Mitigating Hallucinations in Retrieval-Augmented Generation through Stateful Agentic Workflows

Zahra Fakoor Harehdasht
Department of IT and Technology
IU International University of Applied Sciences
Erfurt, Germany
za.fakoor@gmail.com

Marius Schönberger
Department of IT and Technology
IU International University of Applied Sciences
Erfurt, Germany
marius.schoenberger@iu.org

Silke Vaas
Department of IT and Technology
IU International University of Applied Sciences
Erfurt, Germany
silke.vaas@iu.org

***Abstract*—Applications of Large Language Models often produce incorrect information, commonly referred to as hallucinations, which poses a significant challenge for enterprise knowledge management systems that rely on reliable automated responses. This paper investigates whether stateful, multi-agent workflows can mitigate such errors more effectively than conventional linear retrieval-augmented generation pipelines. The study contributes to the growing body of research on generative artificial intelligence and large language model applications by focusing on reliability, verification mechanisms, and the architectural design of retrieval-augmented generation workflows. The main contributions of this paper are a specialized multi-agent RAG architecture with dedicated agents for query rewriting, routing, hybrid retrieval, and verification, as well as a state-machine-based orchestration framework enabling iterative reasoning and conditional execution beyond linear pipelines. The architecture follows a two-stage mitigation strategy combining proactive retrieval refinement and reactive verification, including the ability to abstain when evidence is insufficient. It was evaluated through controlled experiments using synthetic enterprise datasets and compared it with a baseline linear retrieval-augmented generation pipeline. Performance was measured using an established evaluation framework for retrieval-augmented generation systems. The results demonstrate significant improvements across key evaluation metrics, including a 42.5% relative increase in faithfulness and substantial gains in answer accuracy and context recall. These improvements primarily result from separating error sources across specialized agents and the integration of automated verification steps. The findings highlight the importance of structured orchestration and verification mechanisms for enhancing the reliability of large language model applications. The results indicate that stateful, multi-agent architectures can increase the trustworthiness of enterprise generative artificial intelligence systems without requiring fine-tuning of the underlying language model.**

***Keywords*—Retrieval-Augmented Generation; Hallucination Mitigation; Large Language Models; Multi-Agent Systems**

I. INTRODUCTION

Enterprises increasingly rely on large collections of heterogeneous digital information, including technical documentation, operational runbooks, knowledge bases, and API documentation. Unlike traditional structured databases, this information often exists in semi-structured or unstructured formats, making effective retrieval challenging [1]. Conventional enterprise search systems typically rely on keyword matching, which performs poorly when user queries differ semantically from the terminology used in source documents. Such vocabulary mismatches can prevent relevant information from being retrieved and reduce the overall accessibility of organizational knowledge [2]. As a result, knowledge discovery remains inefficient despite the availability of large information repositories [3].

Recent advances in large language models (LLMs) have enabled new approaches for interacting with enterprise knowledge systems. Retrieval-Augmented Generation (RAG) combines LLMs with external document retrieval, allowing generated responses to be grounded in specific knowledge sources rather than relying solely on model training data [4]. While RAG improves semantic search and question answering capabilities, it remains vulnerable to hallucinations [5]. In enterprise environments, such errors can undermine trust and may lead to incorrect technical decisions or misleading information [6].

Conventional RAG systems typically follow a linear retrieve–then–generate pipeline, limiting iterative reasoning and structured verification [7]. Consequently, hallucinations are mainly addressed through retrieval improvements or prompt engineering, while architectural mitigation approaches remain underexplored [8]. To address this limitation, this paper proposes a stateful agentic workflow that decomposes the RAG process into specialized agents for query refinement, retrieval strategy selection, response generation, and output verification. These agents operate within a state-machine-based orchestration framework that enables iterative reasoning and validation before responses are returned to the user [9]. This architecture introduces structured verification and multi-agent coordination to mitigate hallucinations and improve the reliability of RAG-based enterprise knowledge systems [10], [11].

The remainder of this paper is structured as follows. Section 2 reviews related work on RAG, hallucination taxonomies, and agentic AI workflows. Section 3 introduces the proposed system architecture, including the data ingestion pipeline, the agentic graph design, and the hallucination mitigation strategy. Section 4 describes the evaluation methodology, outlining the experimental setup, baselines, metrics, and quantitative findings. Section 5 discusses the results and their technical and practical implications. Section 6 concludes the paper and highlights directions for future research.

