrep, rga, and Linux commands. We also adopt a baseline where the ground-truth full-page evidence is provided directly to GPT-5-mini, bypassing agentic retrieval entirely. This oracle setting achieves 94.00% and establishes an upper bound on the model's reasoning ability given perfect evidence. Our agentic system is within 2 pp of this upper bound which demonstrates its effectiveness. Between GPT-5-mini vs Claude Sonnet 4.5, our harness is equally effective.

5.4 Token Cost and Retrieval Efficiency

Table 4 quantifies the end-to-end token cost of agentic retrieval. We measure total tokens consumed across the full interaction, including model thinking, tool-call arguments, retrieved tool results, and final answer generation. On BRIGHT, AgenticRAG averages 52.3K total tokens per query, compared to 20.4K for Single-shot Search, a 2.6× token overhead. This cost yields a disproportionate quality gain: Claude Sonnet 4.5 with AgenticRAG reaches 49.6% recall@1, compared to 8.41% for Single-shot Search, a 5.9× improvement. FinanceBench is more expensive, averaging 114.8K tokens per query and a 7.8× ratio over single-shot search, which reflects deep navigation over long financial filings. This higher cost is paired with 92.00% answer correctness, close to the 94.00% oracle evidence upper bound. Tool usage in Table 5 further shows that the system operates well within the 15-iteration budget, averaging 4.48–4.79 tool calls per query. The multi-query ablation provides a direct efficiency comparison: the full system achieves comparable recall with 4.79 average tool calls versus 6.79 without multi-query search, a 29% reduction in tool calls.

5.5 Ablation Studies

Single-shot vs Agentic Retrieval. From Table 5 the most significant finding is the dramatic improvement from single-shot search to full agentic tool use. Single-shot search achieves only 8.41% recall@1 on average, while agentic tool use reaches **43.49%** with GPT-5-mini and **49.59%** with Claude Sonnet 4.5—representing **5.2**× and **5.9**× improvements respectively. Notably, the proprietary search stack behind our search tool trades off raw retrieval quality for speed, immense scale, and availability compared to the state-of-the-art embedding-based retrievers in Table 2. However, these quality differences vanish when our agentic harness is employed with a reasoning language model. The improvement is consistent across splits (ref. Table 10).

Model Comparison and Tool Usage Patterns. Claude Sonnet 4.5 achieves a **+6.1 pp** improvement over GPT-5-mini, outperforming on seven of eight splits (detailed per-split results in Appendix Table 10). The two models exhibit distinct strategies that reflect an exploration–exploitation trade-off. Claude favors *exploitation*: it uses fewer search calls (2.51 vs 3.39) but opens more documents (1.54 vs 1.22) and relies more on semantic find (0.42 vs 0.14, a 3× increase), going deeper into candidate documents. GPT-5-mini favors *exploration*: it issues more search calls with reformulated queries rather than using in-document find, casting a wider net across the corpus. In the BRIGHT long-document setting, where queries have only ∼1.9 golden documents on average amid a large corpus, relevant documents are sparse and broad exploration often surfaces irrelevant results.

Failure Patterns. The main observed weakness is broad multi-evidence retrieval, especially the Pony split, where each query has ∼6.9 gold documents on average compared to ∼1.9 across BRIGHT overall. This setting rewards recovering many related documents, whereas our harness is optimized for coarse-to-fine navigation toward a small number of high-value evidence sources. This explains why Pony remains difficult for both of our models despite large gains on scientific and tech-

Table 4: Total token usage comparison between AgenticRAG and Single-shot Search across BRIGHT splits and FinanceBench. All values are averages per query (in thousands) and include system prompt, tool calls, tool results, and any thinking tokens. Cost ratio is AgenticRAG total tokens divided by Single-shot Search total tokens.

		Avg. Total Tokens (K)		Cost
Dataset	N	AgenticRAG	Single-shot	Ratio
Biology	103	49.2	23.0	2.1×
Earth Science	116	52.9	18.7	2.8×
Economics	103	44.6	21.3	2.1×
Psychology	101	58.2	24.8	2.3×
Robotics	101	56.2	23.1	2.4×
Stack Overflow	117	55.7	15.8	3.5×
Sustainable Living	108	59.3	19.9	3.0×
Pony	112	42.6	17.6	2.4×
BRIGHT Avg.	**861**	**52.3**	**20.4**	**2.6×**
FinanceBench	150	114.8	14.7	7.8×

Table 5: Ablation study of agentic components averaged across all BRIGHT splits. Performance is measured by recall (R@k), along with average tool usage and per-tool statistics. Per-split results are in Table 10.

