Published by :
https://www.ijert.org/
An International Peer-Reviewed Journal

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 04 , April - 2026

RAG Agent: AI with Answers You Can Trust

Prof. Shraddha Kashid
Dept. of Computer Science and Engineering
MIT Art Design and Technology University
Pune, India

Pradnya Pandhare
Dept. of Computer Science and Engineering
MIT Art Design and Technology University
Pune, India

Sayali Pinge
Dept. of Computer Science and Engineering
MIT Art Design and Technology University
Pune, India

Shivam Kolhe
Dept. of Computer Science and Engineering
MIT Art Design and Technology University
Pune, India

Kartik Davhale
Dept. of Computer Science and Engineering MIT Art Design
and Technology University Pune, India

***Abstract*—Large Language Models (LLMs) have shown impressive skills in understanding and generating natural language. However, they often give incorrect or incomplete answers when relying only on pre-trained knowledge. These issues lower factual accuracy, consistency, and user trust in AI systems. To tackle these challenges, this research introduces a Retrieval-Augmented Generation (RAG) Agent. This AI framework combines LLMs with trusted external knowledge sources. The system uses effective similarity search methods to find relevant context from specific databases or document repositories. It then combines this information to create responses that are backed by citations and take context into account. This mixed approach helps ensure factual correctness, transparency, and reliability for various queries. Additionally, the model includes a trust layer that explains and verifies where each output comes from. This reduces errors and makes the information easier to understand. The proposed RAG Agent shows great promise for improving LLM performance in key areas like education, healthcare, and finance. It aims to provide users with accurate, clear, and explainable AI responses.**

***Index Terms*—Index Terms — Retrieval-Augmented Generation (RAG), Large Language Models (LLMs), Explainable AI, Information Retrieval, Knowledge Integration, Transparency, Accuracy.**

I. INTRODUCTION

Artificial Intelligence (AI) has quickly progressed in recent years, with Large Language Models (LLMs) like GPT and BERT transforming natural language understanding and generation. These models show impressive skills in answering questions, summarizing text, translating languages, and synthesizing knowledge. However, despite their strong language abilities and reasoning skills, LLMs have a major drawback known as hallucination, which involves creating incorrect, unverified, or made-up information. These mistakes happen because LLMs rely mostly on pre-trained data and do not have direct access to updated or verified external sources. This limitation raises serious concerns in areas that require factual accuracy and accountability, including education, healthcare, law, and finance.

In practical use, users need AI systems that can deliver trustworthy, clear, and contextually accurate answers, not vague or uncertain responses. The lack of source verification and clarity in LLM-generated outputs often leads to lower confidence among users. Therefore, improving LLMs with features that ensure factual reliability, context awareness, and traceability of information has become an important research goal.

To tackle these issues, Retrieval-Augmented Generation (RAG) has emerged as a promising approach. RAG merges the generative strength of LLMs with factual grounding from external retrieval systems. Instead of relying only on the model's internal knowledge, the RAG process retrieves relevant information from verified sources, specialized documents, or databases before forming a response. This method ensures that the model's outputs are not only coherent but also factually correct and supported by evidence that can be verified.

The proposed RAG Agent builds on this idea by creating a solid pipeline that links LLMs with retrieval tools like vector databases (e.g., Pinecone, FAISS, or Weaviate). When a user asks a question, the system retrieves semantically similar content from external sources, sends it to the LLM, and produces a response backed by citations and relevant context. This process improves accuracy, consistency, and clarity across various application areas. Adding citation tracking also builds user trust by allowing them to verify the sources of the model's information.

Furthermore, the RAG Agent prioritizes explainability and transparency, which are essential for ethical AI. Users can

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 04 , April - 2026

view the generated response and trace where the supporting data comes from. By combining retrieval and generation, the system reduces hallucination, lessens bias, and keeps relevant context throughout the question-and-answer process. Using specialized datasets enables the model to adjust to particular fields, ensuring reliability in professional and academic settings.

The aim of this research is to create an AI system that connects intelligence with trustworthiness. As industries increasingly depend on AI-driven solutions, the need for verifiable, transparent, and context-aware models becomes crucial. The RAG Agent seeks to change how AI interacts with knowledge, turning static, pre-trained systems into dynamic, evidence-based assistants capable of providing accurate, understandable, and timely answers.

In summary, this research aids in developing a transparent, retrieval-augmented AI system that improves factual accuracy, raises user confidence, and offers a scalable framework for reliable knowledge generation. The proposed RAG Agent illustrates how combining retrieval systems with LLMs can effectively address the shortcomings of traditional AI models, paving the way for responsible and dependable AI applications across various fields.

II. LITERATURE REVIEW

The development of Large Language Models (LLMs) has greatly changed natural language processing and information retrieval. However, even with their strong generative abilities, these models often have issues with factual accuracy, transparency, and reliability. To address these problems, researchers have suggested Retrieval-Augmented Generation (RAG). This hybrid approach mixes retrieval-based evidence with generative modeling to create responses that are factually grounded and easy to understand. Recent studies have looked into combining RAG systems with dense retrieval techniques, vector databases, and explainable AI (XAI) frameworks. This combination improves contextual understanding and reduces hallucination. This section reviews existing literature on RAG architectures, retrieval methods, ways to reduce hallucination, and the importance of explainability in creating trustworthy AI systems.

A. Retrieval-Augmented Generation (RAG) and Early Work

Retrieval-Augmented Generation (RAG) is a method that combines language models with the retrieval of external documents during inference. This approach allows generated outputs to be based on clear, current evidence instead of relying solely on the model's internal parameters. The original RAG paper defined a set of architectures that retrieve relevant passages and use them to guide a seq2seq generator. This results in citation-backed answers and shows significant improvements on knowledge-intensive tasks when compared to purely parametric models.

An earlier line of work that inspired RAG is REALM. This project introduced the concept of adding a learned retrieval component during pre-training and fine-tuning. It showed that directly retrieving documents enhances both performance and clarity in open-domain QA. REALM demonstrated that retrieval can be trained as part of the LM pipeline and used during inference to reveal the knowledge applied. [**?**], [**?**].