II. Related Work

To provide a foundation for the present study and position it within the existing body of knowledge, the following section reviews relevant related work.

A. Retrieval-Augmented Generation

RAG, first introduced by Lewis et al. [4], combines LLMs with external knowledge retrieval to ground generated responses in updatable document sources. The architecture links a pre-trained Large Language Model (parametric memory) with an external document store (non-parametric memory), enabling access to domain-specific knowledge without retraining the model [4]. Before query time, documents are segmented into smaller chunks, which are converted into embedding vectors and stored in a vector index to enable similarity-based retrieval. As discussed by Gao et al. [10], the chosen chunking strategy significantly affects retrieval performance because segment size influences both semantic coherence and retrieval precision.

During inference, RAG typically follows a two-stage pipeline [4]. First, a retriever performs a similarity-based retrieval to identify the most relevant document chunks. These passages are then incorporated into the prompt presented to the language model, which generates an answer based on the context provided. To improve retrieval quality, several extensions have been proposed. Query transformation methods reformulate user queries to improve retrieval effectiveness [13], [14]. Hybrid search combines dense vector retrieval with lexical approaches, such as Best Matching 25 (BM25) [15], often merging rankings using techniques like Reciprocal Rank Fusion [16]. In addition, reranking methods refine initially retrieved candidates using more precise relevance models, including cross-encoders or LLM-based scoring approaches [17]. These techniques aim to improve the quality of retrieved evidence, which is critical for reducing hallucinations in RAG systems.

B. Hallucination Taxonomy

Hallucinations remain a persistent challenge in LLMs, even when RAG is used to ground responses in external knowledge sources. As noted by Ji et al. [5], hallucinations occur when generated outputs contain statements that are unsupported by the retrieved evidence or contradict the underlying sources.

Within RAG pipelines, Zhang & Zhang [18] distinguish between two primary categories of hallucination. Retrieval-induced hallucinations arise when the retrieval component fails to provide relevant or sufficient context. This can occur when irrelevant documents are retrieved, when passages lack critical information, or when the retrieved sources contain conflicting evidence. In such cases, the language model may rely on its internal parameters to infer missing information.

By contrast, generation-induced hallucinations originate during the generation stage. Even when relevant context is available, the model may misinterpret source material, or introduce unsupported details while producing fluent responses, making these errors difficult to detect [18].

These failure modes indicate that improving retrieval alone is insufficient for reliable RAG systems. Effective mitigation therefore requires architectural mechanisms that verify generated statements against retrieved evidence and enforce stronger grounding during the generation process.

C. Agentic AI Workflows

Managing the complexity of advanced RAG pipelines with a single monolithic architecture can be difficult. Multi-agent systems offer an alternative approach by decomposing complex tasks into smaller components handled by specialized agents. According to Russell & Norvig [19], an agent is an autonomous entity capable of perceiving its environment, making decisions, and acting toward defined goals. Wooldridge [20] further characterizes agents through properties such as autonomy, reactivity, pro-activeness, and the ability to interact with other agents.

Recent research has applied this paradigm to LLM–driven systems, often referred to as LLM-Agents [21]. In such architectures, complex workflows are divided into modular components that perform distinct tasks. Within RAG pipelines, these agents can assume roles such as query refinement, document retrieval, response generation, and verification, enabling clearer separation of responsibilities and more flexible system design.

To coordinate these interactions, frameworks such as LangGraph provide orchestration mechanisms for stateful agent workflows. Unlike linear pipelines, the graph-based architecture allows iterative and conditional execution paths, enabling agents to exchange information, trigger additional steps, or perform verification loops during runtime [9].

However, while agentic workflows have been proposed to structure complex LLM-driven systems, their application as architectural mechanisms for mitigating hallucinations in retrieval-augmented generation remains comparatively underexplored. This multi-agent perspective provides a flexible foundation for improving reliability in RAG systems, as it enables targeted interventions, such as verification steps, to mitigate hallucinations while maintaining modular and extensible system architectures.