Variant	R@1	R@3	Avg. Tools	Search	Open	Find	Summ.
Single-shot Search	8.41±4.83	12.90±5.87	1	1	-	-	-
Claude Sonnet 4.5	49.59±7.79	64.20±7.13	4.48±0.26	2.51±0.19	1.54±0.15	0.42±0.11	0.01±0.02
GPT-5-mini	43.49±8.00	62.53±7.23	4.79±0.71	3.39±0.55	1.22±0.25	0.14±0.08	0.06±0.05
∟ w/o Summarize	43.34±8.07	63.85±7.21	4.92±0.70	3.44±0.54	1.31±0.27	0.16±0.09	-
∟ w/o Semantic Find	46.34±7.97	64.44±7.07	5.02±0.73	3.47±0.55	1.34±0.27	0.17±0.09	0.06±0.05
∟ w/o Multi-query Search	44.84±8.08	62.30±7.26	6.79±0.70	4.38±0.57	2.16±0.27	0.24±0.11	0.03±0.04

nical splits where relevant documents are sparse. The pattern suggests that future trajectory policies should better detect broad evidence needs and shift from depth-first document reading to wider coverage before final ranking.

Component Contributions. We ablate individual components using GPT-5-mini to understand their contributions. The most notable finding concerns multi-query search. In the default configuration, the model can issue up to 5 queries in parallel within a single search tool call, with results de-duped and presented together. Restricting the system to single-query search (**w/o Multi-query Search**, 44.84%), where the model issues only one query per search call but receives the same number of results, achieves comparable recall@1 to the full system but at the cost of increased tool usage—6.79 average tool calls compared to 4.79 for the full system, with notably more search operations (4.38 vs 3.39) and document opens (2.16 vs 1.22). This suggests that multi-query search improves efficiency by finding relevant documents with fewer iterations. Detailed analysis of these ablations is provided in Appendix C.2.

Findings from Pre-Production Deployments In our pre-production evaluation we identified several design choices that guide the model toward more optimal trajectories: (1) **Surfacing document metadata in search results** like title, filename, and file type helps the model disambiguate semantically similar snippets and avoid redundant searches; (2) **Line-numbered document previews** lets the model anchor on specific content and jump to relevant sections in successive open calls; (3) **Candidate reference retention after summarization** in the context window enables the model to go deeper on promising candidates via open/find, rather than restarting retrieval with reformulated queries.(4) **Having a switcher** route complex, multi-intent queries to our agentic rag harness for deeper analysis. Simple queries are routed to traditional rag for faster answers. This is vital for the tradeoff between user experience, cost, model availability. We have good early signals of this being effective and we continue to pursue this hybrid approach.

6 Conclusion

We presented a practical harness for AgenticRAG that equips reasoning language models with search,

find, open, and summarize tools to autonomously retrieve and reason over large enterprise corpora. Across three benchmarks, our approach achieves 49.6% recall@1 on BRIGHT (+21.8 pp over the best embedding baseline), 0.96 factuality on WixQA (+13% relative), and 92.00% answer correctness on FinanceBench—within 2 pp of oracle access. Token analysis shows that these gains require a moderate 2.6× token overhead on BRIGHT relative to single-shot search, while delivering a 5.9× recall@1 improvement. These results demonstrate that our harness effectively extracts the value of reasoning models for enterprise information retrieval tasks requiring deep, multi-step reasoning. Future work will focus on large-scale deployment, budget-aware routing between traditional and agentic RAG, deeper failure analysis, ablations over iteration and window-size budgets, and optimizing retrieval trajectories for fast iterative reasoning via fine tuning.

Acknowledgments

We thank Eli Coon, Kinfe Mengistu and members of the broader Copilot Studio team for feedback and discussions during internal experimentations. Special thanks to James Cai and Alejandro Gutierrez Munoz for technical guidance and project sponsorship.

References

Akari Asai, Zeqiu Wu, Yizhong Wang, Avirup Sil, and Hannaneh Hajishirzi. 2024. Self-rag: Learning to retrieve, generate, and critique through self-reflection. In *International Conference on Learning Representations (ICLR)*.

AzureAISearch. Agentic Retrieval - Azure AI Search — learn.microsoft.com. https://learn.microsoft.com/en-us/azure/search/agentic-retrieval-overview. [Accessed 14-02-2026].

Sebastian Borgeaud, Arthur Mensch, Jordan Hoffmann, Trevor Cai, Eliza Rutherford, Katie Millican, George Bm Van Den Driessche, Jean-Baptiste Lespiau, Bogdan Damoc, Aidan Clark, and 1 others. 2022. Improving language models by retrieving from trillions of tokens. In *International Conference on Machine Learning (ICML)*, pages 2206–2240.

