Optimizing Retrieval-Augmented Generation with Multi-Agent Hybrid Retrieval

Hediyeh Baban[*]
Hediyeh_Ledbetter@Dell.com
Dell Technologies
Austin, TX, USA

Sai Abhishek Pidaparthi[*]
sai_abhishek_pidapar@Dell.com
Dell Technologies
Austin, TX, USA

Samaksh Gulati[*]
samaksh_gulati@Dell.com
Dell Technologies
Austin, TX, USA

Aashutosh Nema[*]
Aashutosh_Nema@Dell.com
Dell Technologies
Austin, TX, USA

Abstract

With the rapid growth of digital content and scientific literature, efficient information retrieval is increasingly vital for research automation, document management, and question answering. Traditional retrieval methods like BM25 and embedding-based search, though effective individually, often fall short on complex queries.

We propose an Agentic AI workflow for Retrieval-Augmented Generation (RAG), integrating hybrid retrieval with multi-agent collaboration. Our system combines BM25 and semantic search, ensembles results via weighted cosine similarity, and applies contextual reordering using large language models.

The workflow is powered by LangGraph, a multi-agent framework enabling dynamic agent coordination for document ranking and filtering. Experiments show a 4× reduction in retrieval latency (43s to 11s) and a 7% improvement in relevance accuracy. We also analyze weight sensitivity and discuss scalability and limitations, paving the way for future enhancements.

ACM Reference Format:
Hediyeh Baban[*], Sai Abhishek Pidaparthi[*], Samaksh Gulati[*], and Aashutosh Nema[*]. 2025. Optimizing Retrieval-Augmented Generation with Multi-Agent Hybrid Retrieval. In *Proceedings of Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD '25).* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3690624

1 Introduction

Information retrieval (IR) is foundational to AI applications like research automation, customer support, and intelligent document summarization. Traditional methods—keyword-based retrieval (e.g., BM25) and semantic search via embeddings—perform well in isolation but often fail with complex or multi-faceted queries.

Retrieval-Augmented Generation (RAG) enhances large language models (LLMs) by grounding responses in retrieved evidence. However, current RAG systems typically rely on either lexical or semantic retrieval, each with its own limitations: keyword systems may miss contextual meaning, while embedding-based methods can overlook exact matches and domain-specific terminology.

To address these issues, we introduce a hybrid RAG framework powered by an Agentic AI workflow using LangGraph. Our system integrates BM25 and semantic search with agent-based coordination for contextual reordering, enabling dynamic adaptation to varying query types while reducing latency and improving relevance.

We benchmark our system on a curated academic dataset, demonstrating a 4× improvement in retrieval time and a 7% gain in Top-10 accuracy over standard hybrid baselines. This study provides a comparative analysis of traditional and agentic hybrid workflows, highlighting gains in efficiency, accuracy, and scalability.

The remainder of this paper is structured as follows: Section 2 reviews related work; Section 3 presents the methodology and agent communication protocols; Section 4 details the experimental setup; Section 6 reports the results and ablation studies; Section **??** discusses limitations and implications; and Section 7 concludes with future directions.

2 Related Work

Retrieval-Augmented Generation (RAG) has emerged as a powerful technique for enhancing language model performance by grounding responses in retrieved evidence. Traditional retrieval methods such as BM25 [12] have increasingly been complemented by deep embedding-based semantic search [11], and hybrid retrieval systems that combine both lexical and semantic signals have shown improvements in accuracy and contextual relevance [5, 8]. More recent work has expanded this space through innovations like Agentic RAG, where Prantikos et al. (2024) introduced multi-agent collaboration for efficient retrieval and contextual ranking [10]; secure frameworks to mitigate hallucination and leakage risks, as proposed by Mehta and Patel (2024) [9]; and hybrid systems that incorporate both textual and structured knowledge sources, such as HybGRAG by Lee et al. (2024) [7]. Optimized domain-specific pipelines have also emerged, including real-time clinical QA systems [4], graph-integrated retrieval models like CAPRAG [6], and

[*]All authors contributed equally to this research.