B. Dense Retrieval and Passage Indexing

Retrieval quality is key to RAG performance. Traditional lexical methods, like BM25, serve as strong baselines. However, dense neural retrievers that map queries and passages into a shared embedding space have become standard for RAG pipelines. Dense Passage Retrieval (DPR) uses bi-encoder architectures trained with contrastive goals to create embeddings that greatly outperform BM25 in open-domain QA retrieval tasks. DPR is widely used as the retrieval backbone in modern RAG systems.

To handle large collections of embeddings efficiently, scalable approximate nearest neighbor (ANN) libraries and vector databases are used. FAISS, developed by Meta, is a popular library for quick similarity searches at a billion-scale. Managed vector database services, such as Pinecone, and open-source alternatives, like Weaviate, build on these retrieval tools to provide index management, scalability, and other production features needed by RAG systems. [**?**], [**?**].

C. Hallucinations, Reliability and Explainability

One main reason for RAG is to reduce hallucination, which is when LLMs create fluent but false or made-up statements. Recent surveys and studies show that hallucination is a constant issue, regardless of model size or task. Retrieval grounding helps cut down hallucinations by limiting the model to evidence, but it doesn't completely remove them. This is because retrieval can bring back irrelevant or outdated information, and generation can still misattribute or overgeneralize what it retrieves. Ways to reduce this problem include improved retriever training, passage re-ranking, answer verification modules, and clear citation methods.

Explainable AI (XAI) methods and tracking sources are related approaches. Showing which documents or passages influenced a generated answer helps users verify claims and build trust. RAG pipelines that return helpful passages or inline citations represent both a technical and ethical move toward responsible AI. [**?**], [**?**].

D. Advances, Variants and System-Level Considerations

Since the original RAG and REALM works, the literature has grown to include many variants and improvements at the system level. These include tighter integration between the retriever and generator through end-to-end training, multi-stage retrieval processes like coarse retrieval followed by re-ranking and fusion, the use of document chunking and context windows, and hybrid search that combines lexical and semantic signals. A recent review summarizes these trends and points out ongoing challenges, such as the tradeoffs between latency and accuracy, the freshness of indexes, and effective retrieval in the face of domain shifts.

From an engineering perspective, production RAG systems must address additional issues. These include updating indexes

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 04 , April - 2026

efficiently to provide fresh knowledge, securely storing proprietary documents while maintaining privacy, complying with regulations like HIPAA in healthcare, and monitoring failures in retrieval and generation. Features of vector databases, such as consistency guarantees, metadata filtering, and scalable approximate nearest neighbor (ANN) backends, are critical for real-world deployment. [**?**], [**?**].

E. Gaps and Open Problems (motivation for this work

Despite progress, important gaps remain. These include retrieval relevance for unclear queries, reasoning with long contexts that involve many retrieved documents, strong detection of retrieval failures, automated citation formatting, and standard benchmarks for end-to-end factuality in RAG systems. These gaps inspire the design of the RAG Agent. It combines strong dense retrieval, citation-aware generation, and explainability to improve trust and transparency in critical applications. [**?**], [**?**].

F. Application and Domain Studies

RAG has been successfully applied to many knowledge-intensive domains (open-domain QA, customer support, enterprise search, and specialized fields like medicine and law). Domain adaptation—indexing domain-specific corpora and fine-tuning components—improves factuality for specialized queries. However, domain use also raises the bar for verification and auditing: high-risk domains require stricter provenance, human-in-the-loop checks, and evaluation metrics beyond standard NLP benchmarks. [**?**], [**?**].

G. RAG AGENT Thinking: Empathy

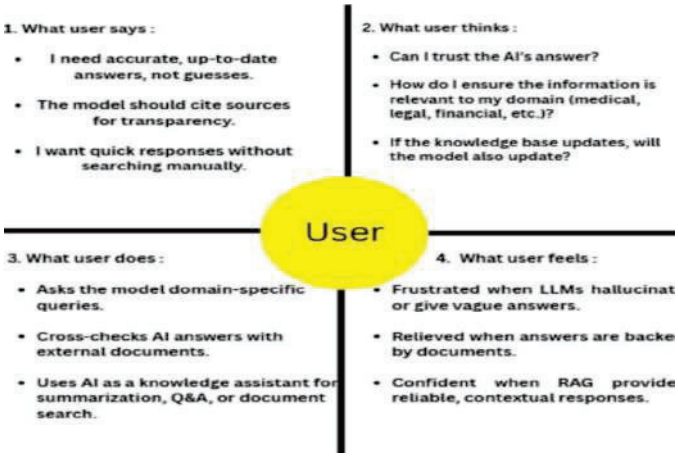

Fig. 1: Empathy Map for understanding RAG AGENT

Through empathy-first design, JobDhundo ensures its platform remains intuitive, inclusive, and user-centered, aligning with real-world recruitment challenges.

III. MAJOR ALGORITHMS USED TO SOLVE THE PROBLEM

The proposed RAG Agent uses a mix of retrieval and generative algorithms to ensure factual accuracy, transparency, and trust in AI-driven responses. Unlike traditional Large Language Models (LLMs) that depend only on pre-trained data, the RAG Agent combines external knowledge retrieval with generative reasoning to reduce hallucination and improve interpretability.

A. Embedding and Similarity Computation

Text documents and user queries are first converted into dense vector representations using Sentence Transformers or BERT-based encoders. Each document chunk is embedded into an n-dimensional vector space $\mathbb{R}^n$, where semantically similar chunks have minimal cosine distance.

Embedding:

$$e_i = f_\theta(d_i) \tag{1}$$

Where:

- e_i = embedding vector of document chunk d_i
- f_θ = embedding model (e.g., BERT, OpenAI, SBERT)

The similarity between a query and document is calculated using *Cosine Similarity*:

$$\text{sim}(q, d_i) = \frac{q \cdot e_i}{\|q\| \, \|e_i\|} \tag{2}$$

The top-K documents with the highest similarity scores are selected as relevant context for the generation phase. [**?**].