X Chen and 1 others. 2024. Hiqa: A hierarchical contextual augmentation rag for multi-documents qa. *arXiv preprint arXiv:2402.12345*.

Dvir Cohen, Lin Burg, Sviatoslav Pykhnivskyi, Hagit Gur, Stanislav Kovynov, Olga Atzmon, and Gilad Barkan. 2025. Wixqa: A multi-dataset benchmark for enterprise retrieval-augmented generation. *arXiv preprint arXiv:2505.08643*.

Darren Edge, Ha Trinh, Nuo Cheng, Joshua Bradley, Alex Chao, Apurva Mody, Steven Truitt, 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.

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

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Ming-Wei Chang. 2020. Realm: Retrieval-augmented language model pre-training. In *International Conference on Machine Learning (ICML)*, pages 3929–3938.

Pranab Islam, Anand Kannappan, Douwe Kiela, Rebecca Qian, Nino Scherrer, and Bertie Vidgen. 2023. Financebench: A new benchmark for financial question answering. *arXiv preprint arXiv:2311.11944*.

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.

Soyeong Jeong, Jinheon Baek, Sukmin Cho, Sung Ju Hwang, and Jong C Park. 2024. Adaptive-rag: Learning to adapt retrieval-augmented large language models through question complexity. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*.

Zhengbao Jiang, Frank F Xu, Jun Araki, and Graham Neubig. 2023. Active retrieval augmented generation. *arXiv preprint arXiv:2305.06983*.

Bowen Jin and 1 others. 2025. Search-r1: Training llms to reason and leverage search engines with reinforcement learning. *arXiv preprint arXiv:2503.09516*.

Omar Khattab and Matei Zaharia. 2020. Colbert: Efficient and effective passage search via contextualized late interaction over bert. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 39–48.

M Lee and 1 others. 2024. Planrag: A plan-then-retrieval augmented generation for generative large language models as decision makers. In *Proceedings*

of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics (NAACL).

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, and 1 others. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 33, pages 9459–9474.

Xiaoxue Li and 1 others. 2025. Search-o1: Agentic search-enhanced large reasoning models. *arXiv preprint arXiv:2501.05366*.

Tie-Yan Liu and 1 others. 2009. Learning to rank for information retrieval. *Foundations and Trends® in Information Retrieval*, 3(3):225–331.

Alex Mallen, Akari Asai, Victor Zhong, Rajarshi Das, Daniel Khashabi, and Hannaneh Hajishirzi. 2023. When not to trust language models: Investigating effectiveness of parametric and non-parametric memories. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 9802–9822.

Rodrigo Nogueira and Kyunghyun Cho. 2019. Passage re-ranking with bert. *arXiv preprint arXiv:1901.04085*.

Agada Joseph Oche, Ademola Glory Folashade, Tirthankar Ghosal, and Arpan Biswas. 2025. A systematic review of key retrieval-augmented generation (rag) systems: Progress, gaps, and future directions. *arXiv preprint arXiv:2507.18910*.

Jeff Z Pan and 1 others. 2024. Unifying large language models and knowledge graphs: A roadmap. *IEEE Transactions on Knowledge and Data Engineering*.

Ofir Press, Muru Zhang, Sewon Min, Ludwig Schmidt, Noah A Smith, and Mike Lewis. 2023. Measuring and narrowing the compositionality gap in language models. *arXiv preprint arXiv:2210.03350*.

Ori Ram, Yoav Levine, Itay Dalmedigos, Doron Schuhmann, Gadi Sasho, Erez Karpas, Ori Shwartz-Ziv, Nitish Gupta, Yasheng Wu, Kevin Leyton-Brown, and 1 others. 2023. In-context retrieval-augmented language models. *arXiv preprint arXiv:2302.00083*.

Parth Sarthi, Salman Abdullah, Aditi Tuli, Shubh Khanna, Anna Goldie, and Christopher D. Manning. 2024. Raptor: Recursive abstractive processing for tree-organized retrieval. In *International Conference on Learning Representations (ICLR)*.

H Scaffidi and 1 others. 2025. Graphrag on technical documents - impact of knowledge graph schema. *Transactions on Graph Data and Knowledge*.

Timo Schick, Jane Dwivedi-Yu, Roberto Dessì, Roberta Raileanu, Maria Lomeli, Luke Zettlemoyer, Nicola Cancedda, and Thomas Scialom. 2023. Toolformer: Language models can teach themselves to use tools. In *Advances in Neural Information Processing Systems (NeurIPS)*.

