Fnu Neha[1] and Deepshikha Bhati[1]

[1]Dept. of Computer Science, Kent State University

August 26, 2025

Traditional RAG vs. Agentic RAG: A Comparative Study of Retrieval-Augmented Systems

Fnu Neha
Dept. of Computer Science
Kent State University
Kent, OH, USA
neha@kent.edu

Deepshikha Bhati
Dept. of Computer Science
Kent State University
Kent, OH, USA
dbhati@kent.edu

***Abstract*—Retrieval-Augmented Generation (RAG) enhances large language models (LLMs) by integrating external knowledge retrieval to improve factual reliability. Traditional RAG employs a fixed, single-pass retrieval process, limiting its ability to handle multi-step reasoning, adaptive queries, and heterogeneous data sources. Agentic RAG extends this framework with autonomous agents that plan, iterate retrieval, integrate tools, and reason over intermediate results. This paper presents a comprehensive comparison of Traditional and Agentic RAG in terms of architecture, capabilities, evaluation metrics, and operational challenges. In addition to synthesizing representative systems, we provide a side-by-side analysis of comparative limitations, failure modes, and corresponding mitigations, mapping domain-specific applications across established and emerging fields. We also outline governance recommendations and propose future research directions, including graph-augmented, multimodal, human-in-the-loop, and domain-specialized Agentic RAG frameworks with standardized model cards. These insights offer both a technical and practical foundation for designing more adaptive, trustworthy, and context-aware retrieval-augmented systems.**

***Index Terms*—Retrieval-Augmented Generation (RAG), Agentic RAG, Large Language Models (LLMs), Multi-Agent Systems, Natural Language Generation (NLG), Knowledge Retrieval, Multi-Step Reasoning, Tool Integration, Adaptive Query Reformulation**

I. INTRODUCTION

Large Language Models (LLMs) have advanced significantly in generating coherent and contextually relevant text. However, their reliance on static parametric knowledge leads to factual inaccuracies and hallucinations in knowledge-intensive tasks [1]. These limitations reduce their reliability in domains where accuracy, verifiability, and traceability are essential.

Retrieval-Augmented Generation (RAG) addresses these shortcomings by coupling LLMs with external information retrieval systems [2]. In a standard RAG pipeline, a user query is used to retrieve relevant documents from a knowledge base, and the retrieved content is incorporated into the model's response. This approach improves data-supported reasoning, broadens domain coverage, and offers transparency by linking outputs to source material. Despite these benefits, Traditional RAG systems follow a fixed, single-pass retrieval process. They are highly sensitive to the quality of the initial query and lack mechanisms for iterative refinement and error correction [3]. Such static designs limit performance in tasks requiring multi-step reasoning, dynamic adaptation, and integration of heterogeneous data sources.

Agentic RAG extends this framework by integrating agent-based mechanisms. Here, an agent is a computational entity that perceives its environment, makes autonomous decisions, and takes actions toward achieving specific goals [4]. Unlike Traditional RAG components, agents are goal-driven and capable of adapting to context, invoking external tools, reasoning over intermediate steps, and coordinating tasks across multiple stages. These capabilities include: (1) perception (understanding the input query, objectives, and contextual signals), (2) reasoning (decomposing complex queries into subtasks and deciding next best actions), (3) action (executing retrieval, tool invocation, or generation), (4) reflection (evaluating intermediate results and adjusting strategies), (5) adaptation (learning from past outputs and feedback), and (6) collaboration (working with other agents in multi-agent settings). Depending on the system design, agents perform specialized roles such as *Planner* (breaking down queries into executable steps), *Retriever Agent* (performing adaptive or multi-hop retrieval), *QA Agent* (synthesizing fact-grounded answers), *Tool Agent* (invoking APIs or structured tools), *Verifier* (ensuring factual consistency), and *Memory Agent* (maintaining contextual memory to guide retrieval and generation).

By enabling multi-step workflows, iterative query refinement, and intermediate evaluation, Agentic RAG improves adaptability and resilience to initial query errors [5]. However, these capabilities introduce additional complexity in system design.