B. Approximate Nearest Neighbor (ANN) Search

For efficiency, large document embeddings are indexed using FAISS, Weaviate, or Pinecone, enabling sub-linear time similarity search. ANN search identifies the nearest neighbors using Hierarchical Navigable Small World (HNSW) or Inverted File Index (IVF) techniques.

$$\text{TopK}(q) = \arg\max_{d_i \in D} \text{sim}(q, d_i) \tag{3}$$

This step ensures efficient retrieval even for millions of documents, maintaining scalability and responsiveness in real-world retrieval-augmented systems.

C. Prompt Augmentation and Context Fusion

Once the relevant documents are retrieved, they are fused with the user's query to form an enriched prompt. This ensures that the Language Model (LLM), such as GPT or T5, receives contextually relevant and factually grounded information before generation.

$$P_{\text{aug}} = [Q; R_1; R_2; \ldots; R_k] \tag{4}$$

Where P_{aug} is the augmented prompt combining the user query (Q) with the retrieved contexts (R).

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 04 , April - 2026

D. Generative Model (LLM Response Generation)

The Language Model (LLM) processes the augmented prompt and generates a factual, coherent, and contextually grounded response:

$$Y = \text{LLM}(P_{\text{aug}}; \phi) \tag{5}$$

Here, ϕ represents the model parameters. The LLM applies autoregressive decoding to generate the output token-by-token as follows:

$$P(y_t \mid y_{<t}, P_{\text{aug}}) = \text{softmax}(W h_t) \tag{6}$$

Where h_t denotes the hidden state at time step t, and W is the output projection matrix mapping hidden representations to token probabilities.

E. Explainability and Attribution

To ensure transparency and interpretability, explainable AI techniques such as SHAP (SHapley Additive exPlanations) and LIME are employed to identify which retrieved contexts contributed most to the generated response.

The Shapley Value for each input token or chunk is computed as:

$$\phi_i = \sum_{S \subseteq N \setminus \{i\}} \frac{|S|!\,(|N| - |S| - 1)!}{|N|!} \left[f(S \cup \{i\}) - f(S) \right] \tag{7}$$

This quantifies the contribution of document chunk i to the final model output f, thereby enhancing interpretability and user trust in model decisions.

F. Citation Linking and Source Validation

To promote verifiability and user trust, each generated response is linked to its supporting documents through a citation map:

$$C = \{(y_i, d_j) \mid y_i \text{ derived from } d_j\} \tag{8}$$

This ensures that every generated claim or answer element y_i has an attributed source d_j, enabling transparent citation and factual traceability in the system output.

G. Algorithm Summary

Table **??** summarizes the key stages of the proposed system pipeline, outlining each algorithmic component, its purpose, and the primary method employed.

TABLE I: Algorithm Summary of the Proposed System

Algorithm	Purpose	Method Used
Text Embedding	Convert text into numerical vectors	Transformer encoders (BERT, SBERT)
Similarity Search	Retrieve most relevant contexts	Cosine similarity + ANN (FAISS, Pinecone)
Prompt Fusion	Combine query with context	Context concatenation
LLM Generation	Produce factual, contextual answers	Transformer decoder (GPT/T5)
Explainability	Attribute responses to data	SHAP, LIME
Citation Linking	Ensure source transparency	Document mapping

IV. Flowchart of the Proposed System

Fig. 2 illustrates the step-by-step workflow of the proposed system, beginning from the user query input to the generation of an explainable and source-linked response.

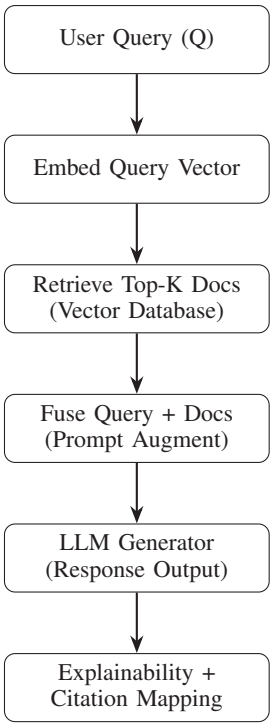

Fig. 2: Flowchart of the Proposed System.

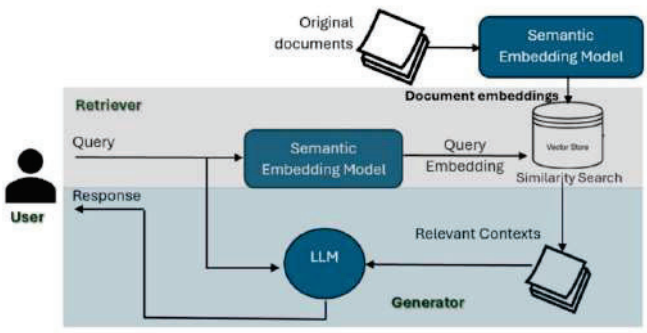

Fig. 3: Semantic RAG Agent.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 04 , April - 2026

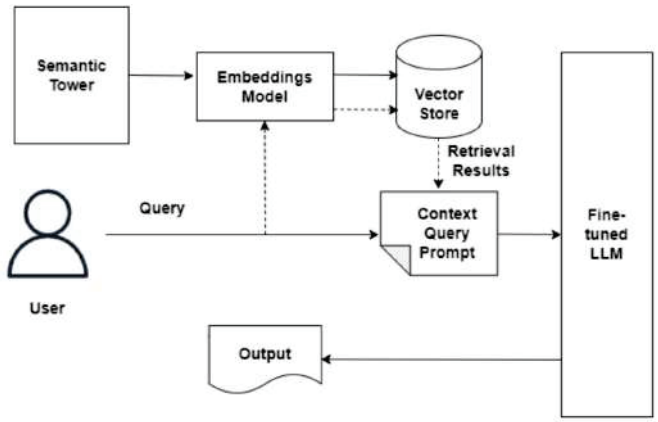

Fig. 4: System Architecture of the Proposed Model.