III. System Architecture

The architecture is divided into an offline data ingestion path and an online multi-agent query pipeline (see Figure 1). Both components use a shared knowledge store implemented with PostgreSQL and the pgvector extension [22].

A. Data Ingestion

The system utilizes a modular Extract, Transform, Load (ETL) pipeline to convert heterogeneous inputs (i.e., PDFs and other documents) into a queryable format. Text is extracted from PDFs while preserving their original layout and hierarchical structure to maintain semantic coherence in complex documents. PostgreSQL with the pgvector extension is used as the unified knowledge store, enabling the storage of transactional metadata alongside 3072-dimensional embeddings generated using OpenAI's text-embedding-3-large model [22], [23].

Content is segmented using recursive character splitting [12], which partitions text along logical boundaries rather than limited arbitrary character counts, thereby preserving

semantic context. Each chunk is enriched with an LLM-generated headline to improve vector search accuracy, representing a form of proactive query transformation [13].

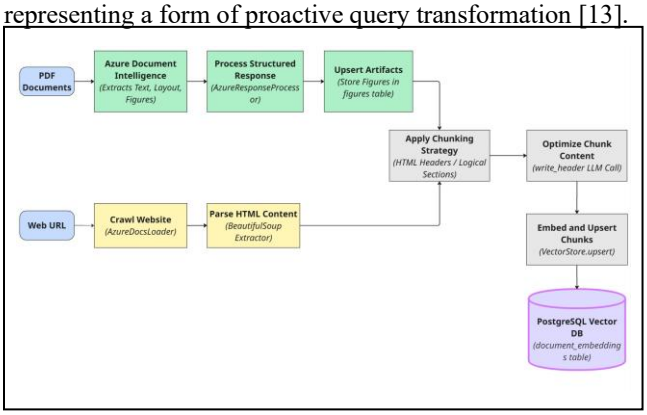

Figure 1. The dual-source data ingestion pipeline (adapted from [10])

To support multimodal queries, the ingestion pipeline, presented in Figure 1, extracts figures and tables and sorts their metadata and captions in a relational table linked to the corresponding text chunks via unique identifiers.

B. The Agentic Graph

The query pipeline is modeled as a stateful graph using LangGraph rather than a brittle linear RAG [9]. This architecture, illustrated in Figure 2, centers on a shared state object that functions as the system's short-term memory, which stores the rewritten query, the retrieved document sets, and the verification scores across all nodes. The shared state acts as a centralized blackboard where each agent writes its output (e.g., the reformulated query or retrieved chunks). This allows the Verification Agent to compare the final answer against the original query and the retrieved context stored earlier in the state, ensuring end-to-end consistency.

By leveraging this shared state, the system moves through nodes that make local decisions, which allows conditional branching and iterative loops instead of a fixed sequence. This allows the system to dynamically reason about and refine information quality before delivering the response. The pipeline consists of the following key nodes:

- Query Rewriter Agent: User prompts are often fuzzy or ambiguous. This agent reformulates user inputs into optimized search queries, improving semantic retrieval precision compared to standard keyword-based approaches [13].
- Router Agent: To optimize performance and cost, this agent analyzes the rewritten query and determines whether to query the internal vector store or external sources based on the detected intent.
- Hybrid Search: Semantic retrieval alone may miss exact technical terms such as error codes or product versions. The system therefore combines keyword-based search (for exact lexical matching and acronyms) with semantic search (for conceptual similarity and intent) executing both in parallel and merging the results to improve coverage and recall.
- Batch Reranker Agent: Initial retrieval may produce noisy or irrelevant results. This agent uses an LLM to score the relevance of retrieved candidates in a single

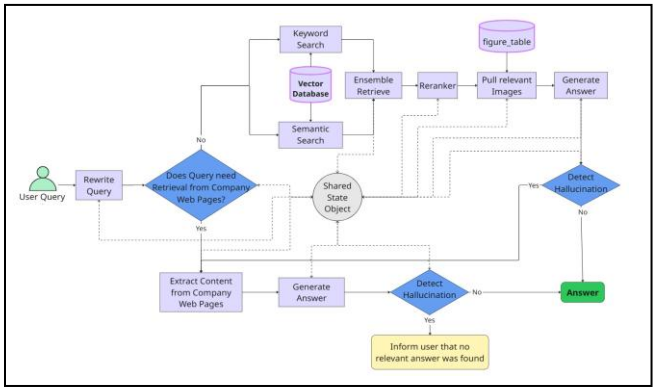

Figure 2. Proposed architecture for RAG hallucination mitigation via agentic workflow.

batch call [17], reducing noise and ensuring that the Generation Agent receives high-quality contextual input.