The shift from static RAG to Agentic, reasoning-enabled retrieval frameworks reflects a broader movement toward more adaptive AI systems. While Agentic RAG offers clear advantages, its added complexity, cost, and latency means it is not always the optimal choice. Thus, this paper compares Traditional and Agentic RAG to evaluate trade-offs in architecture, reasoning capability, and operational constraints, providing guidance for the design and deployment of retrieval-augmented systems. The key contribution of this paper is:

- Comparison of Traditional and Agentic RAG architectures.
- Analysis of challenges, their causes, and mitigation strategies, along with governance recommendations.

- Proposals for graph-augmented, multimodal, human-in-the-loop, and domain-specific Agentic RAG frameworks with standardized model cards.

The paper is organized as follows: Section 2 covers the background; Section 3 reviews related work; Section 4 describes functional capabilities and use cases; Section 5 outlines evaluation metrics and frameworks; Section 6 discusses challenges; Section 7 addresses ethics, security, and governance; Section 8 explores future directions; and Section 9 concludes the paper.

II. Background

A. Traditional RAG

Introduced by Lewis et al. (2020), Traditional RAG combines an external retrieval component with an LLM to enhance factual grounding. Its single-pass, stateless pipeline generally includes:

- **Query Encoding** – The input q is converted into a dense vector using a transformer-based encoder.
- **Document Retrieval** – A similarity search identifies the top-k relevant documents from a pre-indexed corpus.
- **Context Fusion** – Retrieved documents are concatenated with the query and supplied to the generator:
  - **RAG-Sequence**: Processes each document sequentially.
  - **RAG-Token**: Conditions token generation on retrieval at every step.
- **Response Generation** – An encoder–decoder model produces the final output.

Traditional RAG offers low latency and computational efficiency but is sensitive to initial query quality, lacks iterative refinement, and cannot adapt retrieval mid-generation.

B. Agentic RAG

Agentic RAG embeds retrieval within an iterative, reasoning-driven control loop executed by an LLM-based planner. This architecture enables multi-step reasoning, query reformulation, and integration of heterogeneous data sources. The core components include:

- **Planner/Controller Agent** – Decomposes the user request into sub-tasks or sequential goals.
- **Iterative Retrieval** – Executes retrieval for each subtask, evaluates evidence for sufficiency, and reformulates queries when necessary.
- **Tool Integration** – Uses APIs, databases, knowledge graphs, or multimodal pipelines.
- **Episodic Memory** – Maintains prior queries, retrieved evidence, and reasoning paths to improve context retention.
- **Response Synthesis and Verification** – Aggregates evidence into a draft, applies self-consistency or fact-checking modules, looks for stop condition and finalizes the response.

This architecture improves factual robustness and adaptability but increases latency and computational requirements.

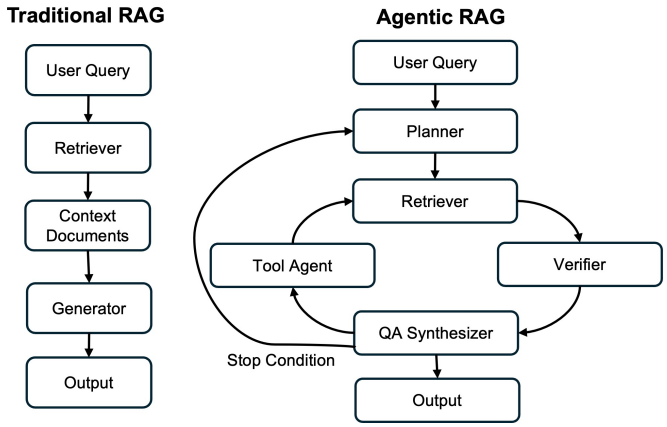

Fig. 1: Traditional RAG vs. Agentic RAG: Comparative architectures

Figure 1 presents the general architecture, highlighting the structural differences between Traditional RAG and Agentic RAG.

Table I summarizes the key distinctions between the two architectures.