Weijia Shi, Sewon Min, Michihiro Yasunaga, Minjoon Seo, Rich James, Mike Lewis, Luke Zettlemoyer, and Wen-tau Yih. 2023. Replug: Retrieval-augmented black-box language models. *arXiv preprint arXiv:2301.12652*.

Aditi Singh, Abul Ehtesham, Saket Kumar, and Tala Talaei Khoei. 2025. Agentic retrieval-augmented generation: A survey on agentic rag. *arXiv preprint arXiv:2501.09136*.

Hongjin Su, Howard Yen, Mengzhou Xia, Weijia Shi, Niklas Muennighoff, Han-yu Wang, Haisu Liu, Quan Shi, Zachary S Siegel, Michael Tang, and 1 others. 2024. Bright: A realistic and challenging benchmark for reasoning-intensive retrieval. *arXiv preprint arXiv:2407.12883*.

Shreyas Subramanian, Wale Akinfaderin, Yanyan Zhang, Ishan Singh, Chris Pecora, Mani Khanuja, Sandeep Singh, and Maira Ladeira Tanke. 2026. Keyword search is all you need: Achieving rag-level performance without vector databases using agentic tool use.

Nandan Thakur, Nils Reimers, Andreas Rücklé, Abhishek Srivastava, and Iryna Gurevych. 2021. Beir: A heterogenous benchmark for zero-shot evaluation of information retrieval models. *arXiv preprint arXiv:2104.08663*.

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.

Liang Wang, Nan Yang, and Furu Wei. 2023. Query2doc: Query expansion with large language models. *arXiv preprint arXiv:2303.07678*.

X Wang and 1 others. 2024. Knowledge graph-enhanced retrieval-augmented generation. *arXiv preprint arXiv:2402.12345*.

Shi-Qi Yan, Jia-Chen Gu, Yun Zhu, and Zhen-Hua Ling. 2024. Corrective retrieval augmented generation. *arXiv preprint arXiv:2401.15884*.

Shunyu Yao, Jeffrey Zhao, Dian Yu, Nan Du, Izhak Shafran, Karthik Narasimhan, and Yuan Cao. 2023. React: Synergizing reasoning and acting in language models. In *International Conference on Learning Representations (ICLR)*.

A Method Details

A.1 Agentic Loop Algorithm

Detailed agentic loop algorithm is shown in Algorithm 1.

Algorithm 1 Agentic Loop

Require: user_query, max_calls, token_threshold
Ensure: Formatted answer with citations

```
conversation.add(user_query)
for i = 1 to max_calls do
    if tokens(conversation) ≥ token_threshold then
        MANAGECONTEXT( )                ▷ Force summarize
    end if
    response ← LLM(conversation, tool_schemas)
    if response.has_tool_calls then
        for each tool_call in response.tool_calls do
            result ← EXECUTETOOL(tool_call)
            conversation.add(tool_call, result)
        end for
    else
        return FORMATANSWER(response.text)
    end if
end for
return FORCEFINALANSWER( )
```

A.2 System Instructions for Tool Use

Overall instructions include:

- Search before answering when uncertain.
- Progressively explore using find or open when snippets are insufficient.
- Reuse previous results rather than performing search again.
- Cite every time when information is used from tool outputs.

When to use search:

- Primary search tool across enterprise corpus.
- First choice for any work-related query.
- When users reference current/changing information, enterprise-specific terms, or acronyms.
- To verify details rather than making assumptions.

When to use find:

- In-document pattern search for relevant files from search results.
- When search results do not give enough details.
- To get a focused view of a result in relation to certain terms.

When to use open:

- Windowed full content retrieval for relevant files from search results.
- When search results snippets are insufficient.
- To pull in more content from the most promising results.
- Can open multiple search results.
- Option to choose a line number close to the relevant content.

B Dataset Details

B.1 BRIGHT Benchmark

We adopt the BRIGHT benchmark (Su et al., 2024), which is designed to capture realistic enterprise scenarios of information retrieval. BRIGHT derives queries from StackExchange posts, reflecting human-authored, highly situational and domain-specific information needs. For each query, the corpus contains positive documents cited in top-voted answers and verified by human annotators, as well as negative documents collected via search engine retrieval. The corpora is normalized web content (e.g., Wikipedia pages, blogs, and reports). This construction has shown to yields realistic retrieval pools with substantial semantic overlap between relevant and irrelevant documents. We choose to evaluate on the long-context setting of BRIGHT, where documents correspond to entire web pages rather than snippets and the task is to retrieve the full relevant document(s) for a given query. Our experiments span eight domains: Biology, Earth Science, Economics, Psychology, Robotics, Stack Overflow, Sustainable Living, and Pony. These domains cover a broad range of scientific, technical, and professional areas commonly encountered in enterprise information retrieval. Across domains, corpora contain hundreds to thousands of documents, with average document lengths ranging from several thousand to over 40k tokens. Queries themselves are also non-trivial in length, with average query sizes exceeding 100 tokens in most domains and reaching several hundred tokens in technical domains such as Robotics and Stack Overflow. Benchmark statistics are detailed in Table 6. We follow the standard evaluation protocol of the BRIGHT benchmark and report Recall@1 for long-context document retrieval.