To ensure consistent behavior, each agent utilizes structured system prompts with strict output constraints. For example, the Verification Agent is prompted: 'Given the context [C], does the answer [A] contain any statements not explicitly supported? Answer only with a JSON object: {is_hallucination: bool, confidence: float}.' Similar constraints are applied to the Query Rewriter and Reranker to ensure the state object receives parseable data for programmatic decision-making.

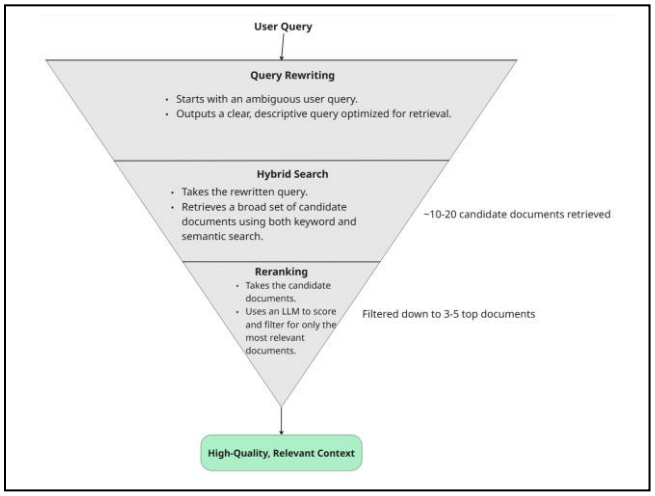

Figure 3. Proposed architecture for RAG hallucination mitigation via agentic workflow (adapted from [16])

C. Hallucination Mitigation Strategy

The core contribution of this architecture is a two-layered hallucination mitigation framework:

The first layer, *proactive mitigation,* targets retrieval-induced hallucinations [18] by reducing noise that may mislead the LLM. Through query rewriting and document

reranking, the system ensures that the generator receives a compact, high-quality set of relevant passages. Figure 3 illustrates this process as a filtering funnel, where each stage progressively refines the context provided to the generator.

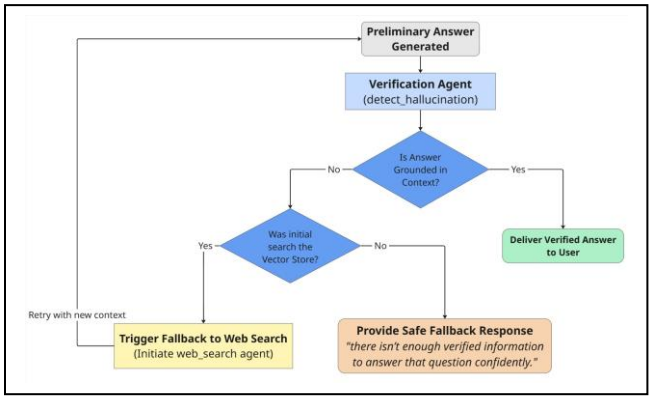

Figure 4. The Reactive Verification and Fallback Loop for hallucination mitigation (adapted from [18])

The second layer, reactive mitigation, addresses generation-induced hallucinations [18]. After a draft answer is generated, it is passed to a Verification Agent that evaluates the response against the reranked source documents. The agent acts as an automated fact-checker and emits a boolean flag (*is_hallucination*) along with a confidence score. If a hallucination is detected, the system withholds the response and triggers a fallback mechanism, either retrieving additional evidence from external sources or returning a safe "insufficient evidence" response. This process is illustrated in Figure 4.

IV. EVALUATION

The evaluation was designed to assess whether the proposed stateful agentic workflow reduces hallucinations and improves answer accuracy in a technical enterprise setting.

