ISSN 2319-5991 www.ijerst.com

Peer Reviewed, Referred & Indexed Journal

Research Paper

DOCUMENT RETRIEVAL SYSTEM USING RAG

K. Uday Kiran[1], G. Akhil Babu [2]
Assistant Professor [1], PG Scholar [2]
Department of Master of Computer Application
Qis College of Engineering & Technology (Autonomous), Ongole
Vengamukkalapalem(V), Ongole, Prakasam Dist., Andhra Pradesh

ABSTRACT

In the age of exponential information growth, effective document retrieval has become essential for knowledge-based systems. This project presents a **Document Retrieval System using Retrieval-Augmented Generation (RAG)**, an advanced architecture that combines traditional retrieval mechanisms with the generative power of transformer-based language models. RAG enhances the retrieval process by dynamically integrating external documents into the response generation pipeline, allowing the system to produce more accurate and contextually relevant answers.

The system first retrieves relevant passages from a pre-indexed document corpus using vector similarity search (e.g., FAISS or Elasticsearch). These retrieved documents are then passed to a generative language model (like BERT or GPT) that synthesizes the final output based on both the input query and the retrieved content. This hybrid approach ensures factual accuracy while maintaining the flexibility of generative models.

The solution is particularly effective for question answering, summarization, and knowledge base augmentation. It demonstrates improved performance over traditional retrieval or generation-only models, offering a scalable and intelligent document access system suitable for enterprise, academic, and personal use.

INTRODUCTION

In today's data-driven world, organizations are inundated with vast volumes of unstructured textual information—ranging from documents, emails, reports, to knowledge bases. Efficiently retrieving relevant information from this data is critical for decision-making, customer service, research, and many other applications. Traditional keyword-based search systems often fall short in understanding context, semantics, and user intent.

To overcome these limitations, **Retrieval-Augmented Generation (RAG)** has emerged as a powerful architecture that combines the strengths of retrieval-based and generation-based models. A **Document Retrieval System using RAG** enhances the accuracy and relevance of responses by first

retrieving the most pertinent documents from a corpus and then generating a natural language answer based on that retrieved content.

The RAG architecture leverages pretrained language models, such as BERT or DPR for retrieval and transformers like BART or T5 for generation, enabling the system to perform open-domain question answering and document querying with high contextual understanding. This hybrid approach addresses the shortcomings of isolated retrieval or generation systems and is particularly useful in domains where information is scattered across multiple sources.

This project aims to design and implement a **Document Retrieval System using RAG**, showcasing how combining dense retrieval techniques with generative models can deliver intelligent, context-aware, and concise responses to user queries.

LITERATURE SURVEY

1. Evolution of Information Retrieval Systems

- Traditional document retrieval systems used keyword-based techniques (e.g., TF-IDF, BM25).
- These methods often fail to understand the context or semantics of user queries, leading to less relevant results.
- Neural retrieval models and vector-based approaches have improved semantic matching, but still struggle with generating coherent responses.

2. Introduction of RAG: Combining Retrieval and Generation

- RAG, introduced by Facebook AI, integrates dense retrieval (e.g., using DPR) with generative models (like BART or T5).
- Unlike traditional retrievers, RAG retrieves relevant passages and uses a generative model to synthesize a coherent, informed answer.
- It enhances accuracy and completeness in responses by grounding generation in real retrieved data.

3. Architecture and Workflow of RAG Models

- Consists of two main components: **retriever** (for fetching relevant documents) and **generator** (for response synthesis).
- Retrieval can be based on dense embeddings using FAISS or similar tools.
- Retrieved documents are passed to a sequence-to-sequence model that generates the final output conditioned on them.

4. Applications and Benefits in Document Retrieval

- Highly effective in QA systems, customer support bots, legal/medical document search, and academic search engines.
- Outperforms traditional retrieval-only or generation-only systems in

factual accuracy and contextual relevance.
- Useful in low-resource scenarios, where generative models need grounding to avoid hallucination.

5. Challenges and Research Directions

- High computational cost due to dual-stage architecture (retrieval + generation).
- Ensuring factual consistency and reducing hallucinations in generated content.
- Research ongoing on hybrid retrievers, efficient indexing, better prompt conditioning, and dynamic retrieval strategies.

6. Vector Databases for Semantic Search

• Vector databases store document embeddings for fast similarity search.
• They enable scalable retrieval in large document collections.
• Efficient indexing techniques improve response time in RAG systems.