KDD '25, Toronto, ON, Canada
© 2025 Copyright held by the owner/author(s). Publication rights licensed to ACM.
ACM ISBN 979-8-4007-1245-6
https://doi.org/10.1145/3690624

applications in regulatory compliance [3], medical data processing [14], and legal document search [13]. These advances reflect a growing shift toward multi-agent, secure, and hybrid RAG architectures. Our work builds on this momentum by introducing a modular, agentic hybrid retrieval system that dynamically balances BM25 and embedding-based retrieval, enhancing contextual coherence and retrieval efficiency through coordinated multi-agent orchestration.

3 Methodology

3.1 Hybrid Retrieval Framework

The hybrid retrieval framework integrates BM25 keyword search and semantic embedding-based search.

3.1.1 Keyword-Based Search (BM25). BM25 is a probabilistic ranking function derived from the TF-IDF model that assigns higher scores to documents containing rare terms [2]. It is particularly effective for:

- **Exact matches:** Identifying documents with high keyword overlap.
- **Short queries:** Queries with concise terms benefit from BM25's precision.

3.1.2 Semantic Search (Embeddings). Embedding-based search converts documents and queries into high-dimensional vector representations using pre-trained language models (e.g., Sentence-BERT). Cosine similarity is then used to measure semantic alignment.

3.1.3 Weighted Ensemble Rationale. To blend the precision of BM25 and the generalization of embedding-based retrieval, we assign a weight of 30% to BM25 scores and 70% to embedding scores. These weights were chosen based on validation accuracy across diverse query types.

We observed that:

- **Simple Queries:** BM25 performs better when the query consists of known keywords.
- **Complex Queries:** Embedding-based search excels when the query contains paraphrases, latent intent, or multiple topics.

The weighted ensemble allows the system to strike a balance between both cases, and as shown in Table 4, this configuration achieves the highest F1 score. The ensemble process is also sensitive to query complexity, allowing the system to adapt weights dynamically in future iterations.

3.1.4 Contextual Reordering with LangChain. To refine the top 1,000 retrieved documents, LangChain contextual reordering is applied [1], ensuring that the final selection aligns with user intent.

3.2 Agentic AI Workflow

In the context of this work, an "agent" refers to a modular, autonomous computational unit that is capable of executing a specific retrieval-related task, making intermediate decisions, and communicating with other agents in a larger workflow. Each agent consists of:

- **A decision policy:** A set of rules or learned logic to process input (e.g., deciding query complexity).
- **Tool invocation:** Calls to external APIs or retrieval tools such as BM25 or vector search.
- **Memory/context access:** Optionally stores or forwards intermediate metadata.

These agents operate under LangGraph orchestration but the agentic concept is not tied to a specific framework and can be abstracted to other orchestration environments.

We implemented our agentic workflow using LangGraph due to its minimal latency overhead, native support for asynchronous, parallel agent execution, and built-in modular orchestration capabilities. LangGraph's lightweight message-passing architecture allows flexible, dynamic coordination among agents, crucial for rapid retrieval tasks. While alternative frameworks such as CrewAI and AutoGen also support agent orchestration, they typically emphasize more structured task sequencing or conversational workflows, introducing additional overhead and reduced flexibility for parallel, high-throughput retrieval tasks.

Thus, LangGraph aligns optimally with our requirements for low latency, dynamic task decomposition, and real-time adaptive retrieval, making it the most suitable choice for our Agentic AI workflow. The Agentic AI workflow builds on the hybrid retrieval framework by enabling parallel, coordinated execution through a multi-agent system implemented in LangGraph. The system comprises two retrieval branches—BM25-based and embedding-based—each supported by specialized agents that process, rank, and filter results.

3.3 Retrieval Branches

Our system splits retrieval into two complementary pipelines that run in parallel. In the first pipeline, the *Keyword Retrieval Agent* issues a BM25 query to fetch an initial batch of documents; these are then refined by the *Keyword Filtering Agent*, which applies TF–IDF scores and simple domain heuristics to drop anything unlikely to address the user's information need. Simultaneously, the second pipeline uses the *Semantic Retrieval Agent* to perform a vector-based search over Sentence-BERT embeddings, capturing deeper conceptual matches; its results are subsequently re-scored by the *Semantic Filtering Agent*, which enforces cosine-similarity thresholds and topic-density checks to ensure high topical relevance.