TABLE I: Comparative Summary of Traditional and Agentic RAG Architectures

Feature	Traditional RAG	Agentic RAG
General Workflow	Query → Retrieve → Generate	Plan → Retrieve → Reflect → Act → Iterate
Reasoning Depth	Single-step, reactive	Multi-hop, iterative, reflective
Planning	None	Explicit planner modules or agents
Tool Use	Static retriever	APIs, calculators, search, database connectors
Multi-agent Support	Not supported	Specialized agents (Planner, QA, Verifier, Memory)
Adaptability	Low	High; supports evolving information needs
Memory	Stateless	Episodic or long-term memory
Data Sources	Static corpus	Multi-source integration
Query Handling	No refinement	Adaptive reformulation
Error Recovery	None	Multi-round correction
Latency	Low	Higher
Evaluation Difficulty	Low to moderate	High; complex behavior tracking

III. Related Work

This section reviews related work on Agentic RAG, highlighting key methodologies and representative systems.

Nguyen et al. proposed MA-RAG, a multi-agent framework with planner, step definer, extractor, and QA agents for query disambiguation, evidence extraction, and answer synthesis [6]. Agents are invoked selectively, enabling interpretable, fine-tuning-free reasoning.

Xiong et al. introduced RAG-Gym, an optimization platform for Agentic RAG through prompt engineering, actor tuning,

and critic training [7]. A reasoning-reflective agent with direct preference optimization improves intermediate reasoning quality.

Zhang et al. proposed ReasonRAG, which replaces fixed retrieval workflows with dynamic retrieval, iterative refinement, and adaptive task handling [8]. It applies process-level rewards for query generation, evidence extraction, and answer synthesis, improving stability, efficiency, and reducing costs. Using the RAG-ProGuide dataset, ReasonRAG outperforms baselines while requiring far fewer training instances.

Luo et al. introduced Graph-R1, a GraphRAG variant using lightweight knowledge hypergraphs and reinforcement learning for multi-turn retrieval, enhancing reasoning accuracy and generation quality [9].

Maragheh et al. proposed ARAG, apersonalized recommendation framework using multi-agent collaboration to capture dynamic user preferences [10].

Asai et al. introduced Self-RAG, an adaptive retrieval with self-reflection tokens for content evaluation, improving factuality and citation accuracy in long-form responses [11].

These works showed the versatility of Agentic RAG across domains, with variations in agent specialization, reasoning strategies, and optimization methods. Limitations, summarized in Table II, include coordination complexity, dataset dependence, computational overhead, and latency.

TABLE II: Limitations of Representative Agentic RAG Methods

Method	Key Limitations
MA-RAG [6]	Requires coordination among multiple agents, increasing organization complexity and latency; sensitive to prompt quality; prone to cascading errors from upstream outputs.
RAG-Gym [7]	Requires extensive experimentation for optimal configurations; critic training adds computational overhead; optimization gains are task-specific.
ReasonRAG [8]	Depends on the RAG-ProGuide dataset, which does not capture all domain-specific nuances; process-level reward design is complex; struggles with unstructured or noisy intermediate steps.
Graph-R1 [9]	Additional cost for building and maintaining knowledge hypergraphs; performance depends on quality of graph construction; less efficient in domains with rapidly changing or sparse structured data.
ARAG [10]	Domain-specific to personalized recommendation; multiple specialized agents increase complexity; requires high-quality, privacy-compliant user data.
Self-RAG [11]	Reflection tokens require model adaptation and calibration; causes excessive retrieval or self-reflection; increases inference latency and computational cost for long-form generation.

IV. Functional Capabilities and Use Cases

The capabilities of Traditional RAG and Agentic RAG differ in their approach to retrieval, reasoning, and adaptability. Traditional RAG is well-suited for straightforward retrieval tasks, whereas Agentic RAG excels in complex, multi-step, and evolving scenarios that require tool usage, memory, and adaptive planning.

Table III summarizes domain-specific applications, contrasting the strengths of Traditional and Agentic RAG across both established and emerging fields.