A. Experimental Setup

To ensure a controlled environment, a fresh PostgreSQL database was populated with synthetic, enterprise-style documents. These documents covered four categories: IT security, human resource policy, technical training, and employee onboarding. The synthetic dataset consisted of 150 documents totaling approximately 600 pages of text. The 'Technical Training' category specifically included 15 multimodal documents containing images, such as server architecture diagrams and network flowcharts. The content was designed to mimic internal wikis, including semi-structured tables for HR policies and unstructured technical logs for IT security.

Two models from OpenAI were used, *GPT-4o* for agentic reasoning, and *text-embedding-3-large* for vector generation [23]. A curated evaluation dataset of questions was used to test different retrieval paths and capabilities, both straightforward fact retrieval and more complex synthesis tasks. The 50 evaluation queries were classified as follows: Text-based (35 queries, e.g., 'What are the steps for employee onboarding?') and Multimodal/Tabular (15 queries, e.g., 'According to the IT diagram, which port is used for the API gateway?'). The query distribution across document categories was as follows: IT Security (15), HR Policy (12), Technical Training (13), and Onboarding (10).

B. Baselines

Two configurations were compared:

- Baseline RAG: A simplified, linear pipeline with a basic semantic retriever and a generator LLM.
- Agentic RAG (the proposed system): The full multi-agent system, which includes the Query Rewriter, Batch Reranker, and the reactive verification loop.

C. Metrics

Performance was evaluated using the Retrieval-Augmented Generation Assessment (RAGAS) framework [11], using GPT-4 as the evaluator LLM. RAGAS provides a set of metrics designed to measure RAG pipelines [11]:

- *Faithfulness*: Measures whether the answer is grounded solely in the retrieved context and serves as the primary indicator for hallucination.
- *Answer Correctness*: The factual accuracy of the response relative to the ground truth.
- *Context Recall*: Evaluates whether the retriever successfully identified all necessary information required to answer the query.
- *Answer Relevancy*: Measures how well the generated response addresses the intent of the user's prompt.

D. Quantitative Findings

The performance of both systems was measured across a set of 50 different queries. The overall results, shown in Table I., represent the average score across all evaluations, where 1.00 indicates a perfect score.

TABLE I. Comparative RAGAS Results

Metric	Baseline RAG	Agentic RAG (Artifact)
Faithfulness	0.59	0.84
Answer Correctness	0.66	0.89
Context Recall	0.65	0.86
Answer Relevancy	0.74	0.93

These quantitative results indicate a consistent performance advantage for the agentic workflow. A detailed analysis of how these specific architectural components contributed to these increases in scores is presented in the following section.

V. RESULTS AND DISCUSSION

Building on the conducted evaluation, the following section presents the obtained results and discusses their implications regarding hallucination reduction and answer accuracy.

A. Retrieval-Augmented Generation Assessment scores

The quantitative evaluation in Table 1 demonstrates consistent improvements across all RAGAS dimensions when comparing the linear baseline RAG pipeline to the proposed stateful agentic architecture.

Most notably, faithfulness increased from 0.59 to 0.84, representing a relative improvement of 42.5%. This substantial gain indicates that the proposed architecture effectively reduces both retrieval- and generation-induced hallucinations.

The improvement in answer correctness (0.66 → 0.89) shows that stronger contextual grounding leads to higher factual accuracy. Importantly, correctness did not increase at the expense of relevancy, which improved from 0.74 to 0.93. This suggests that query rewriting and reranking stages do not merely constrain outputs conservatively but align retrieval more closely with user intent.

The rise in context recall (0.65 → 0.86) further supports the claim that the combination of hybrid retrieval and reranking effectively reduces hallucinations caused by insufficient or noisy context. In baseline RAG systems, such conditions often force the generator to interpolate from parametric memory. By increasing retrieval coverage and reducing the noise prior to generation, the agentic system shifts the process from generative inference to evidence-based synthesis.

A qualitative analysis of execution paths revealed that straightforward factual queries (e.g., 'What is the standard employee notice period?') typically followed a linear path with high initial retrieval confidence. In contrast, ambiguous or complex queries (e.g., 'Cross-reference recent security guidelines with internal protocols to identify compliance gaps') frequently triggered the Query Rewriter to iterate and the Router to pull from external web sources when the internal vector store returned low-confidence scores. However, even with these adaptive paths, the system still faced challenges with multi-hop reasoning and queries requiring information spread across multiple disparate documents. In these cases, the Batch Reranker occasionally filtered out a necessary 'linking' document if its individual relevance score appeared low, contributing to the gap between Context Recall (0.86) and the higher-performing metrics like Answer Relevancy (0.93).

