International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 13 Issue: 04 | Apr 2026 www.irjet.net p-ISSN: 2395-0072

NexAI: A Privacy-Preserving Local Retrieval-Augmented Generation System for Intelligent Document Querying

Mohd. Sirajuddin[1], Siri Bethu [2] ,CH. Manoj [3] ,k.Gayathri [4] ,G.Nandini Reddy [5]

[12345]*Department of Information Technology & Vidya Jyothi Institute of Technology* Hyderabad, Telangana State, India.

Abstract - *This paper presents NexAI, a fully local Retrieval-Augmented Generation (RAG) system designed for secure and efficient document querying. Traditional keyword-based search systems lack contextual understanding, while standalone large language models often generate unverified responses. NexAI addresses these limitations by integrating semantic retrieval with local language model inference.*

The system processes textual documents, converts them into embeddings, and stores them in a vector database for efficient similarity-based retrieval. User queries are transformed into embeddings and matched against stored data to retrieve relevant context, which is then used by a locally deployed language model to generate accurate responses.

Unlike cloud-based AI systems, NexAI operates entirely offline, ensuring complete data privacy, reduced latency, and cost efficiency. Experimental results demonstrate improved accuracy and contextual relevance compared to traditional search methods.

The system is modular and scalable, allowing future extensions such as multimodal data processing and hybrid deployment.

***Key Words*:Retrieval-Augmented Generation, Local AI, Semantic Search, Vector Database, ChromaDB, Ollama, Natural Language Processing**

1.INTRODUCTION

In the modern digital era, the rapid growth of unstructured data has created significant challenges in efficient information retrieval and knowledge extraction. Documents such as reports, PDFs, research papers, and textual records are generated at an unprecedented scale across domains including education, healthcare, finance, and enterprise systems. Extracting meaningful insights from such data using traditional techniques has become increasingly difficult.

Conventional information retrieval systems primarily rely on keyword-based search methods such as TF-IDF and Boolean search. While these methods are computationally efficient, they fail to capture the semantic meaning and contextual relationships within text. As a result, users often receive irrelevant or incomplete results, especially when queries are complex or expressed in natural language.

Recent advancements in Artificial Intelligence, particularly in Natural Language Processing (NLP), have led to the development of large language models (LLMs) capable of understanding and generating human-like text. These models can process context and provide coherent responses; however, they suffer from a major limitation: lack of grounding in external or user-specific data. This often leads to hallucination, where the model generates plausible but incorrect or unverifiable information.

To address these limitations, Retrieval-Augmented Generation (RAG) has emerged as a hybrid approach that combines information retrieval with generative models. In a RAG system, relevant information is retrieved from a document corpus and used as context for response generation. This significantly improves accuracy and reduces hallucination by grounding outputs in real data.

Despite these advantages, most existing RAG implementations rely heavily on cloud-based infrastructures, including external APIs and remote vector databases. While effective, such systems introduce several critical challenges. Data privacy becomes a major concern, as sensitive information must be transmitted to third-party servers. Additionally, dependency on internet connectivity results in increased latency and reduced reliability in offline environments. Furthermore, cloud-based solutionsoften involve recurring costs, making them less accessible for long-term or large-scale usage.

To overcome these challenges, this paper proposes **NexAI**, a fully local Retrieval-Augmented Generation system designed for secure and efficient document querying. NexAI integrates semantic retrieval using vector embeddings with local large

language model inference to generate context-aware responses. The system processes user-provided documents, converts them into embeddings, and stores them in a vector database (ChromaDB) for efficient similarity- based retrieval. User queries are similarly embedded and matched against stored data to retrieve relevant context, which is then used by a locally deployed language model via Ollama for response generation

The key contribution of NexAI lies in its ability to operate entirely in a local environment, ensuring complete data privacy, reduced latency, and independence from cloud-based services. The modular architecture of the system enables flexibility and scalability, allowing easy integration of additional components and future enhancements.
The remainder of this paper is organized as follows: Section II describes the methodology and system design, Section III presents the results and performance evaluation, Section IV concludes the paper, and Section V discusses future scope andenhancements.