TABLE III: Domain-specific capabilities of Traditional RAG vs. Agentic RAG

Domain	Traditional RAG	Agentic RAG
Healthcare	Guideline retrieval	Multi-step diagnosis with longitudinal EHR, reasoning, and medical APIs
Finance	Static answer generation	Fraud detection, calculations, and multi-source data integration
Education	One-shot answers	Personalized tutoring with memory, adaptive feedback, and context-aware retrieval
Enterprise	Document search	Dynamic document routing, meeting summarization, and source verification
Legal	Statute lookup	Case analysis, legal reasoning, and citation tracing
Cybersecurity	Vulnerability database lookup	Threat analysis combining logs, Common Vulnerabilities and Exposures (CVE) data, and sandboxing tools
Scientific R&D	Retrieval of research papers	Hypothesis validation via datasets, simulations, and experimental results
E-commerce	Product catalog search	Personalized recommendations with inventory and competitor price checks
Government	Policy document retrieval	Impact analysis with real-time legislative updates and demographic projections
Customer Support	FAQ answering from static KB	Multi-turn troubleshooting with API calls and contextual personalization
Climate Modeling	Retrieval of historical climate data	Integration of satellite feeds, IoT sensors, and predictive climate simulations
Space Research	Space mission documentation lookup	Mission planning with real-time telemetry, orbital simulations, and anomaly detection
Manufacturing	Equipment manual retrieval	Predictive maintenance using sensor data, supply chain integration, and automated scheduling

Expanding the comparison to include emerging domains such as climate modeling, space research, and manufacturing highlights Agentic RAG's versatility beyond commonly discussed areas like healthcare and finance. These domains demand real-time, high-dimensional, and multi-source data integration—challenges that Traditional RAG struggles to address. For example, climate modeling benefits from continuous integration of satellite feeds, IoT sensor data, and simulation outputs; space research relies on real-time telemetry and anomaly detection; and manufacturing uses predictive maintenance by combining sensor analytics, supply chain data, and operational constraints.

V. Evaluation Metrics and Frameworks

A. Traditional RAG Metrics

The evaluation of Traditional RAG models relies on a combination of Natural Language Generation (NLG) metrics and factuality metrics. These assess lexical overlap, semantic correctness, and factual alignment with reference data. Key metrics include:

- **BLEU (Bilingual Evaluation Understudy):** Evaluates n-gram precision between the generated output and the reference text, applying a brevity penalty to discourage overly short outputs.
- **ROUGE (Recall-Oriented Understudy for Gisting Evaluation):** Measures recall-oriented overlap between generated and reference texts. ROUGE-L, based on the Longest Common Subsequence (LCS), captures fluency and sentence-level structural similarity.
- **Exact Match (EM):** Measures the percentage of predictions that exactly match the reference answer. EM is strict, as it does not account for synonyms or paraphrasing, making it suitable for tasks where a single correct answer format is expected.
- **FactScore:** Calculates the proportion of correct factual statements in the output compared to a verified reference source. This metric is particularly important in high-stakes domains such as medical or legal document generation.

Together, these metrics provide a balanced evaluation of both linguistic quality (BLEU, ROUGE) and factual reliability (EM, FactScore).

B. Agentic RAG Metrics

Agentic RAG evaluation must assess not only output quality but also reasoning, tool usage, and multi-step execution. Key metrics include:

- **Plan Fidelity:** Measures how closely execution follows the generated task plan, indicating adherence to structured reasoning steps.
- **Tool Success Rate:** Evaluates accuracy and effectiveness of API or tool calls, including correct parameters and result integration.
- **Reasoning Traceability:** Assesses clarity, depth, and correctness of reasoning steps, supporting interpretability and error analysis.
- **Task Completion Rate:** Percentage of tasks where the agent achieves the end goal across multi-step processes.
- **Process Reward:** RL-derived score for reasoning path optimality, encouraging efficiency and accuracy.
- **Cost vs Quality Tradeoff:** Balances computational resources (e.g., API calls, latency) against output reliability, aiding large-scale optimization.

VI. Challenges

The transition from Traditional RAG to Agentic RAG mitigates issues like retrieval noise and static corpus limitations but introduces challenges in planning, tool integration, and system complexity. Traditional RAG's constraints are mostly due to static query formulation and fixed knowledge sources, while Agentic RAG must manage multi-step coordination, real-time data integration, and security compliance.