Beyond the primary evaluation using GPT-4o, supplementary tests were conducted using other large language models, specifically Claude 3.5 Sonnet and the open-source Llama 3.1 (70B). These preliminary runs showed that the relative performance gains remained consistent across different model architectures. While baseline scores varied, the agentic workflow consistently improved the faithfulness and accuracy of the output compared to linear pipelines, confirming that the benefits of stateful orchestration are consistent across models.

Overall, the improvements across all metrics indicate that effective hallucination mitigation emerges primarily from architectural orchestration rather than from improvements in the underlying foundation model.

B. Why the agentic system outperformed the baseline

The performance improvements observed can be attributed to three structural characteristics of the system.

1) Architectural Separation of Failure Modes

Linear RAG pipelines implicitly assume that retrieval quality alone determines output reliability [4]. In contrast, the proposed system models hallucination as a two-class problem: retrieval-induced and generation-induced. The proactive mitigation layer addresses the former by improving context quality prior to generation, while the reactive verification layer targets the latter through post-generation validation. This operationalizes the hallucination taxonomy [18] within an executable workflow.

2) Graph-Based Orchestration

Unlike linear pipelines, the system maintains a shared workflow state that propagates rewritten queries and intermediate results. This enables conditional branching, iterative refinement, and fallback strategies. As a result, RAG is transformed from a feedforward pipeline into a controlled, stateful optimization process.

3) Verification as Confidence Calibration

The verification agent acts as a control mechanism by evaluating the consistency between generated claims and source documents. This shifts the optimization objective from plausibility to evidence.

Importantly, the verification loop allows the system to abstain (“insufficient evidence”) rather than generate unsupported content, which contributes significantly to the observed improvements in accuracy.

Baseline RAG systems lack such a mechanism and therefore tend to produce overly confident responses under uncertainty.

C. Technical and Practical Implications

The results provide several insights for the deployment of retrieval-augmented generation systems in enterprise environments.

First, they suggest that hallucination mitigation should be understood primarily as an architectural challenge rather than a purely model-level problem. Simply improving the underlying LLM is not sufficient to ensure reliable results [5]. Instead, improvements in reliability result from structured orchestration of retrieval and generation components [12]. In particular, the coordinated use of query optimization, retrieval refinement, reranking, and verification significantly strengthens the factual grounding of the generated responses.

From a practical perspective, this architectural approach offers an additional advantage: reliability improvements can be achieved without modifying or fine-tuning the underlying base model. This enables a model-agnostic design, allowing organizations to switch or upgrade LLM providers without redesigning the surrounding system. Such flexibility is particularly valuable in enterprise settings where model capabilities, costs, and vendor strategies evolve rapidly.

Second, the introduction of a verification loop substantially increases the system’s trustworthiness. In compliance-sensitive areas such as human resources or IT security, incorrect or unsupported statements can lead to

operational or legal risks. A verification layer that validates generated responses against retrieved evidence therefore serves as an important safeguard, preventing unfounded or fabricated statements from being returned to the user.

A potential risk in this architecture is 'error propagation,' where an initial misjudgment by the Query Rewriter or Reranker biases all subsequent agents. Without human-in-the-loop validation, the system relies on the LLM's internal 'self-correction' capability. Future iterations should include a human-flagging mechanism for low-confidence outputs.

Overall, these findings indicate that the reliability of enterprise RAG systems depends less on increasingly powerful language models and more on carefully designed retrieval, orchestration and verification processes.

VI. Conclusion and Future Work

Building upon the presented results and discussion, the final section draws conclusions, addresses limitations, and identifies opportunities for future work.

A. Architectural Contributions to Reliable RAG Systems

This study demonstrates that hallucination reduction in retrieval-augmented generation systems can be significantly improved through architectural design. In particular, separating error modes via a two-stage mitigation strategy, combined with a graph-based orchestration, and a verification agent capable of abstaining when evidence is insufficient, substantially enhances system reliability.