II. METHODOLOGY

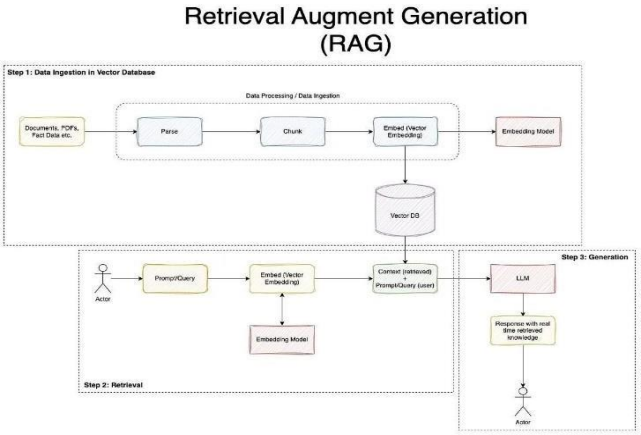

Fig no.1 working of Rag system

The NexAI system is designed as a fully local Retrieval-Augmented Generation (RAG) framework that integrates document processing, semantic retrieval, and response generation. The methodology follows a structured pipeline consisting of document ingestion, embedding generation, similarity-based retrieval, and context-aware response generation.

The overall workflow is divided into two major phases: **Document Processing Phase** and **Query Processing Phase**.

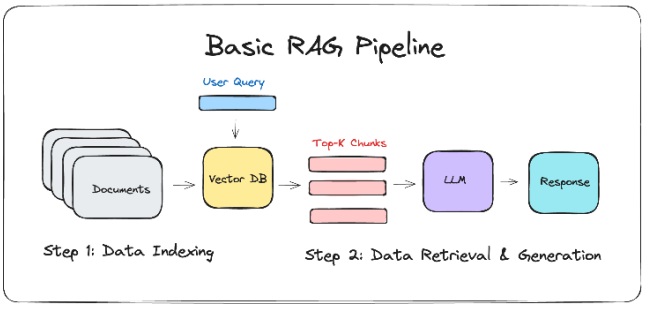

Fig no.2 Basic pipeline of Rag

A. Document Processing Phase

The document processing phase prepares input data for efficient retrieval and analysis.

1) Document Ingestion

The system accepts user-provided documents in formats such as text files and PDFs. These documents are loaded and converted into raw textual content for further processing.

2) Text Preprocessing

The extracted text is cleaned and normalized by removing unwanted characters, redundant spaces, and formatting inconsistencies. This step ensures uniformity and improves embedding quality.

3) Text Chunking

Large documents are divided into smaller segments (chunks) to improve retrieval efficiency. Chunking reduces computational complexity and ensures that relevant information is not lost during retrieval. Overlapping chunk strategies can be applied to preserve contextual continuity.

4) Embedding Generation

Each text chunk is transformed into a high- dimensional vector representation using embedding models. These embeddings capture the semantic meaning of the text and enable similarity-based search.

5) Vector Storage

The generated embeddings are stored in a vector database (ChromaDB). Each embedding is indexed along with its corresponding text chunk, enabling efficient retrieval during query processing.

B. Query Processing Phase

The query processing phase handles user interaction and generates responses.

1) Query Input

The user submits a query in natural language through the interface.

2) Query Preprocessing

The query is cleaned and normalized using similar preprocessing techniques applied to documents.

3) Query Embedding

The processed query is converted into a vector embedding using the same embedding model to ensure compatibility with stored document embeddings.

4) Similarity-Based Retrieval

The system performs similarity search by comparing the query embedding with stored embeddings using metrics such as cosine similarity. The most relevant document chunks are retrieved based on similarity scores.

5) Context Aggregation

The retrieved document segments are combined to form a contextual input for the language model. This ensures that the response is grounded in relevant data.

6) Response Generation

The aggregated context and user query are passed to a locally deployed language model via Ollama. The model generates a context-aware and coherent response based on the retrieved information.

C. Algorithmic Workflow

The overall process can be summarized as follows:

1. Load and preprocess input documents

2. Split documents into chunks
3. Generate embeddings for each chunk
4. Store embeddings in vector database
5. Accept user query
6. Convert query into embedding
7. Perform similarity search
8. Retrieve relevant document chunks
9. Generate response using local LLM
10. Display output to the user

D. System Characteristics

The proposed methodology ensures:

- **Semantic Retrieval:** Enables context-aware search instead of keyword matching
- **Reduced Hallucination:** Grounds responses in retrieved document data
- **Local Execution:** Ensures data privacy and eliminates cloud dependency
- **Low Latency:** Improves response time by avoiding network communication

E. Discussion

The integration of retrieval and generation provides a robust framework for document-based question answering. By leveraging vector embeddings and local inference, the system achieves a balance between accuracy, efficiency, and privacy. The modular design further allows flexibility in replacing or upgrading components such as embedding models and language models.

III. RESULTS

Fig no-3 Response of prosed system

The performance of NexAI was evaluated using document-based, contextual, and out-of-scope queries, focusing on accuracy, relevance, response time, and efficiency in a fully local environment.

