Agentic Retrieval-Augmented Generation (RAG) Framework with Quadruple-Based Reasoning and Reinforcement Learning (RL) Optimization

Sumedha Arya

Independent Research, India

DOI: https://dx.doi.org/10.51584/IJRIAS.2026.110200073

Received: 21 February 2025; Accepted: 27 February 2026; Published: 12 March 2026

ABSTRACT

Retrieval-Augmented Generation (RAG) has emerged as an effective technique to reduce hallucinations in large language models (LLMs) but they follow a static retrieve-then-generate pipeline. This process is insufficient for complex financial question answering system that require multi-step reasoning, numerical precision, and factual verification. Therefore, in this research, we proposed an RL-Driven Agentic Multi-HyDE RAG framework designed to improve factual correctness and informativeness through structured reasoning and reinforcement learning optimization. The proposed methodology comprises of six major components: query diversification, hypothetical answer generation (HyDE), dense embedding-based retrieval, quadruple-based atomic knowledge representation, reinforcement learning-based evaluation, and tool-augmented refinement. Experimental evaluation on financial queries using Sentence Transformers, FAISS, and Mistral-7B-Instruct demonstrates that the framework achieves high factual alignment (faithfulness score = 1.0) while maintaining informativeness, without unnecessary calling of the external tools. The results indicate that integrating agentic reasoning, structured knowledge extraction, and reinforcement learning significantly overcomes hallucinations and improves reliability. The proposed architecture provides a scalable and robust solution for high-stakes financial question answering systems.

Keywords: Retrieval-Augmented Generation (RAG); Agentic AI; Reinforcement Learning; Hypothetical Document Embeddings (HyDE); Financial Question Answering

INTRODUCTION

As the use of LLMs are getting broader, RAGs have become an important method to improve their accuracy by reducing hallucinations. They connect LLMs with external knowledge sources, helping them to extract latest information stored outside the model. In financial domain, the information changes quickly as it is more dynamic in nature. However, even when using conventional RAG, the model can still produce hallucinations by answering that conflict with the actual evidence. For instance, a retrieved report clearly states that the company's earnings per share (EPS) was 100 dollars as of May 31, 2025. But the model incorrectly associated this value with May 15, 2025 and may give wrong financial values also, showing a temporal inconsistency.

Retrieval can be improved by using better embeddings, by using advanced methods such as Hypothetical Document Embeddings (HyDE). In HyDE, the LLM first generates a possible answer, converts it into an embedding, and then retrieves real documents similar to that generated answer, improving retrieval accuracy. More recently, Agentic RAG systems have been developed where RAG is linked with an AI Agent. Here, the LLM acts like a brain that can break complex questions into smaller parts, retrieve information step-by-step, use tools, and verify results before generating the final answer. This approach is important in many applications including finance, where questions may require analyzing multiple reports, earnings statements, and financial news.

Recent studies have attempted to reduce hallucinations in RAG systems using reinforcement learning. These approaches usually depend on human-annotated reference answers and provide simple binary reward signals as correct or incorrect. However, they face two major challenges. First, creating human-labeled answers is

expensive and time-consuming. Second, binary rewards are too coarse and do not provide detailed guidance, making model training less stable and less effective.

The paper is further divided into following sections; review of the literature review, research methodology, results analysis, conclusion and references.

LITERATURE REVIEW

RAG has emerged as a powerful technique for improving LLMs against hallucination by integrating external vector database or knowledge retrieval into the generation process. Early research primarily focused on improving LLM performance through document denoising techniques. They aim to reduce irrelevant or noisy contextual information form the data before generation. Such preprocessing technique significantly improves response accuracy by providing high quality contextual data.

Another major research direction involves in rewriting the user query. In this method, the original query of the user is reformulated into semantically enriched or diversified variants to improve retrieval effectiveness. This approach enhances recall and overcomes the complexity or ambiguity in user inputs. Complementing this, iterative retrieval mechanisms were proposed to refine retrieved evidence across multiple retrieval cycles for better accuracy.