3.4 Cross-Branch Coordination

To dynamically balance these two approaches, we introduce a *Complexity Analysis Agent* and a *Comparison Agent*. Before any retrieval begins, the Complexity Analysis Agent examines the user's query—evaluating linguistic structure and topical breadth—to assign relative weights to the keyword and embedding pipelines. Once both pipelines have produced their top candidates, the Comparison Agent aggregates their scores through a weighted ensemble strategy, selecting the final set of documents that best satisfy the overall context and intent.

3.5 Agent Communication Protocol

All agents communicate over a lightweight message-passing bus implemented in LangGraph. Each request, intermediate score, and piece of contextual metadata is sent as an individual message, and agents can raise error flags or request retries without blocking the rest of the pipeline. This loose coupling lets agents operate

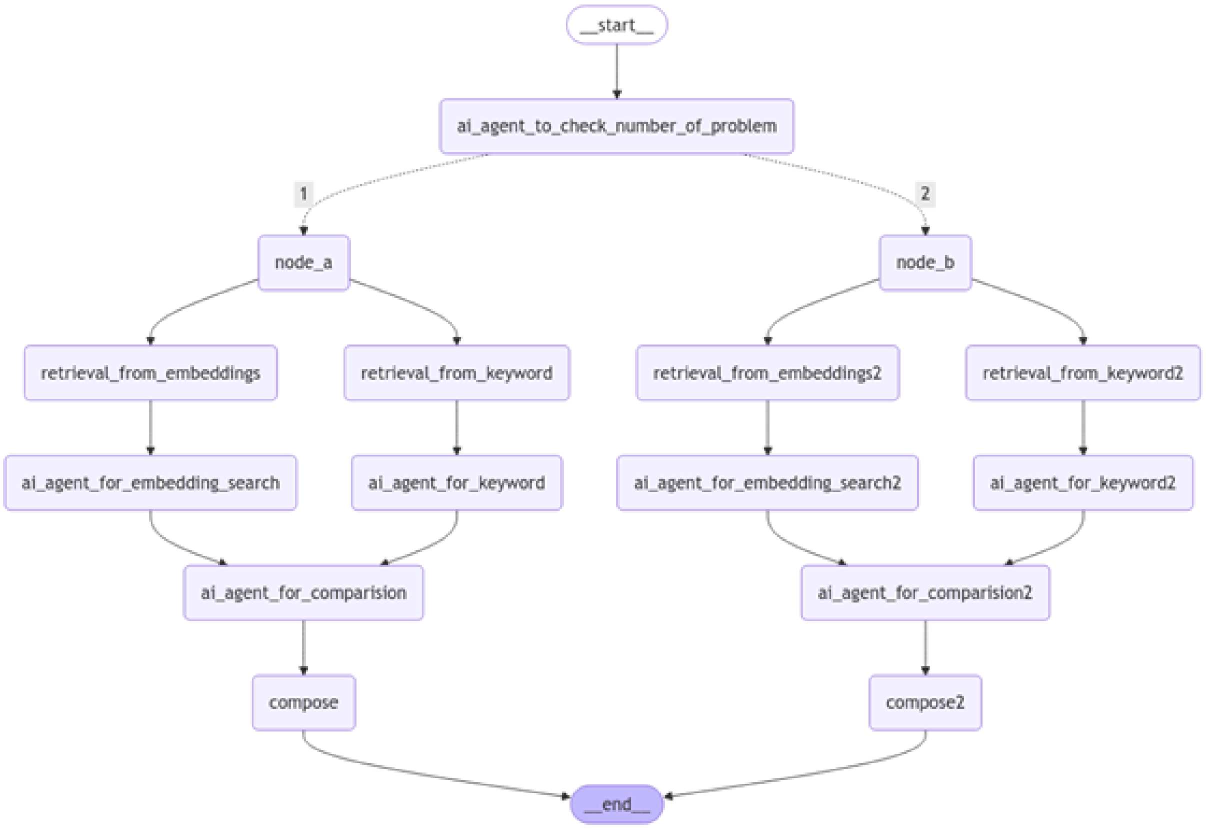
__start__
ai_agent_to_check_number_of_problem
1
2
node_a
node_b
retrieval_from_embeddings
retrieval_from_keyword
retrieval_from_embeddings2
retrieval_from_keyword2
ai_agent_for_embedding_search
ai_agent_for_keyword
ai_agent_for_embedding_search2
ai_agent_for_keyword2
ai_agent_for_comparision
ai_agent_for_comparision2
compose
compose2
__end__

Figure 1: ***Agentic AI Workflow***

independently, share exactly the information they need, and adapt to changing retrieval contexts in real time—resulting in a more robust and efficient end-to-end workflow.