B.2 WixQA Benchmark

WixQA targets procedural, long-form queries that require multi-step reasoning and specialized enterprise vocabulary, closely matching real-world support and troubleshooting scenarios. We utilize both the subsets in WiXQA for our experiments: *Expert Written*, containing authentic customer queries with step-by-step answers authored and validated by human domain experts, and *Simulated*, derived from multi-turn user–chatbot interactions and curated into single-turn queries with expert-validated procedural correctness. A defining characteristic of WixQA is its multi-article dependency, where an-

Table 6: Dataset statistics for the BRIGHT benchmark (Su et al., 2024) long-context splits used in our evaluation.

Split	# Query	# Docs	Avg Query Len	Avg Doc Len	Avg # Gold Docs
Biology	103	524	115.2	9,422.4	1.3
Earth Science	116	601	109.5	27,312.3	1.6
Economics	103	516	181.5	11,896.4	1.1
Psychology	101	512	149.6	12,411.7	1.1
Robotics	101	508	818.9	14,998.2	1.1
Stack Overflow	117	1,858	478.3	40,759.7	1.1
Sustainable Living	108	554	148.5	12,077.7	1.2
Pony	112	577	102.6	1,361.0	6.9
Total/Avg	861	5,650	263.0	16,279.9	1.9

Table 7: Dataset statistics for the WixQA (Cohen et al., 2025) benchmark (median values).

Dataset	# Query	# Query Tokens	# Answer Tokens	Multi-Article %
ExpertWritten	200	19	172	27%
Simulated	200	12	50	14%

swering a query may require retrieving and synthesizing information from multiple documents. All queries are grounded in a shared enterprise-scale knowledge base of 6,221 domain-specific help articles, making WixQA well suited for evaluating agentic RAG that must coordinate retrieval and reasoning over complex, multi-document enterprise corpora. Datasets statistics are presented in 7

B.3 FinanceBench

Table 8: FinanceBench (Islam et al., 2023) data statistics.

Statistic	Value
# Queries	150
# Ground docs	84
Avg. # pages / doc	143
Avg. # tokens / doc	116,715
Total tokens	9,804,065

FinanceBench (Islam et al., 2023) is a human-evaluated benchmark consisting of financial questions over public company filings (10-K, 10-Q, 8-K, and earnings reports in PDF form). Questions span metrics-generated and domain-relevant categories: metrics-generated questions target specific financial line items or ratios that require the model to locate the relevant data in the document and often perform a calculation, making them straightforward to verify since each has a single unambiguous answer. Domain-relevant questions require deeper financial reasoning, such as identifying drivers of margin changes or assessing capital intensity. Each query pertains to a single document. We choose this benchmark because of the large size of its documents (averaging $\sim$143 pages and $\sim$117K tokens per PDF), which is representative of enterprise domains where knowledge workers routinely work with dense, information-heavy manuals, reports, and regulatory filings. Corpus statistics are presented in Table 8. The evaluation metric we use is answer correctness. LLM as a judge is used for this process and manual review of the results is also conducted.

C Additional Results

C.1 Full BRIGHT Retrieval Results

Table 9 presents the complete retrieval results on the BRIGHT benchmark across all baseline models.

C.2 Detailed Ablation Analysis

Table 10 provides per-split ablation results across all BRIGHT domains. Removing the summarization tool (**w/o Summarize**, 43.34% avg recall@1) has minimal impact, indicating that this component is rarely needed for the retrieval task. Removing semantic find (**w/o Semantic Find**, 46.34%) slightly *improves* average recall@1, likely because the lexical find fallback is sufficient for most in-document searches and removing the semantic option reduces latency, allowing more search iterations within the same compute budget.

C.3 WixQA Simulated Results

As shown in Figure. 4, on simulated questions with expert validated ground truth answers i.e. *Simulated* split of WixQA, our method achieves **0.94** factuality, compared to 0.77 for both E5+GPT-4o

Table 9: Full long-context retrieval performance on unsplit web pages of StackExchange data from BRIGHT benchmark. Scores are reported in recall@1.