To improve retrieval, researchers also introduced pre-retrieval query transformation. In this method, the query is improved before searching. One important technique is Hypothetical Document Embeddings (HyDE). Instead of directly using the query, the system first generates a hypothetical answer and then uses its embedding for retrieval. This changes the process from query-to-document matching by answer-to-answer similarity matching and improves the performance. Another method is multi-query generation, where multiple variations of the user query are created to capture different aspects of the information need [16]. This increases recall but may reduce precision if the generated queries are too similar [14].

The rapid advancement of LLMs has further improved the working of RAG systems. In particular, chain-ofthought reasoning (CoT) has demonstrated significant improvements in handling complexity by performing multi-step queries and encouraging stepwise logical reasoning during generation. These developments have made RAG systems more capable of addressing challenges in financial and analytical tasks.

Traditional RAG follows a simple retrieve-and-generate pipeline. However, complex queries require multi-step reasoning and dynamic information gathering. This limitation led to the development of Agentic RAG, where autonomous agents manage reasoning and tool usage. Financial RAG systems face unique challenges. These include handling very long reports, disambiguating similar sections, maintaining numerical precision, and meeting regulatory requirements. Even small numerical errors can have serious consequences.

More recently, reinforcement learning (RL) techniques have been incorporated to optimize RAG-based LLMs. These approaches typically rely on reference answers and reward systems to perform policy optimization. However, such methods do have certain limitations. First, they require costly human annotation to construct desired dataset. Second, reward system may provide unstable or insufficient optimization sue to reward hacking behaviour of LLMs.

The review highlights important techniques used In RAGs to improve its performance. Though powerful, but still, they do have issues to mitigate against hallucinations. Therefore, by combining the capacity of these techniques with new architecture is better.

RESEARCH METHODOLOGY

In this section, we highlighted the methodology used in this research by building an advanced RAG system. It comprises of query diversification, hypothetical answer generation, structured atomic reasoning, and rewarddriven verification to improve factual correctness and informativeness in financial question answering. The overall methodology comprises of six major phases. These are as follows:

1. Query Expansion

ISSN No. 2454-6194 | DOI: 10.51584/IJRIAS |Volume XI Issue II February2026

2. Hypothetical Answer Generation
3. Embedding and Retrieval
4. Quadruple-Based Deep Reasoning
5. Reinforcement Learning-Based Evaluation
6. Tool-Augmented Refinement

Query Diversification

Let the original user query be:

$$Q$$

The query is passed to a Large Language Model (LLM), which generates multiple semantically diverse reformulations:

$$\{Q_1', Q_2', Q_3\}$$

The purpose of diversification is to improve retrieval coverage and reduce semantic ambiguity.

Thus, the expanded query set becomes:

$$Q' = \{Q, Q_1, Q_2, Q_3\}$$

Hypothetical Answer Generation (HyDE)

For each diversified query Q_i, the LLM generates a hypothetical answer:

$$A_i = \text{LLM}(Q_i)$$

These hypothetical answers enrich semantic representation before retrieval.

The merged semantic representation is constructed as:

$$A_{merged} = \bigcup_{i=1}^{n} A_i$$

where $n = 4$(original + 3 diversified queries).

Embedding and Vector Retrieval

The merged text is converted into a dense embedding using a Sentence Transformer model:

$$\mathbf{e}_q = \text{Embed}(A_{merged})$$

This embedding is used to retrieve top-k similar documents from a vector database:

$$D = \text{TopK}(\mathbf{e}_q)$$

ISSN No. 2454-6194 | DOI: 10.51584/IJRIAS |Volume XI Issue II February2026

where:

- $D = \{d_1, d_2, \ldots, d_k\}$
- $k$is the retrieval size.

Quadruple-Based Atomic Knowledge Representation

The retrieved documents are passed to an AI Agent for structured reasoning.

Each fact extracted from the documents is represented as a Quadruple Atomic Knowledge Unit (AKU):

$$(\text{Entity}, \text{Metric}, \text{Value}, \text{Timestamp})$$

Each quadruple represents a minimal factual statement.