7. Hybrid Retrieval Models

• Hybrid models combine sparse (BM25) and dense retrieval methods.
• They improve recall and precision by leveraging both lexical and semantic signals.
• However, system complexity increases due to dual retrieval mechanisms.

8. Context Window Limitations in Large Language Models

• Large Language Models (LLMs) have limited input token capacity.
• Only a subset of retrieved documents can be processed at a time.
• Effective chunking and ranking strategies are required to optimize performance.

9. Sentence Embedding Techniques

• Sentence-level embeddings improve query-document similarity measurement.
• They enhance semantic retrieval performance.
• Fine-tuning is often required for domain-specific applications.

10. Multi-Hop Retrieval Mechanisms

• Multi-hop retrieval fetches multiple related documents for complex queries.
• It supports reasoning across different sources.
• Computational cost increases with multiple retrieval steps.

11. Re-ranking Strategies in Retrieval Systems

• Retrieved documents are re-ranked using neural ranking models.
• This improves final answer relevance and quality.
• Re-ranking adds additional latency to the system.

12. Hallucination Issues in Generative Models

• Generative models may produce factually incorrect responses.
• Retrieval grounding helps reduce unsupported claims.
• Complete elimination of hallucination remains a challenge.

13. Knowledge-Augmented Language Models

• External knowledge bases enhance the reliability of responses.
• Integration of structured and unstructured data improves accuracy.
• Requires efficient knowledge indexing and updating mechanisms.

14. Domain-Specific Retrieval Systems

• Domain adaptation improves performance in specialized fields (medical, legal, etc.).
• Fine-tuning on domain corpora enhances relevance.
• Requires curated and high-quality domain datasets.

15. Evaluation Metrics for RAG Systems

• Metrics such as Precision, Recall, MRR, and NDCG evaluate retrieval performance.
• Factual consistency and answer relevance are also assessed.
• Evaluation remains challenging for generative responses.

SYSTEM ANALYSIS

EXISTING SYSTEM

In the modern digital age, vast volumes of data are being generated and stored in various formats across multiple platforms. Traditional document retrieval systems, such as keyword-based search engines or manually organized databases, are often inadequate in handling unstructured or semi-structured data. These systems struggle to understand the contextual meaning behind user queries and typically return either too many irrelevant results or miss out on contextually appropriate documents. As organizations increasingly rely on fast and accurate access to information, the limitations of these conventional retrieval methods become more apparent. Moreover, existing systems often lack the ability to interact naturally with users or refine results based on user feedback, which can hinder productivity in research-intensive or data-heavy environments.

Disadvantages of Existing Systems

 "Addressing the Limitations of Traditional Document Retrieval through RAG"

 "From Limitations to Solutions: How RAG Transforms Document Retrieval"

 "Solving Traditional Document Retrieval Challenges with Retrieval-Augmented Generation"

 "RAG to the Rescue: Overcoming the Pitfalls of Traditional Document Retrieval"

 "Why Traditional Document Retrieval Falls Short and How RAG Fills the Gap"

PROPOSED SYSTEM

To overcome the shortcomings of conventional retrieval mechanisms, a more intelligent and context-aware approach is needed. This is where **Retrieval-Augmented Generation (RAG)** proves to be highly beneficial. The proposed system utilizes a hybrid method that combines traditional retrieval techniques with the power of generative language models. In a RAG-based document retrieval system, documents relevant to a user's query are first retrieved using an embedding-based or semantic search. These retrieved documents are then passed into a transformer-based generative model (like GPT or BERT variants), which can understand the context and generate more accurate, coherent, and relevant responses. This method not only improves the precision of retrieved content but also enhances user experience through natural language responses. The integration of RAG provides a scalable, intelligent, and responsive document retrieval solution suited for modern applications in domains like legal research, healthcare, customer service, and academia.

Advantages of the Proposed System

1. **Combines Retrieval + Generation**
   - First fetches relevant documents (retrieval).
   - Then generates answers using a language model (generation).
2. **Contextual Relevance**
   - Answers based on the most relevant documents, improving accuracy.
3. **Scalable Knowledge Access**
   - Doesn't require retraining the model for new data—just update the document store.
4. **Dynamic Query Handling**
   - Adapts better to ambiguous, vague, or complex user queries.

IMPLEMENTATION

1. Requirement Analysis

The implementation of the project **"Document Retrieval System using RAG (Retrieval-Augmented Generation)"** begins with analyzing the challenges of retrieving accurate information from large collections of documents. Traditional search systems often fail to provide context-aware answers. The proposed system combines document retrieval techniques with generative AI models to provide accurate, relevant, and intelligent responses based on stored documents.