	Bio.	Earth.	Econ.	Psy.	Rob.	Stack.	Sus.	Pony	Avg.
Sparse models									
BM25	10.7	15.4	10.7	8.4	7.4	22.2	10.7	5.4	11.4
Open-sourced embedding models									
BGE	16.4	27.7	20.9	11.6	10.9	13.3	16.9	0.4	14.8
Inst-L	24.6	29.9	13.1	20.3	12.9	15.0	25.4	3.9	18.1
SBERT	25.6	34.1	18.9	15.8	10.9	15.0	18.0	1.2	17.4
E5	29.9	36.3	26.2	46.7	17.3	14.5	32.2	1.1	25.5
SFR	30.3	37.0	24.3	47.7	17.3	14.5	35.0	2.0	26.0
Inst-XL	21.5	31.0	13.1	20.5	13.9	15.0	20.1	6.0	17.6
GritLM	37.5	40.3	25.7	34.4	17.8	20.1	32.4	0.0	26.0
Qwen	39.2	36.1	25.7	42.3	21.3	23.5	33.1	1.3	27.8
Proprietary embedding models									
Cohere	31.5	34.5	18.9	20.5	9.9	15.8	15.2	0.8	18.4
OpenAI	32.1	31.4	23.8	34.2	11.9	10.7	26.3	0.0	21.3
Voyage	34.4	35.4	26.7	41.6	12.9	12.8	31.1	1.3	24.5
Google	30.9	38.0	21.9	30.7	12.9	19.2	25.7	0.3	22.4
Reasoning enhanced methods									
Claude-3-Opus (BM25)	26.8	13.5	13.4	28.2	7.9	28.2	11.8	–	18.5
GPT-4 (BM25)	26.8	15.8	10.2	30.7	5.9	26.5	9.7	–	17.9
DeepSeek-R1 (BM25)	26.8	20.0	14.4	30.2	14.9	33.3	10.6	–	21.5
ReDI (BM25)	28.4	22.4	21.2	32.0	19.8	36.3	21.7	–	26.0
Claude-3-Opus (SBERT)	34.8	31.6	21.8	15.8	8.9	15.8	16.6	–	20.8
GPT-4 (SBERT)	37.7	35.3	19.9	18.3	12.4	11.5	22.6	–	22.5
DeepSeek-R1 (SBERT)	35.6	34.8	16.0	15.3	8.9	15.0	19.9	–	20.8
ReDI (SBERT)	36.2	32.8	22.8	20.8	10.9	16.2	22.2	–	23.1
Our Agentic methods (search/find/open)									
GPT-5-mini	61.7	48.1	41.4	65.3	39.4	**40.6**	46.6	4.8	43.5
Claude Sonnet 4.5	**62.3**	**60.0**	**58.7**	**67.9**	**55.0**	34.1	**51.7**	**7.1**	**49.6**

Table 10: Per-split ablation study of agentic components on BRIGHT. Performance is measured by recall (R@k), along with average tool usage and per-tool usage statistics.