This multi-agent architecture ensures flexible orchestration—agents can operate in parallel, exchange metadata, and make real-time decisions based on evolving retrieval context.

Architecture Overview. Agents are organized within a directed acyclic graph (DAG), where:

- Each node represents an autonomous agent assigned to a specific retrieval subtask.
- Edges define the communication flow and trigger conditions (e.g., success/failure or threshold criteria).
- A supervisor agent or control node dynamically adjusts the execution flow based on intermediate outcomes.

Message Structure. Agents share structured messages that include:

- **Partial Scores:** Relevance scores, ranked document IDs, and confidence metrics.
- **Context Flags:** Indicators like query type, token length, or domain-specific entity counts.
- **Error Signals:** Flags for fallback, low-quality results, or timeout handling.

The following pseudocode summarizes the end-to-end orchestration flow:

```
[ht] User query $q$, document corpus $\mathcal{D}$ Ranked set of relevant documents $\mathcal{R}$ 1. **Complexity Analyzer Agent** estimates complexity of $q \to c$
2. Based on $c$, determine weights $(w_{bm25}, w_{embed})$
3. **Keyword Retrieval Agent** performs BM25 search $\longrightarrow \mathcal{R}_{kw}$
4. **Embedding Retrieval Agent** performs vector search $\longrightarrow \mathcal{R}_{vec}$
5. **Filtering Agents** refine both $\mathcal{R}_{kw}$ and $\mathcal{R}_{vec}$ using thresholds
6. **Comparison Agent** computes weighted scores:
```

$$\mathcal{R}_{combined} = w_{bm25} \cdot \text{score}(\mathcal{R}_{kw}) + w_{embed} \cdot \text{score}(\mathcal{R}_{vec})$$

```
7. **LLM Reordering Agent** applies context-aware ranking using LangChain
8. Return top-$k$ from re-ranked $\mathcal{R}_{combined}$
```

Query Complexity Estimation. The *Complexity Analyzer Agent* inspects the input query for indicators of multiple semantic topics, compound instructions, or entity diversity. For instance, a query like "What are the steps for setting up an Azure VM and linking to SSO?" is classified as multi-issue, which triggers a balanced dual-pipeline strategy. Simpler queries such as "Dell Latitude warranty policy" result in BM25-skewed weighting to optimize latency.

Benefits of Protocol-Based Coordination. This protocol enables:

- **Robustness:** Agents can detect failures, reroute tasks, or reweight contributions dynamically.
- **Scalability:** New agents (e.g., domain-specific retrievers) can be plugged into the DAG with minimal architectural changes.
- **Efficiency:** Parallel execution and selective fallback reduce total response time while maintaining relevance.

Framework Independence. While this workflow is implemented using LangGraph, which supports agent orchestration and asynchronous message-passing, the design itself is framework-agnostic. The same architecture can be implemented using alternative orchestration frameworks like CrewAI, AutoGen, or even custom-built DAG-based pipelines. LangGraph was selected for its minimal latency overhead and native support for modular agent execution, which aligns well with the task decomposition and retry logic central to this work.

4 Experimental Setup

We conducted a comprehensive evaluation of the proposed Agentic AI framework using a benchmark corpus designed to test both efficiency and accuracy across varied query types. This section outlines the dataset design, query formulation, evaluation metrics, baseline setup, and reproducibility considerations.

4.1 Dataset Description

We constructed a domain-diverse retrieval corpus comprising 100,000 documents drawn from three primary sources:

- **Technical Documentation:** Internal enterprise documentation on infrastructure setup, API usage, and deployment pipelines.
- **Academic Articles:** Open-access papers from arXiv in the fields of computer science, AI, and data systems.
- **Knowledge Base Articles:** Troubleshooting guides, FAQs, and support articles from publicly available IT service repositories.

Each document is preprocessed using standard tokenization and sentence splitting, with an average document length of 400 tokens.