These improvements are achieved without fine-tuning the underlying foundation model, highlighting that central role of architectural design in ensuring reliable system behavior.

From a design science research perspective, the resulting artifact offers both a functional prototype for use in companies and a conceptual model for structuring hallucination-aware RAG systems.

B. Trade-off: Reliability vs. Latency

The improved reliability of the proposed architecture comes at the cost of additional computational overhead. Additional agents, reranking steps, and the verification loop introduce the number of model calls and, consequently, higher response latency compared to a minimal linear RAG pipeline. This reflects an inherent trade-off between response speed and output reliability.

C. Directions for Future Research

In enterprise settings where incorrect information may lead to operational, financial, or compliance risks, prioritizing reliability over minimal latency is often justified. Nevertheless, future implementations may benefit from adaptive strategies that dynamically adjust verification depth based on query criticality, enabling a balance between efficiency and robustness. Several directions for future research emerge from this work. One promising avenue is the integration of knowledge graphs into the architecture. While the current pipeline primarily operates on unstructured text, extending it to incorporate structured representations - such as entities (i.e., services, products, or infrastructure components) - could enable more advanced multi-hop reasoning across documents. However, this approach introduces additional challenges, including schema design and entity disambiguation, which must be addressed carefully to maintain system reliability [24].

Another direction involves domain-specific fine-tuning of core models or reranking components, which may enhance grounding in specialized enterprise corpora such as technical documentation or regulatory guidelines.

Finally, future studies could evaluate agentic RAG architectures on larger and more diverse real-world datasets to better assess scalability and robustness under production conditions. Such investigations would provide deeper insights into the performance of architectural hallucination mitigation strategies in dynamic enterprise knowledge environments.

References

[1] A, McAfee and E. Brynjolfsson, “Big data: The management revolution,” Harvard Business Review, vol. 90, no. 10, pp. 61–68, Oct. 2012. [Online]. Available from https://hbr.org/2012/09/big-datas-management-revolutio, 2026.03.27.

[2] G. W. Furnas, T. K. Landauer, L. M. Gomez and S. T. Dumais, “The vocabulary problem in human-system communication,” Communications of the ACM, vol. 30, no. 11, pp. 964-971, 1987, doi: 10.1145/32206.3221.

[3] M. Alavi and D. E. Leidner, “Review: Knowledge management and knowledge management systems: Conceptual foundations and research issues,” MIS Quarterly, vol. 25, no. 1, pp. 107-136, 2001, doi: 10.2307/3250961.

[4] P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Küttler, M.Lewis, W-T.Yih, T. Rocktäschel, S. Riedel and D. Kiela, “Retrieval-augmented generation for knowledge-intensive NLP tasks,” Advances in Neural Information Processing Systems, vol. 33, pp. 9459-9474, 2020, doi: 10.48550/arXiv.2005.11401.

[5] Z. Ji, N. Lee, R. Frieske, T. Yu, D. Su, Y. Xu, E. Ishii, Y. Bang, D. Chen, W. Dai, H.S. Chan, A. Madotto and P. Fung, “Survey of hallucination in natural language generation,” ACM Computing Surveys, vol. 55, no. 12, pp. 1-38, 2023, doi: 10.1145/3571730.

[6] A. B. Arrieta, N. Díaz-Rodríguez, J. Del Ser, A. Bennetot, S. Tabik, A. Barbado, S. García, S. Gil-Lopez, D. Molina, R. Benjamins, R. Chatila and F. Herrera, “Explainable Artificial Intelligence (XAI): Concepts, taxonomies, opportunities and challenges toward responsible AI,” Information Fusion, vol. 58, pp. 82-115, 2020, doi: 10.48550/arXiv.1910.10045.

[7] S. Mishra, S. Niroula, U. Yadav, D. Thakur, S. Gyawali, and Shiva Gaire, “SoK: Agentic Retrieval-Augmented Generation (RAG): Taxonomy, Architectures, Evaluation, and Research Directions, arXiv preprint, 07 Mar. 2026, doi: 10.48550/arXiv. 2603.07379 [Online]. Available from https://arxiv.org/html/2603.07379v1. 2026.03.27.