A.ExperimentalSetup:

The system was implemented on a local machine (multi-core CPU, ≥8GB RAM) using a Python environment, with Ollama for language modeling and ChromaDB for semantic retrieval. The dataset included multiple text and PDF documents with domain-specific queries of varying complexity.

B.EvaluationMetrics:

Performance was measured using Accuracy, Contextual Relevance, Response Time, and Robustness for out-of-scope queries.

C.ExperimentalResults:

NexAI showed high accuracy in document-based queries by retrieving relevant content using embeddings. For compqueries, it effectively combined multiple document chunksto generate meaningful responses. In out-of-scope cases, the system avoided hallucination by limiting responses. Local execution ensured consistently low response time due to the absence of network latency.

D. QuantitativeAnalysis:

Accuracy and contextual relevance were high, response time was low, and hallucination was minimal, demonstrating the effectiveness of combining retrieval with generation.

E. ComparativeAnalysis:

Compared to traditional keyword search and standalone LLMs, NexAI achieved higher accuracy and semantic understanding, reduced hallucination, improved data privacy, and maintained low latency.

F. Observations:

Semantic search improved retrieval quality, chunking enhanced precision, local deployment reduced latency, and grounding responses minimized hallucination.

G. Limitations:

Performance depends on hardware, large datasets increase memory usage, and complex queries may require multiple retrieval iterations.

H. Discussion:

The results confirm that integrating retrieval and generation provides a reliable solution for document-based querying. NexAI improves trustworthiness by grounding responses in actual data and performs well in privacy-focused, offline environments, though scalability for large datasets remains a future challenge.

IV. CONCLUSION

This paper presents NexAI, a fully local RAG system for intelligent and privacy-focused document querying. It combines semantic retrieval with local language models to generate accurate, context-aware responses. The system reduces hallucinations by grounding outputs in retrieved data using vector databases like ChromaDB.

Local deployment via Ollama ensures data privacy and low-latency performance without cloud dependency. Overall, NexAI is a secure, cost-effective, and efficient solution for document-based question answering.

V. FUTURE SCOPE

NexAI currently focuses on text-based document querying using a local RAG framework with strong accuracy and privacy. Future enhancements include multimodal support (image, audio, video) using OCR and speech-to-text technologies.

Adopting advanced and fine-tuned language models can improve response quality and domain-specific accuracy. Scalability can

be improved through hybrid architectures combining local and cloud processing. Further optimizations and UI enhancements will makeNexAI more efficient, user-friendly, and adaptable for real-world applications.

VI. References

[1] P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Küttler, M. Lewis, W. Yih, T. Rocktäschel, S. Riedel, and D. Kiela, "Retrieval- Augmented Generation for Knowledge-Intensive NLP Tasks," Advances in Neural Information Processing Systems (NeurIPS), vol. 33, pp. 9459–9474, 2020.

[2] Johnson, M. Douze, and H. Jégou, "Billion-Scale Similarity Search with GPUs," IEEE Transactions on Big Data, vol. 7, no. 3, pp. 535–547, 2021.

[3] L. Gao, X. Ma, J. Lin, and others, "Retrieval- Augmented Generation for Large Language Models: A Survey," arXiv preprint arXiv:2312.10997, 2023.

[4] Goodfellow, Y. Bengio, and A. Courville, Deep Learning. Cambridge, MA, USA: MIT Press, 2016.

[5] T. Brown et al., "Language Models are Few-Shot Learners," Advances in Neural Information Processing Systems (NeurIPS), vol. 33, 2020.

[6] I.Vaswani et al., "Attention Is All You Need," Advances in Neural Information Processing Systems (NeurIPS), 2017.

[7] S. Bubeck et al., "Sparks of Artificial General Intelligence: Early Experiments with GPT-4," arXiv preprint arXiv:2303.12712, 2023.

[8] ChromaDB, "Chroma: The Open-Source Embedding Database," 2023. [Online]. Available: https://www.trychroma.com/

[9] Ollama, "Running Large Language Models Locally," 2024. [Online]. Available: https://ollama.com/

[10] H. Chen, J. Xu, and others, "Semantic Search Using Vector Embeddings," IEEE Access, vol. 10, pp. 12345–12358, 2022.

[11] S. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient Estimation of Word Representationsin Vector Space," arXiv preprint arXiv:1301.3781, 2013.

[12] T. Kenter and M. de Rijke, "Short Text Similarity with Word Embeddings," Proceedings of the ACM International Conference on Information and Knowledge Management, 2015.