Table IV summarizes the limitations of Traditional and Agentic RAG, highlighting both performance gains and operational risks. These high-level limitations are further linked to specific failure modes in Table V, which provides examples and corresponding mitigation strategies.

TABLE IV: Comparative limitations of Traditional RAG and Agentic RAG

Aspect	Traditional RAG	Agentic RAG
Retrieval Noise	High due to static, single-shot queries.	Lower via iterative refinement, planner coordination, and multi-hop retrieval.
Tool Failures	N/A — no external tool usage.	Tool chains failure; needs fallback and redundancy mechanisms.
Planning Errors	N/A — no explicit planning.	Suboptimal decomposition can derail execution; mitigated by plan verification.
Explainability	Moderate—steps are linear and traceable.	Lower due to layered reasoning and multi-agent orchestration; requires detailed logs.
Security/Access Control	Static corpus; no dynamic permissions.	Runtime queries respect real-time ACLs but need strong governance.
Content Ops Bottleneck	Corpus quality depends on periodic updates.	Continuous updates via document-management pipelines.
Training Complexity	Low—simple retriever and generator tuning.	High—may require supervised traces, RL rewards, and multi-agent tuning.

VII. Ethics, Security, and Governance

Agentic RAG systems use more tools and outside data, which increases security risks and compliance challenges in many domains. To reduce these risks, it is recommended to:

1) give tools only the access they need and keep detailed audit logs;
2) use automated redaction and, when possible, on-premises retrieval for sensitive information;
3) add citation checks and use do-not-answer rules when evidence is weak;
4) provide dataset and model cards that show data sources, known biases, and possible failure modes; and
5) require human review for outputs in high-stakes or safety-critical situations.

VIII. Future Directions

The evolution of RAG toward Agentic architectures opens several promising research and development avenues. The potential directions include:

- **Graph-Augmented Agentic RAG:** Integrating knowledge graphs, ontologies, and temporal-causal structures

TABLE V: Common failure modes in RAG systems and corresponding mitigations

Failure Mode	Example	Mitigation
Retrieval noise	Inclusion of irrelevant passages that compromise answer quality	Hybrid BM25 + dense retrieval; query reformulation; novelty penalty; per-chunk citation
Temporal drift	Outdated medical dosage conflicting with current clinical guidelines	Time-aware indices; recency filters; temporal knowledge graphs
Cascading reasoning error	Early subplan error propagating through subsequent reasoning steps	Plan verification; branch-and-bound search; verifier agent veto
Tool misuse	Invocation of API with invalid parameters	Schema-constrained tool calls; automated unit testing; tool success scoring
Citation hallucination	Fabricated publication identifiers or URLs in generated output	Strict retrieval-before-generation; citation verification agent
Privacy leakage	Presence of protected health information in prompts or logs	Automated redaction; on-premises retrieval; access control with audit logging
Speculative over-retrieval	Excessive retrieval hops leading to unnecessary cost and latency	Budget-constrained search; marginal-utility stopping rules

into the Agentic loop to improve reasoning over sequences, dependencies, and event causality. Such systems could dynamically expand or prune knowledge graphs during task execution, enhancing precision in domains like clinical diagnostics, supply chain analytics, and legal precedent analysis.

- **Multimodal Agentic Systems:** Extending Agentic RAG to process and reason over heterogeneous inputs such as text, images, audio, tabular data, and structured signals. This includes integrating medical imaging with EHR data, or satellite imagery with sensor feeds for climate monitoring, enabling richer context fusion.
- **Human-in-the-loop Oversight:** Embedding domain experts within the execution loop for real-time validation and correction in high-stakes contexts such as clinical decision support, legal reasoning, and financial compliance. This hybrid approach can mitigate risks from erroneous automated actions.
- **Domain-specialized Agentic Frameworks:** Creating domain-optimized Agentic RAG variants (e.g., *Med-RAG* for healthcare, *EduAgent* for education, *FinRAG* for finance) with specialized retrievers, ontologies, tool stacks, and evaluation metrics for their respective knowledge ecosystems.
- **Model Cards for Agentic RAG:** Developing standardized documentation formats that detail agent roles, behavior logs, decision policies, tool stacks, data provenance, and known limitations. These *Agentic model cards* would enhance transparency, reproducibility, and trust in enterprise and regulatory environments.