[8] L.M. Amugongo, P. Mascheroni, S. Brooks, S. Doering, F. Seidel, “Retrieval augmented generation for large language models in healthcare: A systematic review”. PLOS Digit Health. 2025 Jun 11;4(6):e0000877. doi: 10.1371/journal.pdig.0000877. PMID: 40498738; PMCID: PMC12157099. [Online]. Available from https://pmc.ncbi.nlm.nih.gov/articles/PMC12157099/, 2026.03.27.

[9] LangChain, “LangGraph,” [Online]. Available from https://python.langchain.com, 2026.03.27.

[10] Confident AI Inc., *DeepEval. The LLM Evaluation Framework*. [Online]. Available from https://deepeval.com, 2026.03.27.

[11] S. Es, J. James, L. E. Anke, and S. Schockaert, “RAGAs: Automated evaluation of retrieval augmented generation,” Proc. 18th Conference of the European Chapter of the Association for Computational Linguistics System Demonstrations (EACL 2024), March 2024, pp. 150-158, doi: 10.18653/v1/2024.eacl-demo.

[12] Y. Gao et al., “Retrieval-Augmented Generation for Large Language Models: A Survey,” arXiv preprint, pp. 1-24, 2024, doi: 10.48550/arXiv.2312.10997.

[13] X. Ma, Y. Gong, P. He, H. Zhao and N. Duan, “Query Rewriting for Retrieval-Augmented Large Language Models,” Proc. Conference on Empirical Methods in Natural Language Processing (EMNLP 2023), Dec. 2023, pp. 5303-5315, doi: 10.18653/v1/2023.emnlp-main.322.

[14] H. S. Zheng, S. Mishra, X. Chen, H. T. Cheng, E. H. Chi, Q. V. Le and D. Zhou, “Take a step back: Evoking reasoning via abstraction in large language models,” arXiv preprint, pp. 1-38, Mar. 2024, doi: 10.48550/arXiv.2310.06117.

[15] S. Robertson and H. Zaragoza, “The probabilistic relevance framework: BM25 and beyond,” Foundations and Trends in Information Retrieval, vol. 3, no. 4, pp. 333-389, 2009, doi: 10.1561/1500000019.

[16] G. V. Cormack, C. L. Clarke and S. Buettcher, “Reciprocal rank fusion outperforms condorcet and indi-vidual rank learning methods,” Proc. 32nd International ACM SIGIR Conference on Research and Development in Information Retrieval, 2009, pp. 758-759. [Online]. Available from https://cormack.uwaterloo.ca/cormacksigir09-rrf, 2026.03.27.

[17] N. Thakur, N. Reimers, A. Rucklé, A. Srivastava and I. Gurevych, “BEIR: A heterogenous benchmark for zero-shot evaluation of information retrieval models,” arXiv preprint, pp. 1-24, Oct. 2021, doi: 10.48550/arXiv.2104.08663.

[18] W. Zhang and J. Zhang, “Hallucination Mitigation for Retrieval-Augmented Large Language Models: A Review,” Mathematics, vol. 13, no. 5, 856, 2025. doi: 10.3390/math13050856.

[19] S. J. Russell and P. Norvig, Artificial intelligence: A modern approach. 4th ed., Harlow, UK: Pearson Education Limited, 2022.

[20] M. Wooldridge, An introduction to multiagent systems. 2nd ed., West Sussex, UK: John Wiley & Sons.

[21] Z. Xi, W. Chen, X. Guo, W. He, Y. Ding, B. Hong, ... and T. Gui, “The rise and potential of large language model based agents: A survey,” Science China Information Sciences, vol. 68, no. 2, 121101, 2025, doi: 0.1007/s11432-024-4222-0.

[22] A. Lamba, “pgvector (v0.7.2),”, GitHub. [Online]. Available from https://github.com/pgvector/pgvector, 2026.03.27.

[23] OpenAI, “New embedding models and API updates,” 2024. [Online]. Available from https://openai.com/index/new-embedding-models-and-api-updates, 2026.03.27.

[24] S. Pan, L. Luo, Y. Wang, C. Chen, J. Wang and X. Wu, “Unifying Large Language Models and Knowledge Graphs: A Roadmap,” IEEE Transactions on Knowledge and Data Engineering, vol. 36, no. 7, pp. 3580-3599, July 2024, doi: 10.1109/TKDE.2024.3352100.