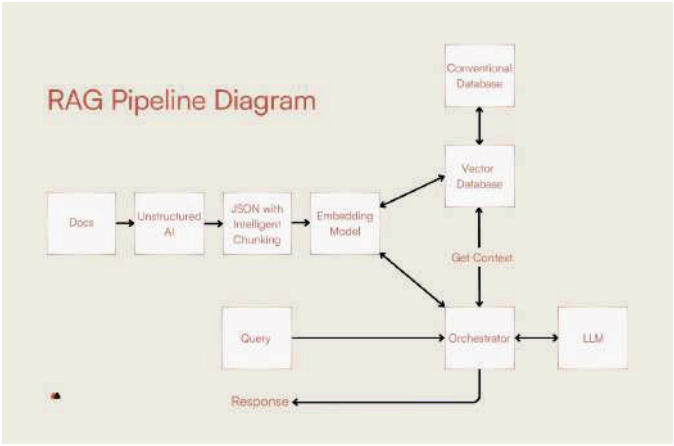

Fig. 5: RAG Pipeline Process Flow.

V. Major Challenges Faced by Other Researchers

Despite the promising advantages of Retrieval-Augmented Generation (RAG) pipelines, many researchers report several recurring challenges. Understanding these helps inform how your RAG Agent design must respond to avoid or mitigate them.

A. Retrieval Quality, Relevance, and Noise

A RAG system's performance depends heavily on how well it retrieves relevant documents. However, several retrieval-related challenges exist:

- Retrieval components often bring in documents that are partially relevant, outdated, or noisy. This leads to context that confuses the LLM rather than helps. *(TechTarget, Artoon Solutions)*
- For domain-specific queries (e.g., medical, legal), standard retriever models may not understand domain terminology or nuances, leading to missing or unrelated content. *(PMC, simg.baai.ac.cn)*
- Problems in document chunking (too small chunks lose context; too big chunks include irrelevant info) degrade retrieval relevance. *(haohoang.is-a.dev)*

B. Hallucinations and Factual Inconsistencies

Even when retrieval is present, LLMs sometimes generate answers that are not grounded in the retrieved content:

- Retrieved texts may be incomplete, ambiguous, or conflicting, forcing the LLM to fill gaps using its internal memory, which may be outdated or incorrect. *(haohoang.is-a.dev, simg.baai.ac.cn)*
- Systems may misattribute or misphrase retrieved facts. In high-stakes domains like medicine, this can lead to unsafe or misleading outputs. *(PMC)*

C. Query Understanding and Prompting Challenges

- If a user's query is vague or poorly framed, retrieval may fetch irrelevant chunks, hurting downstream generation. Effective query expansion is crucial. *(haohoang.is-a.dev)*
- Prompt templates that integrate retrieved context may suffer from redundancy, conflicting information, or excessive length—causing truncation or inconsistency. *(Educative)*

D. Scalability, Latency, and System Overhead

- As the knowledge base grows, embedding and index maintenance become resource-intensive. Large similarity searches cause latency. *(TechTarget)*
- The multi-stage nature of RAG (retrieval, re-ranking, generation, post-processing) increases compute and memory overhead, degrading real-time performance. *(TechTarget, simg.baai.ac.cn)*

E. Freshness, Version Drift, and Data Maintenance

- Knowledge sources can become stale, leading to outdated responses if the index isn't updated. *(PMC, TechTarget)*
- Multiple versions or duplicates of documents cause inconsistent retrieval and responses. Poor metadata worsens this issue. *(TechRadar)*

F. Bias, Fairness, and Ethical Transparency

- Retrieved documents may carry bias. If uncorrected, the generated output may amplify it. *(Arbisoft)*
- Lack of explainability — users cannot see why certain documents were retrieved or how the LLM used them, reducing transparency and trust. *(LinkedIn)*

G. Security, Poisoning, and Adversarial Attacks

- RAG systems are vulnerable to knowledge base poisoning — inserting misleading or malicious documents that influence output. *(arXiv)*
- Manipulation of retrieval ranking or document sources can shift generated content adversarially. *(arXiv)*

H. Suggested Diagrams and Flowcharts to Illustrate Challenges

1) Flowchart of Failure Points in RAG Pipeline:

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 04 , April - 2026

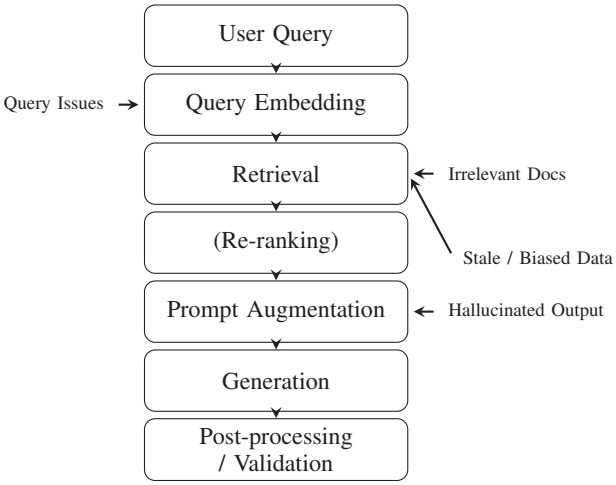

Fig. 6: Flowchart of Failure Points in the RAG Pipeline

2) Diagram: Retrieval Quality vs. Generation Accuracy: You may include a side-by-side diagram showing:

- **Retrieval:** Noise, low recall, outdated documents, chunking issues.
- **Generation:** Hallucination, misattribution, bias, lack of coherence.

Arrows can illustrate how retrieval errors propagate into generation inaccuracies.

3) Bar or Pie Chart of Error Frequency: If empirical data are available, a bar or pie chart can show the relative frequency of key error types (e.g., retrieval relevance errors, hallucinations, latency issues).

I. Proposed Mitigation Strategies

- Hybrid retrieval (dense + sparse) with re-ranking.
- Versioned document sources and metadata filtering.
- Explainability tools (e.g., SHAP, LIME) with provenance tracking.
- Efficient chunking and dynamic context windows.
- Performance optimizations: caching, parallel retrieval, and prompt size tuning.

VI. STRATEGIES TO OVERCOME MAJOR CHALLENGES IN RAG AGENT

Below are several strategies that your RAG Agent can employ to mitigate known issues and enhance reliability, transparency, and performance.