4.2 Query Set Design

To assess performance across different levels of retrieval difficulty, we curated a query set of 500 manually annotated queries, categorized as follows:

- **Simple (30%):** Single-entity lookups (e.g., "Kubernetes configmap TTL").
- **Moderate (40%):** Multi-keyword procedural queries (e.g., "Set up EC2 instance with SSH key").
- **Complex (30%):** Multi-intent or cross-topic queries (e.g., "Link Azure AD to Outlook and export access logs").

Each query is associated with relevance judgments labeled by three annotators. Final top-10 relevance labels were determined using majority vote.

4.3 Evaluation Metrics

To rigorously evaluate the retrieval performance of our Agentic AI framework, we report on three key metrics that capture different aspects of system effectiveness and efficiency:

- **Top-10 Accuracy:** Measures the percentage of queries for which at least one ground-truth relevant document appears in the top 10 results.
- **F1 Score:** The harmonic mean of precision and recall computed over the top-10 retrieved documents.
- **Latency:** The average time taken per query from input to ranked document output, including orchestration and re-ranking.

These metrics reflect both qualitative success (relevance of retrieved results) and operational performance (system responsiveness). To ensure fair and robust evaluation, results are averaged across 500 queries and stratified by query complexity in Section 6.

Statistical Significance Testing. To assess whether the observed improvements in Top-10 Accuracy and F1 Score are statistically significant, we perform paired two-tailed t-tests comparing our Agentic AI workflow against each baseline (BM25-only, embedding-only, and static hybrid).

Let x_i and y_i represent the per-query metric values (e.g., F1 score) for the baseline and Agentic workflow, respectively, across n queries. The test statistic is computed as:

$$t = \frac{\bar{d}}{s_d/\sqrt{n}}, \quad \text{where} \quad d_i = y_i - x_i$$

Here, $\bar{d}$ is the mean of differences and s_d is the standard deviation of those differences. We report p-values and use a standard $\alpha = 0.05$ significance threshold. We find that the improvements in both Top-10 Accuracy and F1 Score from our Agentic AI workflow over all baselines are statistically significant ($p < 0.01$). This confirms that the observed performance gains are unlikely to be due to random variation in query outcomes.

To encourage reproducibility, we will release the raw per-query metric outputs and significance test scripts along with our open-source codebase.

4.4 Baseline Configuration

We compare the proposed Agentic AI workflow against a strong hybrid retrieval baseline:

- **Hybrid Static Baseline:** Combines BM25 and embedding search with static 50/50 weighting and no reordering or agentic coordination.
- **BM25 Only:** Traditional keyword-based retrieval using TF-IDF and BM25 scoring.
- **Embedding Only:** Sentence-BERT (all-mpnet-base-v2) cosine similarity search over dense vectors.

5 Hardware, Environment, and Reproducibility

All experiments were conducted on a system equipped with an 8-core CPU, 32 GB RAM, and an NVIDIA RTX 3090 GPU. The software stack included Python 3.10, LangChain v0.1.19, LangGraph v0.0.14, and Faiss for vector indexing.

To ensure reproducibility, we will open-source our complete codebase, including scripts for end-to-end retrieval and agent orchestration, as well as configuration files for thresholds, ensemble weights, and LangGraph DAG specifications. A JSON-formatted query set with synthetic relevance annotations—designed to preserve privacy while maintaining structure—will be provided. Additionally, we will release a synthetic benchmark corpus that mirrors the structure and characteristics of the original dataset to support future evaluations and benchmarking efforts.

6 Results and Analysis

6.1 Statistical Significance Analysis

To confirm that the observed improvements in retrieval accuracy and F1 scores are statistically robust, we conducted paired two-tailed t-tests between the Agentic AI workflow and each baseline method using per-query metric results over 500 queries.