IX. Conclusion

This paper presents a comparative analysis of Traditional and Agentic RAG, examining their architectures, functional capabilities, evaluation approaches, and operational challenges. Traditional RAG offers efficiency and low latency for straightforward retrieval tasks but is limited by static queries and fixed knowledge sources. Agentic RAG enhances adaptability, reasoning depth, and multi-source integration, mitigating issues such as retrieval noise and the absence of iterative refinement, while introducing added complexity in coordination, tool reliability, and explainability. In high-stakes domains, ethical, security, and governance considerations remain critical. Future research should focus on dynamic knowledge graph integration, multimodal reasoning, human-in-the-loop mechanisms, and domain-optimized frameworks, supported by standardized evaluation protocols and transparent model documentation. Hence, these insights can also inform the design of retrieval systems beyond RAG, enabling autonomous multi-agent coordination for more adaptive, trustworthy, and context-aware performance across domains.

References

[1] L. Huang, W. Yu, W. Ma, W. Zhong, Z. Feng, H. Wang, Q. Chen, W. Peng, X. Feng, B. Qin *et al.*, "A survey on hallucination in large language models: Principles, taxonomy, challenges, and open questions," *ACM Transactions on Information Systems*, vol. 43, no. 2, pp. 1–55, 2025.

[2] P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Küttler, M. Lewis, W.-t. Yih, T. Rocktäschel *et al.*, "Retrieval-augmented generation for knowledge-intensive nlp tasks," *Advances in neural information processing systems*, vol. 33, pp. 9459–9474, 2020.

[3] E. Martell and A. Prellner, "Closed loop retrieval-augmented generation (rag) for content-based recommendations in e-commerce," *LU-CS-EX*, 2025.

[4] T. Masterman, S. Besen, M. Sawtell, and A. Chao, "The landscape of emerging ai agent architectures for reasoning, planning, and tool calling: A survey," *arXiv preprint arXiv:2404.11584*, 2024.

[5] G. d. A. e Aquino, N. d. S. de Azevedo, L. Y. S. Okimoto, L. Y. S. Camelo, H. L. de Souza Bragança, R. Fernandes, A. Printes, F. Cardoso, R. Gomes, and I. G. Torné, "From rag to multi-agent systems: A survey of modern approaches in llm development," 2025.

[6] T. Nguyen, P. Chin, and Y.-W. Tai, "Ma-rag: Multi-agent retrieval-augmented generation via collaborative chain-of-thought reasoning," *arXiv preprint arXiv:2505.20096*, 2025.

[7] G. Xiong, Q. Jin, X. Wang, Y. Fang, H. Liu, Y. Yang, F. Chen, Z. Song, D. Wang, M. Zhang *et al.*, "Rag-gym: Systematic optimization of language agents for retrieval-augmented generation."

[8] W. Zhang, X. Li, K. Dong, Y. Wang, P. Jia, X. Li, Y. Zhang, D. Xu, Z. Du, H. Guo *et al.*, "Process vs. outcome reward: Which is better for agentic rag reinforcement learning," *arXiv preprint arXiv:2505.14069*, 2025.

[9] H. Luo, G. Chen, Q. Lin, Y. Guo, F. Xu, Z. Kuang, M. Song, X. Wu, Y. Zhu, L. A. Tuan *et al.*, "Graph-r1: Towards agentic graphrag framework via end-to-end reinforcement learning," *arXiv preprint arXiv:2507.21892*, 2025.

[10] R. Y. Maragheh, P. Vadla, P. Gupta, K. Zhao, A. Inan, K. Yao, J. Xu, P. Kanumala, J. Cho, and S. Kumar, "Arag: Agentic retrieval augmented generation for personalized recommendation," *arXiv preprint arXiv:2506.21931*, 2025.

[11] A. Asai, Z. Wu, Y. Wang, A. Sil, and H. Hajishirzi, "Self-rag: Learning to retrieve, generate, and critique through self-reflection," 2024.