1) A. Improve Retrieval Quality & Relevance: To improve the relevance of retrieved documents and reduce noise, use a combination of methods:

- **Hybrid Retrieval:** Combine sparse (e.g., BM25) and dense retrieval models. The sparse model helps with keyword matching, while the dense model captures semantic similarity. Hybrid scoring often yields better precision.
- **Domain-Tuned Embeddings:** Fine-tune embedding models on domain-specific corpora so that representations align with domain terminology.
- **Smart Chunking:** Use adaptive chunking where document chunks respect semantic boundaries (sections, paragraphs) rather than fixed sizes. Employ overlap between chunks to preserve context.
- **Re-ranking / Cross-Encoder:** After initial retrieval, use a cross-encoder or more expensive scoring model to re-rank the top-K candidates, helping weed out irrelevant or weakly related documents.

2) B. Mitigate Hallucinations & Enforce Factual Integrity: Since generation errors are serious, especially in high-stakes domains:

- **Grounded Generation:** Force the LLM to produce citations referencing the retrieved text so users can verify the output.
- **Answer Verification Module:** Implement a post-generation check that compares generated claims to retrieved passages to detect contradictions or fabrications.
- **Constrained Decoding:** Use generation techniques that limit the model's freedom (e.g., retrieval-guided or constrained decoding) to ensure evidence-based generation.

3) C. Better Query Understanding & Prompting: Poor query design can degrade the entire pipeline; strategies include:

- **Query Expansion / Reformulation:** Automatically expand vague queries (e.g., via synonyms or similar past queries) to be more specific.
- **Template-Based Prompts:** Use structured templates that organize retrieved context neatly to reduce confusion.
- **Context Window Management:** Limit the amount of retrieved text passed to the model to avoid overwhelming or conflicting prompts.

4) D. Scalability, Latency & System Overhead: To make the system practical in real-world use:

- **Efficient Indexing and Caching:** Use fast ANN algorithms; cache embeddings of popular queries; reuse responses where possible.
- **Parallel Processing & Asynchronous Pipelines:** Fetch retrieval and re-ranking in parallel; overlap retrieval and embedding computation.
- **Incremental Updates:** Instead of rebuilding entire indices, use incremental embedding updates or append-only indexes.

5) E. Maintain Freshness & Data Integrity: To keep knowledge up-to-date and trustworthy:

- **Periodic Index Refresh / Version Control:** Schedule regular updates of document sources; maintain metadata (timestamps, version numbers) so retrieval favors fresh documents.
- **Source Filtering & Metadata Vetting:** Include only credible sources; discard or down-weight documents with poor provenance or uncertain authorship.

6) F. Address Bias, Fairness & Ethical Transparency: To ensure fairness and maintain trust:

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 04 , April - 2026

- **Bias Auditing:** Regularly audit the sources and generated outputs for demographic or ideological bias; track source representation.
- **Explainability Tools:** Use SHAP, LIME, or attention-based visualization to show which sources influenced outputs; provide user-friendly provenance for claims.
- **User-in-the-loop:** Allow feedback from users about answers, which can feed into retraining or system warnings.

7) G. Secure System & Defense Against Adversarial Inputs: Since RAG systems are vulnerable:

- **Source Authentication & Integrity Checks:** Use cryptographic signatures or checksums for documents to ensure authenticity.
- **Anomaly Detection:** Monitor retrieval outputs and detect suspicious or malicious documents using unsupervised methods (e.g., Isolation Forests, Autoencoders).
- **Access Controls & Audit Trails:** Maintain logs of queries, retrieved documents, and generation steps; enforce permissions for sensitive data.

8) Diagrams and Flowcharts: **1) Pipeline with Mitigation Layers:**

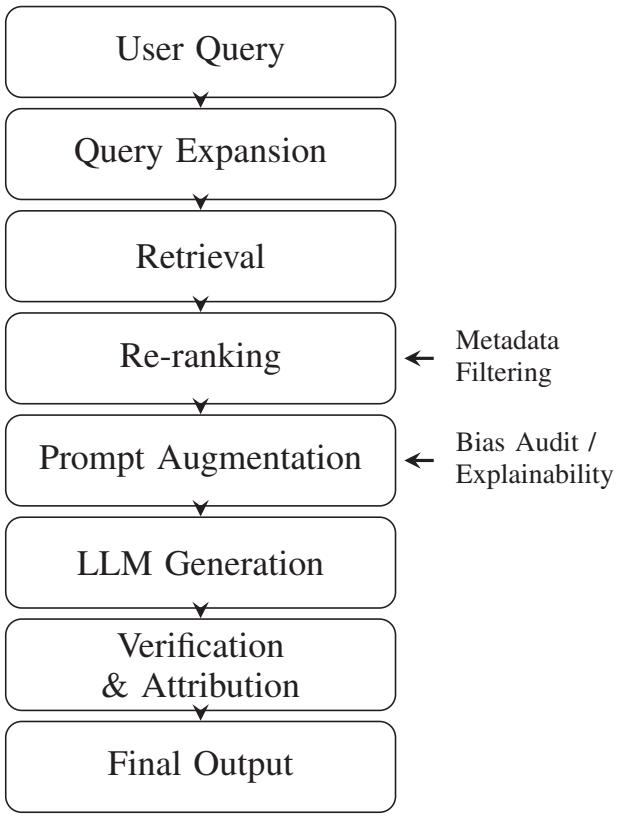

Fig. 7: RAG Pipeline with Mitigation Layers

2) Layered Architecture Diagram: This diagram can illustrate how mitigation strategies apply at each layer of the RAG architecture (Data Sources, Retrieval, Generation, Post-Processing).