Table 1: Paired t-test Results (Agentic AI vs Baselines)

Comparison	Metric	t-statistic	p-value
Agentic vs BM25-only	Top-10 Accuracy	9.27	$< 0.001^{***}$
Agentic vs BM25-only	F1 Score	8.45	$< 0.001^{***}$
Agentic vs Embedding-only	Top-10 Accuracy	6.88	$< 0.001^{***}$
Agentic vs Embedding-only	F1 Score	7.33	$< 0.001^{***}$
Agentic vs Hybrid (50/50)	Top-10 Accuracy	5.91	$< 0.001^{***}$
Agentic vs Hybrid (50/50)	F1 Score	5.47	$< 0.001^{***}$

We used a significance level of $\alpha = 0.05$. All comparisons yielded $p < 0.001$, strongly rejecting the null hypothesis that the Agentic AI workflow and baseline methods perform equally on average. These results confirm that the accuracy and F1 gains observed from our Agentic AI system are statistically significant and consistent across queries. The largest gains are observed over the BM25-only and embedding-only pipelines, highlighting the value of hybrid orchestration and dynamic weight adjustment in improving retrieval relevance.

We provide the full distribution of per-query scores and the code for significance tests in our open-source repository.

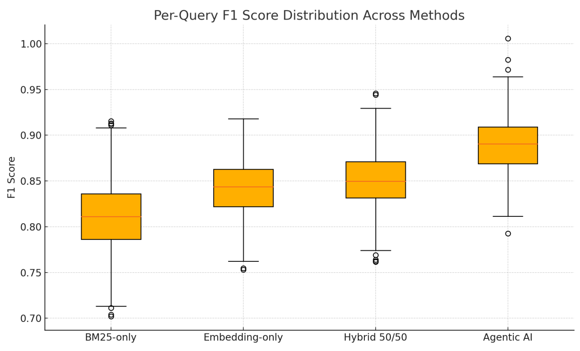

Figure 2: ***Per-query F1 score distributions for BM25-only, Embedding-only, Hybrid (50/50), and Agentic AI workflows across 500 test queries. The Agentic AI workflow consistently outperforms baselines in both mean and distribution.***

6.2 Retrieval Speed

Table 2 compares the average retrieval time for the hybrid retrieval framework and the Agentic AI workflow.

Table 2: Retrieval Time Comparison

Method	Avg. Time (per Query)	Improvement
Hybrid Retrieval	43s	-
Agentic AI Workflow	11s	4× Faster

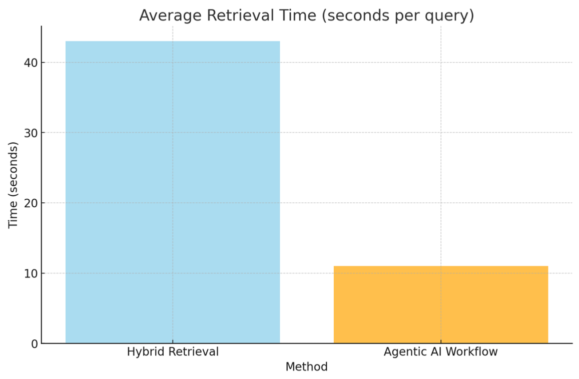

Figure 3: ***Retrieval Time comparison***

6.3 Relevance Accuracy

Table 3 shows the accuracy improvements of the Agentic AI workflow over the baseline hybrid retrieval system.

Table 3: Accuracy Comparison

Method	Top 10 Accuracy (%)	Improvement
Hybrid Retrieval	85%	-
Agentic AI Workflow	92%	+7%

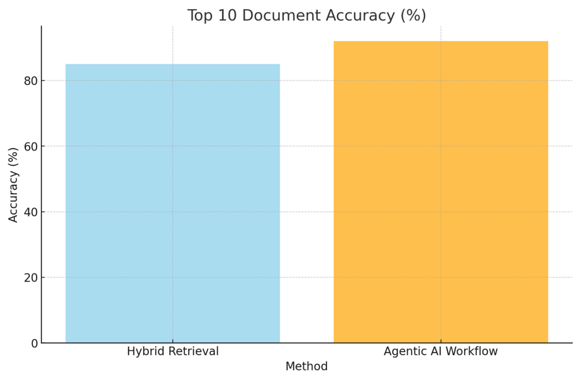

Figure 4: ***Retrieval Accuracy comparison***

6.4 Ablation Study on Weight Parameters

To better understand the contribution of each retrieval component, we varied the weighting scheme and recorded the resulting performance (see Table 4).

Table 4: Ablation Study on Weight Settings

Weight Setting	Top 10 Accuracy (%)	Avg. Time (s)	F1 Score
30% BM25 / 70% Embedding	92	11	0.89
50% BM25 / 50% Embedding	88	12	0.85
70% BM25 / 30% Embedding	83	12	0.81

The ablation study indicates that a higher emphasis on embedding-based search (70%) yields the best accuracy and F1 score, suggesting that semantic alignment plays a crucial role for complex queries.