Variant	R@1	R@3	Avg. Tools	Search	Open	Find	Sum.
Bio.							
Single-shot Search	10.88±5.70	14.63±6.55	1	1	-	-	-
Claude Sonnet 4.5	**62.34±8.33**	80.47±7.19	4.36±0.28	2.12±0.20	1.72±0.14	0.52±0.12	0.01±0.02
GPT-5-mini	61.72±8.66	88.28±5.53	4.54±0.78	3.46±0.60	0.98±0.25	0.09±0.06	0.02±0.03
∟ w/o Summarize	58.16±8.59	83.33±6.68	4.52±0.80	3.38±0.57	1.06±0.30	0.07±0.05	-
∟ w/o Semantic Find	59.43±8.42	86.53±6.31	4.49±0.71	3.32±0.57	1.11±0.28	0.04±0.04	0.02±0.03
∟ w/o Multi-query	60.35±8.86	85.44±6.40	6.63±0.80	4.34±0.64	2.20±0.30	0.09±0.07	-
Earth.							
Single-shot Search	7.35±4.48	13.62±6.09	1	1	-	-	-
Claude Sonnet 4.5	**60.01±7.38**	78.51±5.99	4.54±0.25	2.28±0.19	1.70±0.14	0.54±0.12	0.01±0.02
GPT-5-mini	48.10±8.02	72.06±7.30	5.06±0.78	3.52±0.60	1.36±0.27	0.12±0.07	0.07±0.05
∟ w/o Summarize	48.44±8.33	72.66±6.93	4.88±0.69	3.36±0.55	1.41±0.27	0.11±0.07	-
∟ w/o Semantic Find	56.10±7.87	75.93±6.25	4.99±0.74	3.41±0.56	1.38±0.28	0.14±0.09	0.06±0.04
∟ w/o Multi-query	54.43±7.73	75.00±6.61	6.81±0.71	4.30±0.58	2.28±0.27	0.20±0.10	0.03±0.04
Econ.							
Single-shot Search	17.39±7.61	21.74±8.15	1	1	-	-	-
Claude Sonnet 4.5	**58.74±9.47**	68.28±8.74	4.50±0.24	2.37±0.15	1.62±0.16	0.50±0.13	0.01±0.02
GPT-5-mini	41.41±9.60	67.68±9.09	3.65±0.69	2.52±0.51	1.04±0.27	0.06±0.06	0.03±0.03
∟ w/o Summarize	45.96±9.85	72.22±8.84	4.07±0.79	3.05±0.63	0.90±0.26	0.12±0.10	-
∟ w/o Semantic Find	46.00±9.50	67.50±9.25	3.97±0.77	2.99±0.58	0.86±0.25	0.05±0.05	0.07±0.07
∟ w/o Multi-query	42.27±9.54	67.53±9.28	6.32±0.70	4.00±0.59	2.20±0.29	0.12±0.07	-
Psy.							
Single-shot Search	4.88±4.27	6.95±5.24	1	1	-	-	-
Claude Sonnet 4.5	**67.86±8.93**	83.57±6.99	4.08±0.24	2.08±0.17	1.62±0.15	0.37±0.11	0.01±0.02
GPT-5-mini	65.26±8.95	78.84±7.89	3.87±0.68	2.82±0.54	0.86±0.23	0.07±0.06	0.12±0.07
∟ w/o Summarize	58.95±9.47	81.26±7.63	3.97±0.65	2.89±0.50	1.03±0.27	0.04±0.04	-
∟ w/o Semantic Find	65.26±9.21	83.89±6.89	4.36±0.76	2.89±0.53	1.29±0.31	0.05±0.05	0.12±0.07
∟ w/o Multi-query	67.06±9.42	80.04±8.04	6.51±0.77	4.08±0.65	2.27±0.33	0.13±0.08	0.02±0.03
Rob.							
Single-shot Search	10.20±6.12	15.31±7.14	1	1	-	-	-
Claude Sonnet 4.5	**54.95±9.41**	71.29±8.66	4.92±0.33	2.97±0.22	1.54±0.16	0.41±0.12	-
GPT-5-mini	39.39±9.34	60.61±9.34	5.22±0.75	3.65±0.56	1.44±0.29	0.13±0.09	-
∟ w/o Summarize	39.80±9.18	63.27±9.44	5.13±0.69	3.54±0.56	1.46±0.29	0.13±0.07	-
∟ w/o Semantic Find	43.68±9.74	65.26±9.21	5.63±0.78	3.78±0.59	1.61±0.30	0.22±0.12	0.02±0.03
∟ w/o Multi-query	46.94±9.69	65.82±9.44	6.95±0.68	4.62±0.58	2.09±0.25	0.24±0.10	-
Stack.							
Single-shot Search	7.28±4.61	12.14±6.07	1	1	-	-	-
Claude Sonnet 4.5	34.05±8.19	43.53±8.62	4.76±0.28	3.31±0.26	1.07±0.15	0.36±0.11	0.02±0.02
GPT-5-mini	40.62±8.71	53.12±8.93	6.03±0.76	4.88±0.65	0.85±0.18	0.25±0.15	0.05±0.05
∟ w/o Summarize	37.84±8.56	54.95±9.01	6.71±0.75	5.11±0.64	1.26±0.21	0.34±0.15	-
∟ w/o Semantic Find	**41.52±8.71**	54.02±8.93	6.62±0.78	5.20±0.66	1.18±0.21	0.19±0.10	0.05±0.04
∟ w/o Multi-query	33.77±8.55	43.42±8.77	7.93±0.71	6.25±0.63	1.24±0.19	0.43±0.21	-
Sus.							
Single-shot Search	8.87±5.51	18.28±7.26	1	1	-	-	-
Claude Sonnet 4.5	51.65±8.97	69.15±8.19	4.48±0.21	2.32±0.15	1.72±0.14	0.44±0.11	-
GPT-5-mini	46.65±9.19	73.45±8.13	5.01±0.65	2.98±0.45	1.88±0.33	0.09±0.06	0.06±0.05
∟ w/o Summarize	53.25±9.22	76.91±7.54	5.17±0.71	3.13±0.47	1.91±0.34	0.13±0.10	-
∟ w/o Semantic Find	**55.53±9.16**	75.83±7.96	5.39±0.67	3.14±0.45	2.02±0.33	0.15±0.09	0.08±0.06
∟ w/o Multi-query	50.53±9.54	76.47±8.02	7.25±0.61	4.06±0.51	2.92±0.27	0.23±0.10	0.04±0.05
Pony							
Single-shot Search	0.40±0.35	0.52±0.46	1	1	-	-	-
Claude Sonnet 4.5	**7.12±1.63**	18.79±2.67	4.22±0.29	2.62±0.19	1.36±0.12	0.25±0.10	-
GPT-5-mini	4.79±1.54	6.20±1.63	4.93±0.58	3.31±0.45	1.35±0.22	0.27±0.11	-
∟ w/o Summarize	4.34±1.38	6.16±1.63	4.89±0.56	3.09±0.42	1.45±0.21	0.35±0.18	-
∟ w/o Semantic Find	3.19±1.18	6.58±1.76	4.72±0.63	3.00±0.44	1.22±0.22	0.50±0.17	-
∟ w/o Multi-query	3.34±1.27	4.66±1.51	5.92±0.63	3.35±0.41	2.07±0.23	0.50±0.17	-