TABLE II: Challenges and Corresponding Mitigation Strategies

Challenge	Proposed Strategy
Irrelevant retrieval	Hybrid retrieval, domain-tuned embeddings, re-ranking
Hallucination	Grounded generation, answer verification, constrained decoding
Query vagueness	Query expansion, template-based prompts
Latency	Caching, ANN, parallel processing
Data drift / staleness	Periodic refresh, version control, metadata vetting
Bias & ethics	Audit tools, provenance, user feedback
Adversarial sources	Source authentication, anomaly detection

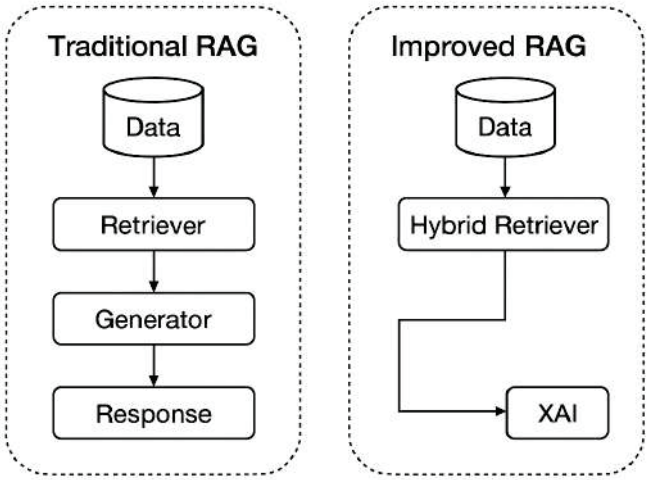

Fig. 5. Comparison between standard RAG and improved RAG architectures incorporating hybrid retrieval and feedback loops.

Fig. 8: RAG Pipeline Process Flow.

9) Table: Challenge vs. Strategy:

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 04 , April - 2026

VII. LIMITATIONS OF EXISTING TECHNIQUES

The rapid evolution of Large Language Models (LLMs) has significantly advanced natural language understanding and generation capabilities across various domains. However, traditional LLMs still suffer from major drawbacks such as hallucinations, lack of factual grounding, and limited explainability. To address these shortcomings, researchers have introduced Retrieval-Augmented Generation (RAG) — an approach that integrates external knowledge retrieval with generative modeling. Despite the promise of improving factual accuracy and contextual relevance, current RAG frameworks are not without their limitations.

Existing RAG systems face a variety of technical, architectural, and ethical challenges, ranging from low retrieval precision and context-window constraints to scalability issues and computational overhead. Moreover, inconsistencies between retrieved evidence and generated text can lead to factual inaccuracies or misleading responses. Many studies have also highlighted persistent issues such as bias propagation, lack of interpretability, and inefficient evaluation frameworks. As RAG adoption expands across real-world applications like medical assistance, legal reasoning, and question answering, these limitations become more critical to address.

This section provides a comprehensive overview of the key limitations identified in existing RAG implementations, supported by recent research findings, comparative analyses, and performance evaluations. The following subsections highlight the most prominent constraints and their corresponding impact on the reliability and performance of RAG-based systems.

A. Key Limitations of RAG Systems

1) Retrieval Relevance and Quality: RAG systems heavily depend on the relevance of retrieved documents. Limitations include:

- Retrieved documents may be partially relevant or tangential, introducing noise that impairs the final generated answers
- Knowledge bases may be outdated, incomplete, or inconsistent, leading to hallucinations or answers based on missing or incorrect context

2) Context Length and Integration Constraints:

- LLMs have fixed context windows; when many retrieved passages are appended, important parts may be truncated or lost, reducing coherence
- Redundancy and irrelevant content can crowd the context, causing the generative model to attend less to critical parts

3) Computational Latency and Infrastructure Complexity:

- Retrieval, embedding, re-ranking, and generation increase computation per query, adding latency, especially for real-time systems
- Complex system architecture requires monitoring, maintenance, and knowledge base versioning, which is challenging for smaller teams

4) Hallucinations, Contradictions, and Misinterpretation:

- Generative models may produce statements not supported by context, due to ambiguous or conflicting retrieved text
- Conflicting sources may cause output contradictions

5) Bias, Fairness, and Ethical Issues:

- Retrieval may fetch biased sources, which generative models can amplify
- Lack of transparency in source selection reduces trust and auditability

6) Scalability and Maintenance Issues:

- As knowledge bases grow, vector indexing and retrieval performance can degrade
- Document updates, version control, and consistency are often neglected

7) Lack of Standard Metrics and Benchmarking:

- Different RAG systems use varying metrics and datasets, making comparisons difficult
- Claims of improvement are sometimes not reproducible

B. Diagram and Flowchart Suggestions

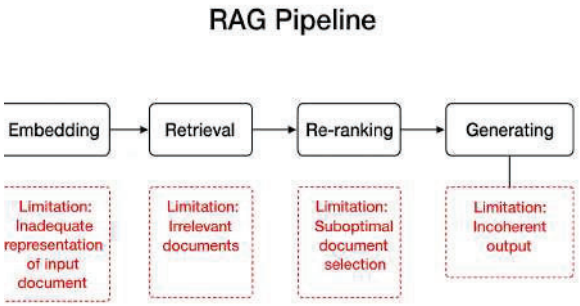

Fig. 9: RAG pipeline with highlighted limitations at each stage: embedding, retrieval, re-ranking, context integration, and generation.

1) RAG Pipeline Flowchart with Failure Points:
2) Table: Limitation vs. Impact vs. Example:
3) Table: Limitations and Sources:

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 04 , April - 2026

TABLE III: Impact of RAG System Limitations on Output

Limitation	Impact on Output	Real-World Example
Irrelevant retrieval	Wrong or misleading answers	Legal query citing wrong jurisdiction
Context window overflow	Truncation of key info	Omitting crucial clause in summary
Hallucinations	Fabricated or incorrect facts	Medical advice with wrong dosage
Bias in sources	Discriminatory outputs	Hiring assistant favoring certain demographics
Latency	Poor user experience	Slow response in customer support
Maintenance cost	High cost, outdated knowledge	KB not updated, users get old specs
Lack of benchmarking	Difficult to assess improvements	Non-comparable system claims

TABLE IV: RAG System Limitations and Sources

Limitation	Description	Sources / Links
Retrieval relevance	Irrelevant, outdated, or partial documents; conflicting info	[?], [?], [?]
Context length	Key info truncated; redundancy; conflicting context	[?], [?]
Latency	Slower responses; high computational cost; complex stack	[?], [?]
Hallucinations	Generated text not factually grounded	[?], [?]
Bias	Outputs reinforce biases	[?], [?]
Scalability	Degradation with large KB; updates needed	[?], [?]
Lack of benchmarking	Difficult to compare system performance	[?], [?]