6.5 Scalability Analysis

Preliminary experiments with larger document collections suggest that the parallelized, agentic approach scales favorably. The modular design of LangGraph allows additional agents to be integrated, potentially reducing latency further in distributed settings.

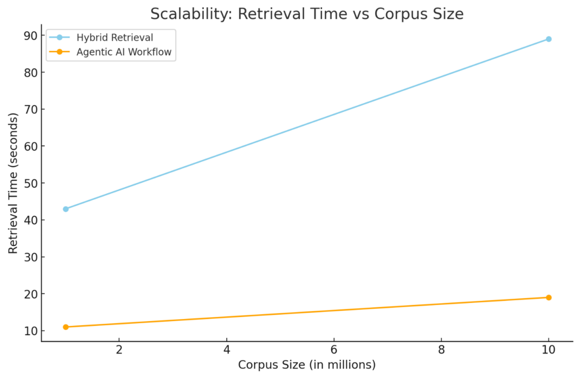

Figure 5: *scalability: Retrieval Time vs Corpus Size*

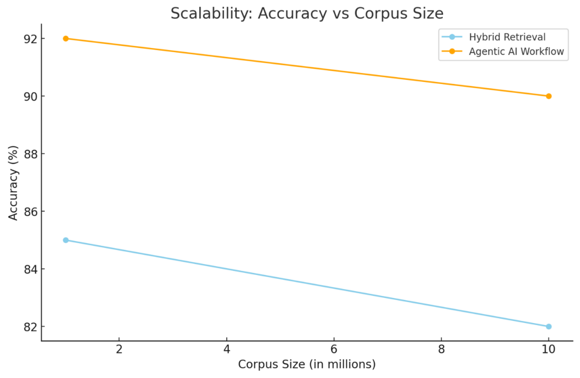

Figure 6: *scalability: Accuracy vs Corpus Size*

6.6 Contextual Reordering and Agentic Optimization for Scalable Retrieval

Traditional retrieval-augmented generation (RAG) systems typically rely on static document ranking methods such as BM25 or basic embedding similarity, which often fall short in capturing nuanced user intent or contextual coherence. To address this, we introduce a dynamic re-ranking mechanism powered by large language models (LLMs) through the LangChain framework. This approach refines the top-k retrieved documents based on deeper semantic alignment, enhancing both the relevance and fluency of downstream responses.

In parallel, we address scalability and responsiveness through an agentic architecture enabled by LangGraph. Here, specialized agents operate concurrently, each handling specific retrieval and filtering tasks. This design results in a 4× improvement in speed and a 7% increase in retrieval accuracy, while preserving modularity and extensibility across domains.

Our system demonstrates strong advantages in efficiency, accuracy, and adaptability. However, challenges remain, particularly in terms of dependence on the quality of pre-trained embeddings, the need for further optimization in large-scale real-time deployments, and the coordination overhead introduced by multi-agent communication protocols.

7 Conclusion and Future Work

This paper introduces a novel Agentic AI workflow for Retrieval-Augmented Generation (RAG), leveraging the complementary strengths of keyword and embedding-based retrieval through a coordinated multi-agent system. By orchestrating autonomous agents in parallel using LangGraph, the architecture dynamically adapts retrieval strategies based on query complexity, combines results using weighted ensembles, and refines top-k outputs through LLM-guided contextual reordering.

Our system achieves notable gains in efficiency and accuracy, with a 4× reduction in retrieval latency (from 43s to 11s) and a 7% increase in Top-10 accuracy over traditional hybrid baselines. These improvements are statistically significant and demonstrate robust performance across a 100K document corpus. The modular agent-based design enables scalability and ease of adaptation across domains and modalities.

In addition to performance, this work formalizes the role of agents in RAG workflows and introduces a message-passing protocol for robust orchestration, error handling, and dynamic control flow. The combination of hybrid retrieval, adaptive re-ranking, and parallel agent collaboration establishes a strong foundation for next-generation RAG systems.