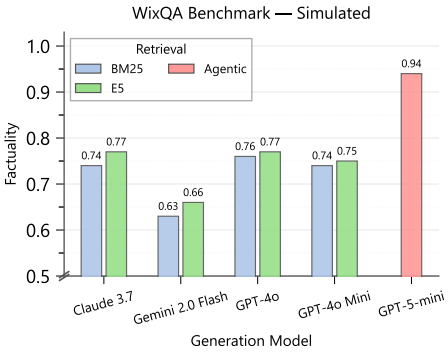

Figure 4: Factuality performance on the WixQA Simulated dataset. The performance gap between agentic retrieval and traditional methods is even larger on synthetic questions that require more complex reasoning.

and E5+Claude 3.7. The improvement is even more pronounced on this dataset, with a **22%** relative gain. This suggests that agentic retrieval is particularly effective when questions require more complex reasoning or multi-hop information gathering.

C.4 Example Conversation

Figure 5 shows an example conversation from the FinanceBench.

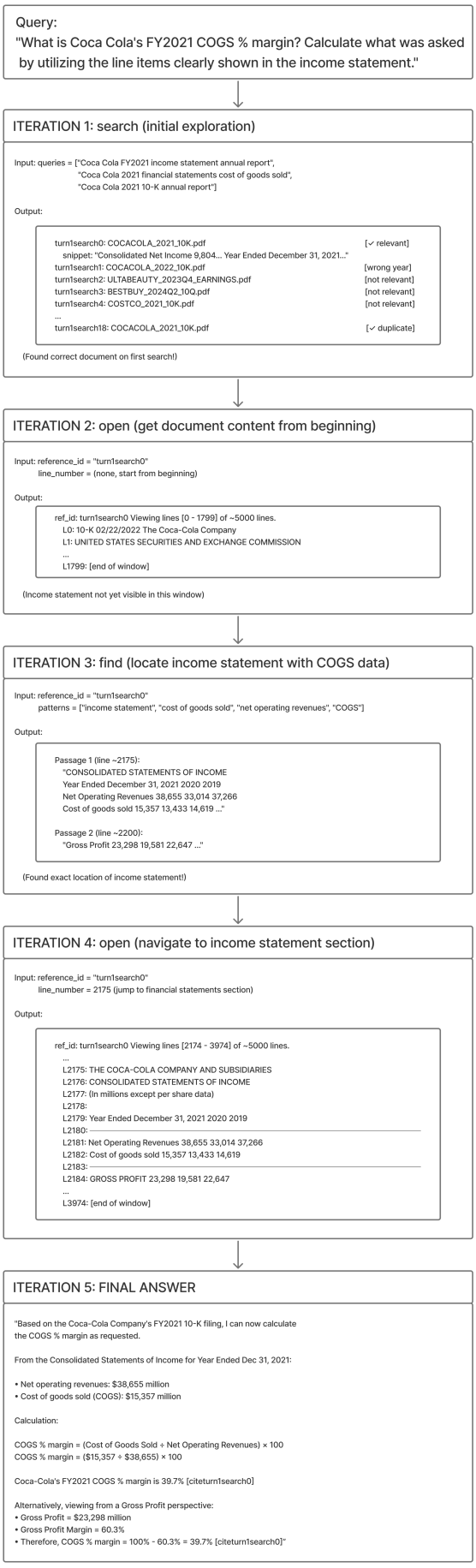

Figure 5: Example conversation from FinanceBench.