Example:

$$(\text{ABC Bank}, \text{Profit Before Tax}, \text{Decline}, \text{March 2025})$$

Let the agent produce $k$atomic units:

$$\{s_1, s_2, \ldots, s_k\}$$

The AI Agent performs:

- Exploration (analyzing retrieved documents)
- Reasoning (deriving structured facts)
- Verification (checking factual consistency with evidence)

Dual-Model Framework

In addition to the AI Agent, a second baseline LLM generates a response directly from the original query:

$$A_{base} = \text{LLM}(Q)$$

The baseline response is also decomposed into atomic units:

$$\left\{s_1^{(0)}, s_2^{(0)}, \ldots, s_{k_0}^{(0)}\right\}$$

where $k_0$represents the number of atomic units in the baseline response.

The agent-generated response and baseline response are compared using reinforcement learning rewards.

Reinforcement Learning-Based Evaluation

The reward mechanism consists of two components:

1. Faithfulness Reward
2. Informativeness Reward

ISSN No. 2454-6194 | DOI: 10.51584/IJRIAS |Volume XI Issue II February2026

Faithfulness Reward (r_f)

Objective

- Reward factually correct statements
- Penalize incorrect statements

Step 1: Error Counting

Let:

$$\text{score} = \sum_{i=1}^{k} I(s_i = 0)$$

where:

$$I(s_i = 0) = \begin{cases} 1 & \text{if fact } s_i \text{ is incorrect} \\ 0 & \text{otherwise} \end{cases}$$

This counts the total number of incorrect atomic units.

Step 2: Exponential Decay Reward

The faithfulness reward is defined as:

$$r_f = \frac{1}{e^{\eta \cdot \min(\text{score}, \gamma)}}$$

where:

- η= decay rate (penalty strength)
- γ= maximum penalty cap
- $\min(\text{score}, \gamma)$limits extreme penalties

Why Exponential?

If:

$$\text{score} = 0$$
$$r_f = \frac{1}{e^0} = 1$$

If errors increase:

$$r_f \to 0$$

Example:

If $\text{score} = 2$and $\eta = 1$:

$$r_f = \frac{1}{e^2} \approx 0.135$$

ISSN No. 2454-6194 | DOI: 10.51584/IJRIAS |Volume XI Issue II February2026

Thus, more mistakes lead to rapidly decreasing rewards.

Informativeness Reward (r_i)

Problem

A model may reduce errors by giving extremely short answers. For example, instead of providing 5 facts, the model may provide only 1 fact. To prevent this, we define:

$$r_i = \begin{cases} 1 & \text{if } k \geq k_0 \\ 0 & \text{otherwise} \end{cases}$$

where:

- k= number of atomic units in agent response
- k_0= number of atomic units in baseline response

Thus:

- If the agent provides at least as much information → reward = 1
- Otherwise → reward = 0

Final Combined Reward

The final reward is computed as:

$$r = \frac{r_f + r_i}{2}$$

This ensures the response is factually accurate and sufficiently detailed.

Decision and Tool-Augmented Refinement

A threshold $\tau$is defined.

If:

$$r \geq \tau$$

The AI Agent's response is accepted as final output.

If:

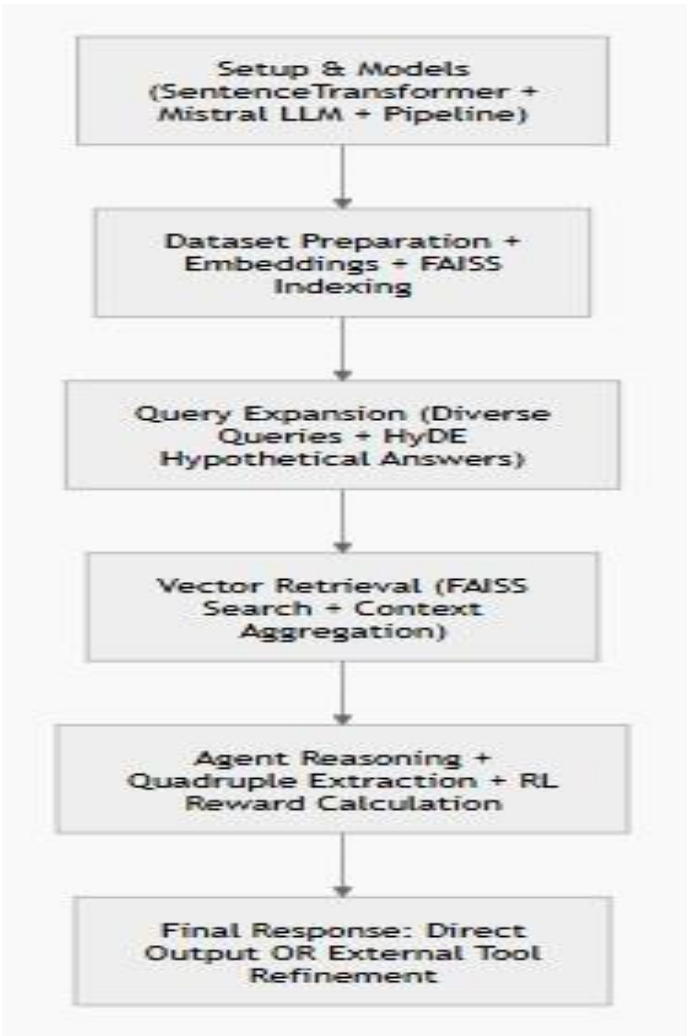
$$r < \tau$$

The agent activates auxiliary tools, for example additional retrieval, refined reasoning, or external knowledge modules to regenerate an improved response. Evaluation process is repeated until the reward exceeds the threshold, or maximum refinement iterations are reached.

The flow of the RL-Driven Agentic Multi-HyDE RAG Architecture is given in the figure 1.

ISSN No. 2454-6194 | DOI: 10.51584/IJRIAS |Volume XI Issue II February2026

Figure 1: RL-Driven Agentic Multi-HyDE RAG Architecture



RESULTS AND ANALYSIS

In this section, we evaluated the result obtained by RL-Driven Agentic Multi-HyDE RAG Architecture. All the core components of the architecture were utilized for this task. These are:

- Embedding Model: Sentence Transformers (all-MiniLM-L6-v2)
- LLM: Mistral AI (Mistral-7B-Instruct-v0.3)
- Vector Database: FAISS
- Hardware: CUDA-enabled GPU (cuda:0)

The reward threshold was set to 0.7, and the exponential faithfulness scaling parameter was $\eta = 1.0$.

The system first accepts the user financial query as “What happened to DCB Bank's profit before tax around May 2020?” and then generated 3 diverse reformulations of it. These variants enhanced semantic coverage and reduced retrieval bias. Subsequently, 4 hypothetical answers (HyDE responses) were generated in which 3 were from reformulated queries and 1 is from the original query. These were merged into a single enriched context representation for embedding. This step improved semantic density and retrieval robustness. Using FAISS L2 similarity search, the top-3 relevant documents were retrieved. These are:

1. DCB Bank reported a decline in profit before tax in May 2020 due to COVID-19 disruptions.
2. In Q1 2020, DCB Bank saw increased provisioning impacting profitability. 3. DCB Bank improved digital banking adoption during the pandemic period.

ISSN No. 2454-6194 | DOI: 10.51584/IJRIAS |Volume XI Issue II February2026

The retrieval stage successfully captured direct profit decline information, broader Q1 provisioning impact and pandemic adaptation measures. This demonstrates that multi-HyDE embedding enhanced contextual recall. Next, these results were shared with Agent to verify them based on contextual grounding. The agent response is processed using quadruple extraction with subject, relation, object and evidence as parameters. These structured facts were used to evaluate faithfulness.

Two reward components were computed. These are faithfulness reward and informativeness reward. The faithfulness reward is calculated as:

$$R_f = e^{-\eta \cdot errors}$$

Since no unsupported factual statements were detected, therefore, its value is:

$$R_f = e^{-1 \cdot 0} = 1.0$$

In informativeness reward, the agent-generated response contained equal or greater factual triples compared to baseline.

$$R_i = 1.0$$

The final reward, is calculates by taking the mean of faithfulness and informativeness rewards.

$$R = \frac{R_f + R_i}{2}$$
$$R = \frac{1.0 + 1.0}{2} = 1.0$$

Since:

$$R \geq 0.7$$

The reward threshold condition was satisfied, and external tool refinement was not triggered. The final response obtained is correctly stated as:

- DCB Bank experienced a decline in profit before tax in May 2020
- The decline was associated with COVID-19 disruptions
- Increased provisioning impacted Q1 profitability
- Digital banking adoption improved during the pandemic