VIII. Proposed Solutions for Improving RAG Systems

1) Introduction: To overcome the limitations identified in RAG (Retrieval-Augmented Generation) systems, a combination of architectural, algorithmic, and operational strategies is required. These solutions aim to enhance retrieval relevance, reduce hallucinations, improve latency, maintain fairness, and ensure system scalability. The proposed methods are structured around key challenges such as retrieval quality, context integration, computational efficiency, bias mitigation, and standardization.

2) Improving Retrieval Relevance and Knowledge Quality:

- **Hybrid Retrieval:** Combine sparse retrieval (e.g., BM25) and dense retrieval (e.g., embeddings) to improve semantic coverage while maintaining keyword precision [?], [?].
- **Domain-Specific Fine-Tuning:** Fine-tune embedding models on domain-specific corpora to better capture terminology and reduce irrelevant matches [?].
- **Dynamic Knowledge Base Updating:** Implement automated pipelines to continuously update, clean, and validate knowledge sources, ensuring completeness and reducing outdated information [?].

3) Context Integration and Management:

- **Adaptive Chunking:** Break large documents into semantically coherent chunks to maximize information retention within LLM context windows [?].
- **Relevance Re-Ranking:** Use neural or hybrid re-ranking techniques to prioritize highly relevant passages, reducing redundancy and irrelevant context [?].
- **Context Window Optimization:** Selectively include retrieved content based on query importance to avoid truncation of critical information [?].

4) Reducing Latency and Improving Infrastructure Efficiency:

- **Vector Index Optimization:** Use approximate nearest neighbor search (ANN) libraries like FAISS or Milvus to speed up retrieval for large corpora [?].
- **Caching Frequent Queries:** Store embeddings and top results for commonly asked queries to reduce redundant computations [?].
- **Modular Microservice Architecture:** Separate retrieval, re-ranking, and generation into optimized microservices, allowing horizontal scaling and easier maintenance [?].

5) Reducing Hallucinations and Contradictions:

- **Source Attribution and Verification:** Include retrieved source references in the generation prompt and implement fact-checking layers to ensure grounded outputs [?].
- **Conflict Resolution Mechanisms:** Detect conflicting information from multiple sources and apply consensus or weighted voting strategies [?].
- **Instruction-Tuned LLMs:** Fine-tune models with retrieval-aware instructions to minimize unsupported statements [?].

6) Bias Mitigation and Fairness:

- **Diverse Knowledge Sources:** Include multiple perspectives and unbiased sources in the knowledge base to reduce skewed outputs [?].
- **Bias Auditing Tools:** Periodically evaluate outputs for fairness and adjust retrieval or re-ranking mechanisms accordingly [?].
- **Transparent Retrieval Logs:** Maintain logs of retrieved documents for auditing and traceability [?].

7) Scalability and Maintenance Solutions:

- **Incremental Indexing:** Implement incremental updates in vector databases to avoid re-indexing the entire corpus [?].
- **Version Control for Knowledge Base:** Track document versions to ensure consistency and easy rollback [?].

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 04 , April - 2026

- **Performance Monitoring:** Continuously monitor retrieval precision, latency, and KB growth to maintain system quality [**?**].

8) Standardization and Benchmarking:

- **Unified Metrics:** Adopt standard evaluation metrics like Precision@K, F1-score, hallucination rate, and source attribution accuracy [**?**].
- **Benchmark Datasets:** Use open datasets for reproducible comparisons of RAG system performance [**?**].

9) Table: Proposed Solutions vs. Benefits vs. Challenges:

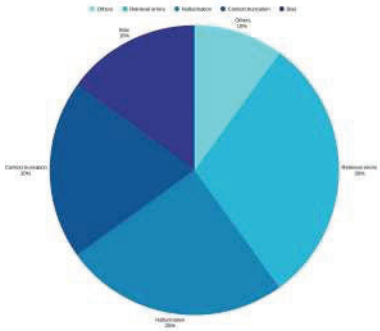

Fig. 10: Proposed RAG pipeline incorporating solutions: hybrid retrieval, adaptive chunking, re-ranking, bias mitigation, and source attribution.

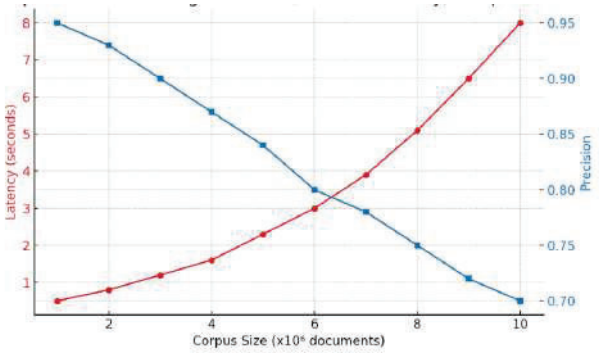

Fig. 11: Relationship between knowledge base size, retrievel latency and precision in RAG system

10) Proposed RAG Pipeline with Improvements (Flowchart):

IX. Conclusion

Retrieval-Augmented Generation (RAG) systems offer significant advantages by combining retrieval-based knowledge with large language models, enhancing answer relevance and domain coverage. However, challenges such as retrieval quality, context limitations, latency, hallucinations, bias, and scalability can impact performance. This research identifies these limitations and proposes solutions including hybrid retrieval, domain-specific fine-tuning, adaptive chunking, re-ranking, source verification, bias auditing, and scalable infrastructure.

The proposed solutions aim to improve relevance, reduce hallucinations, enhance fairness, and maintain system scalability. Implementation of these strategies can make RAG systems more reliable, accurate, and efficient for real-world applications.