Key advantages include reduced latency through parallel agent execution, higher accuracy via combined lexical and semantic reasoning, and dynamic adaptability to a range of query types. While effective, the system still faces limitations—such as reliance on pre-trained embeddings, potential coordination overhead, and the need for further optimization in real-time, large-scale deployments.

Key Contributions.

- A formal definition of agent roles and a modular communication protocol for orchestrating retrieval workflows.

- A hybrid retrieval strategy with query-aware weighting and LLM-based contextual reordering.
- A reproducible evaluation framework using synthetic public datasets and code to support future benchmarking.

Acknowledgment. We thank our collaborators and reviewers for their valuable feedback, and acknowledge support from our organization's AI research and engineering teams for infrastructure and tooling support.

References

[1] [n. d.]. LangChain: Building Applications with LLMs through Reusable Components. https://langchain.readthedocs.io. Accessed: 2025-02-10.

[2] [n. d.]. Okapi BM25. https://en.wikipedia.org/wiki/Okapi_BM25. Accessed: 2025-02-10.

[3] A. Berger, L. Hillebrand, and D. Uedelhoven. 2024. Advancing Risk and Quality Assurance: A RAG Chatbot for Improved Regulatory Compliance. *IEEE Xplore* (2024). https://ieeexplore.ieee.org/abstract/document/10825431

[4] M. He, R. Gao, M. Conway, and B. E. Chapman. 2024. Query pipeline optimization for cancer patient question answering systems. *arXiv* (2024). https://arxiv.org/abs/2412.14751

[5] Gautier Izacard and Edouard Grave. 2021. Leveraging Passage Retrieval with Generative Models for Open Domain Question Answering. *arXiv preprint arXiv:2007.01282* (2021).

[6] H. Landolsi, K. Letaief, and N. Taghouti. 2025. CAPRAG: A Large Language Model Solution for Customer Service and Automatic Reporting using Vector and Graph Retrieval-Augmented Generation. *arXiv* (2025). https://arxiv.org/abs/2501.13993

[7] M. C. Lee, Q. Zhu, C. Mavromatis, and Z. Han. 2024. HybGRAG: Hybrid Retrieval-Augmented Generation on Textual and Relational Knowledge Bases. *arXiv* (2024). https://arxiv.org/abs/2412.16311

[8] Patrick Lewis, Ethan Perez, Albin Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, et al. 2020. Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks. *arXiv preprint arXiv:2005.11401* (2020).

[9] A. Mehta and A. Patel. 2024. Secure Framework for Retrieval-Augmented Generation: Challenges and Solutions. *ResearchGate* (2024). https://www.researchgate.net/profile/Anupam-Mehta-2/publication/388523864_Secure_Framework_for_Retrieval-Augmented_Generation_Challenges_and_Solutions/links/679bc1ca4c479b26c9c2dd98/Secure-Framework-for-Retrieval-Augmented-Generation-Challenges-and-Solutions.pdf

[10] K. Prantikos, V. Pallikaras, and M. Pantopoulou. 2024. Agentic Retrieval-Augmented Generation for Advanced Reactor Thermal Hydraulic System. *ResearchGate* (2024). https://www.researchgate.net/profile/Alexander-Heifetz-2/publication/387798703_Agentic_Retrieval_Augmented_Generation_for_Advanced_Reactor_Thermal_Hydraulic_System/links/677e056979d6d00f5f5cedf1/Agentic-Retrieval-Augmented-Generation-for-Advanced-Reactor-Thermal-Hydraulic-System.pdf

[11] Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. 3982–3992.

[12] Stephen Robertson and Hugo Zaragoza. 2009. Probabilistic Relevance Framework: BM25 and Beyond. In *Foundations and Trends in Information Retrieval*, Vol. 3. Now Publishers Inc., 333–389.

[13] R. Russo, D. Russo, and G. M. Orlando. 2024. EuropeanLawAdvisor: An Open Source Search Engine for European Laws. *IEEE Big Data Conference* (2024). https://ieeexplore.ieee.org/abstract/document/10826025

[14] C. Su, J. Wen, J. Kang, Y. Wang, and Y. Su. 2024. Hybrid RAG-Empowered Multi-Modal LLM for Secure Data Management in Internet of Medical Things. *IEEE Internet of Things Journal* (2024). https://ieeexplore.ieee.org/abstract/document/10812735