Based on it, the answer was proved to be:

- Factually grounded
- Context-supported
- Multi-perspective
- Reward-validated

The key observations of the research based on the results achieved are:

1. Multi-HyDE improves retrieval grounding by enriching semantic space.

2. Quadruple-based verification ensures factual alignment.

3. Reward-based gating prevents hallucinated outputs.

4. The system avoided unnecessary external tool invocation due to high confidence score.

CONCLUSION

In conclusion, the proposed architecture demonstrates successfully on financial query by producing answer with highest faithfulness score and maintaining informativeness. It also achieved the maximum reward by successful integration of retrieval, reasoning, and reinforcement learning. This validates the effectiveness of the RL-Driven Agentic Multi-HyDE RAG Architecture for financial question answering system. Therefore, the major problem of LLMs as hallucination is likely to be solved by our proposed work.

REFERENCES

1. Chen, M., Li, T., Sun, H., Zhou, Y., Zhu, C., Wang, H., Pan, J. Z., Zhang, W., Chen, H., Yang, F., Zhou, Z., & Chen, W. (2025). ReSearch: Learning to reason with search for LLMs via reinforcement learning.
2. Eibich, M., Nagpal, S., & Fred-Ojala, A. (2024). ARAGOG: Advanced RAG output grading. arXiv preprint arXiv:2404.01037.
3. Fang, F., Bai, Y., Ni, S., Yang, M., Chen, X., & Xu, R. (2024). Enhancing noise robustness of retrievalaugmented language models with adaptive adversarial training. Proceedings of the ACL, 10028–10039.
4. Gao, L., Ma, X., Lin, J., & Callan, J. (2023). Precise zero-shot dense retrieval without relevance labels. Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics, 1762–1777.
5. Guo, D., Yang, D., Zhang, H., et al. (2025). DeepSeek-R1: Incentivizing reasoning capability in LLMs via reinforcement learning.
6. Guan, X., Zeng, J., Meng, F., Xin, C., Lu, Y., Lin, H., Han, X., Sun, L., & Zhou, J. (2025). DeepRAG: Thinking to retrieve step by step for large language models.
7. Hong, G., Kim, J., Kang, J., Myaeng, S.-H., & Whang, J. (2024). Why so gullible? Enhancing the robustness of retrieval-augmented models against counterfactual noise. Findings of NAACL 2024, 2474–2495.
8. Jaech, A., Kalai, A., Lerer, A., et al. (2024). OpenAI o1 system card.
9. Jiang, Z., Xu, F., Gao, L., Sun, Z., Liu, Q., Dwivedi-Yu, J., Yang, Y., Callan, J., & Neubig, G. (2023).
10. Active retrieval augmented generation. Proceedings of EMNLP, 7969–7992.
11. LangChain. (2023). Query transformations.
12. Li, X., Dong, G., Jin, J., Zhang, Y., Zhou, Y., Zhu, Y., Zhang, P., & Dou, Z. (2025). Search-o1: Agentic search-enhanced large reasoning models.
13. Li, Y., Luo, Q., Li, X., Li, B., Cheng, Q., Wang, B., Zheng, Y., Wang, Y., Yin, Z., & Qiu, X. (2025). R3RAG: Learning step-by-step reasoning and retrieval for LLMs via reinforcement learning.
14. Ma, X., Gong, Y., He, P., Zhao, H., & Duan, N. (2023). Query rewriting in retrieval-augmented large language models. Proceedings of EMNLP, 5303–5315.
15. Song, H., Jiang, J., Min, Y., Chen, J., Chen, Z., Zhao, W. X., Fang, L., & Wen, J.-R. (2025). R1-Searcher: Incentivizing the search capability in LLMs via reinforcement learning.
16. Yu, T., Zhang, S., & Feng, Y. (2024). Auto-RAG: Autonomous retrieval-augmented generation for large language models.
17. Zhang, T., Li, K., Luo, H., Wu, X., Glass, J. R., & Meng, H. M. (2024). Adaptive query rewriting: Aligning rewriters through marginal probability of conversational answers. Proceedings of EMNLP, 13444–13461.