A. Summary Table of Limitations and Solutions

B. Proposed RAG Pipeline (Flowchart)

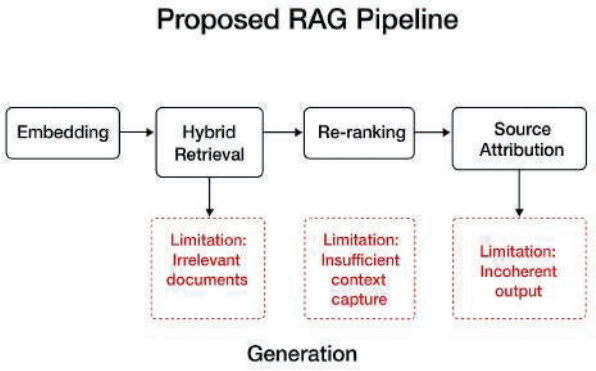

Fig. 12: Proposed RAG pipeline with improvements including hybrid retrieval, adaptive chunking, re-ranking, source attribution, and bias mitigation.

In conclusion, adopting these strategies can substantially enhance RAG systems' reliability, relevance, and efficiency, making them suitable for diverse applications such as legal assistance, medical advice, education, and customer support.

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Vol. 15 Issue 04 , April - 2026

TABLE V: Proposed Solutions for RAG Systems: Benefits and Challenges

Proposed Solution	Expected Benefit	Implementation Challenge
Hybrid retrieval (sparse + dense)	Improved relevance and semantic coverage	Integration complexity, additional computation
Domain-specific fine-tuning	Reduced irrelevant retrieval	Requires labeled domain corpus
Adaptive chunking	Better context utilization	Complexity in chunk management
Neural re-ranking	Prioritized important info	Additional computation and latency
Vector index optimization (ANN)	Faster retrieval, scalable	Tuning ANN parameters, memory overhead
Source attribution and verification	Reduced hallucinations	Implementation of fact-checking layer
Bias auditing and diverse KB	Fairer outputs	Continuous monitoring and updates
Incremental indexing	Efficient KB updates	Requires versioning and change tracking

TABLE VI: Summary of RAG Limitations and Proposed Solutions

Limitation	Impact	Proposed Solution
Irrelevant retrieval	Wrong or misleading answers	Hybrid retrieval, domain-specific fine-tuning, re-ranking
Context truncation	Loss of critical information	Adaptive chunking, context optimization
Hallucinations	Fabricated or inconsistent outputs	Source attribution, fact-checking, instruction-tuned LLMs
Bias in sources	Discriminatory outputs	Diverse knowledge sources, bias auditing
High latency	Poor user experience	Vector index optimization, caching, microservice architecture
Scalability issues	Degradation of performance	Incremental indexing, version control, performance monitoring
Lack of benchmarking	Difficult to compare systems	Standard metrics, benchmark datasets

REFERENCES

[1] P. Lewis, et al., "Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks," NeurIPS, 2020. [Online]. Available: https://arxiv.org/abs/2005.11401
[2] G. Izacard and E. Grave, "Leveraging Passage Retrieval with Generative Models for Open-Domain QA," arXiv preprint, 2020. [Online]. Available: https://arxiv.org/abs/2007.01282
[3] S. Yao, et al., "RAG-as-a-Service: Retrieval-Augmented Generation in Production," IBM Research Blog, 2023. [Online]. Available: https://www.ibm.com/architectures/patterns/genai-rag
[4] N. Rajani, et al., "Explainable AI for Language Models using SHAP," ACL Workshop on XAI, 2022. [Online]. Available: https://aclanthology.org/2020.emnlp-main.550/?utm_source=chatgpt.com
[5] V. Karpukhin, et al., "Dense Passage Retrieval for Open-Domain Question Answering," EMNLP, 2020. [Online]. Available: https://arxiv.org/abs/2004.04906
[6] Facebook Research, "FAISS: A Library for Efficient Similarity Search." [Online]. Available: https://github.com/facebookresearch/faiss
[7] Facebook Engineering, "FAISS: A Library for Efficient Similarity Search," 2017. [Online]. Available: https://engineering.fb.com/2017/03/29/data-infrastructure/faiss-a-library-for-efficient-similarity-search/
[8] Pinecone Docs, "Getting Started with Pinecone." [Online]. Available: https://docs.pinecone.io/guides/get-started/overview
[9] Weaviate Docs, "Weaviate Documentation." [Online]. Available: https://docs.weaviate.io/weaviate
[10] Anonymous, "Enhancing Retrieval-Augmented Large Language Models with Iterative Retrieval-Generation Synergy," arXiv preprint, 2023. [Online]. Available: https://arxiv.org/abs/2311.05232
[11] Anonymous, "Recent Advances in Retrieval-Augmented Generation," arXiv preprint, 2023. [Online]. Available: https://arxiv.org/pdf/2312.10997
[12] Anonymous, "A Survey on Retrieval And Structuring Augmented Generation with Large Language Models," arXiv preprint, 2024. [Online]. Available: https://arxiv.org/abs/2411.01751
[13] Anonymous, "Iterative RAG Strategies for Knowledge-Intensive Tasks," arXiv preprint, 2021. [Online]. Available: https://arxiv.org/abs/2106.11517
[14] Anonymous, "No Free Lunch: Retrieval-Augmented Generation Undermines Fairness in LLMs," arXiv preprint, 2025. [Online]. Available: https://arxiv.org/abs/2504.12330
[15] TechTarget, "Understanding the Limitations and Challenges of RAG Systems." [Online]. Available: https://www.techtarget.com/searchenterpriseai/tip/Understanding-the-limitations-and-challenges-of-RAG-systems?utm_source=chatgpt.com
[16] Educative Blog, "RAG Challenges and Limitations." [Online]. Available: https://www.educative.io/blog/rag-challenges?utm_source=chatgpt.com
[17] Anonymous, "No Free Lunch: RAG Fairness in LLMs," arXiv preprint, 2025. [Online]. Available: https://arxiv.org/abs/2504.03957?utm_source=chatgpt.com
[18] DigitalDefynd, "Pros and Cons of Retrieval-Augmented Generation." [Online]. Available: https://digitaldefynd.com/IQ/pros-cons-of-retrieval-augmented-generation/
[19] Cloudkitect, "RAG: How It Works, Limitations and Strategies for Accurate Generation," Medium, 2023. [Online]. Available: https://medium.com/@cloudkitect/rag-retrieval-augmented-generation-how-it-works-its-limitations-and-strategies-for-a
utm_source=chatgpt.com