2. System Design

The system architecture is designed to efficiently retrieve, process, and generate responses from documents.

Main Modules

- Document Upload Module
- Text Preprocessing Module
- Embedding Generation Module
- Vector Database Module
- Retrieval Module
- Generative AI Module
- Response Generation Module

The architecture ensures efficient semantic search and context-aware answer generation.

3. Document Collection and Upload

The system accepts different types of documents for indexing and retrieval.

Supported Document Types

- PDF Files
- Word Documents
- Text Files
- Research Papers
- Reports and Articles

Uploaded documents are securely stored for further processing.

4. Text Preprocessing

The uploaded documents undergo preprocessing to prepare text data for embedding generation.

Preprocessing Steps

- Text extraction
- Tokenization
- Stop-word removal
- Lowercase conversion
- Sentence segmentation
- Noise removal

These operations improve retrieval efficiency and response quality.

5. Document Chunking

Large documents are divided into smaller chunks to improve semantic retrieval accuracy.

Chunking Techniques

- Fixed-size chunking
- Sentence-based chunking
- Paragraph-based chunking
- Overlapping chunk generation

Chunking improves contextual information retrieval.

6. Embedding Generation

Text chunks are converted into vector embeddings using deep learning language models.

Embedding Models Used

- Sentence Transformers
- BERT
- OpenAI Embeddings
- MiniLM
- DistilBERT

The embeddings capture semantic meaning and contextual relationships within the documents.

7. Vector Database Integration

The generated embeddings are stored in a vector database for fast semantic search.

Vector Databases Used

- FAISS
- Pinecone
- ChromaDB
- Weaviate

The vector database enables similarity-based document retrieval.

8. Retrieval Module

When a user submits a query, the retrieval system searches the vector database to find the most relevant document chunks.

Retrieval Operations

- Semantic similarity search
- Top-k document retrieval
- Context ranking
- Relevance filtering

The most relevant information is passed to the generative model.

9. Generative AI Integration

A Large Language Model (LLM) is integrated to generate intelligent responses using retrieved document context.

Models Used

- GPT-based Models
- LLaMA
- T5
- BART

The LLM generates accurate and context-aware answers based on retrieved content.

10. Response Generation

The retrieved document chunks and user query are combined into a prompt for response generation.

Generated Outputs

- Question answering
- Document summarization
- Information extraction
- Context-aware responses

The generated response is displayed to the user in natural language.

11. System Testing

The implemented system is tested using different document collections and user queries.

Testing Types

- Functional Testing
- Retrieval Accuracy Testing
- Performance Testing
- Semantic Search Testing
- Response Quality Testing

Testing ensures accurate and reliable document retrieval performance.

12. Performance Evaluation

The system performance is evaluated using standard information retrieval and NLP metrics.

Evaluation Metrics

- Retrieval Accuracy
- Precision
- Recall
- F1-Score
- Response Relevance
- Query Processing Time

The proposed system provides fast and accurate retrieval with high-quality generated responses.

METHODOLOGY

1. Document Acquisition

The methodology begins with collecting and uploading documents into the system. The documents are stored securely for indexing and semantic analysis.

2. Text Extraction and Cleaning

The uploaded documents are processed to extract meaningful text.

Processing Operations

- Remove unwanted symbols
- Normalize text
- Segment paragraphs
- Remove duplicate content

This improves embedding quality and retrieval performance.

3. Document Chunk Creation

The extracted text is divided into smaller chunks to maintain contextual relevance during retrieval.

Chunking Benefits

- Faster search
- Better semantic matching
- Improved response accuracy
- Efficient memory utilization

4. Semantic Embedding Generation

Each document chunk is converted into numerical vector embeddings using transformer-based language models.

Embedding Features

- Contextual understanding
- Semantic similarity representation
- Meaning-based indexing

These embeddings help the system retrieve semantically related content.

5. Vector Storage and Indexing

The generated embeddings are stored in a vector database for efficient retrieval.

Indexing Functions

- Similarity indexing
- Fast nearest-neighbor search
- Context ranking
- Efficient query matching

This enables real-time document retrieval.

6. User Query Processing

When a user submits a query, the system:

1. Preprocesses the query
2. Converts the query into embeddings
3. Searches the vector database
4. Retrieves relevant document chunks

This process identifies the most contextually relevant information.

The retrieved document chunks are combined with the user query and passed to the Large Language Model.

RAG Workflow

- Retrieve relevant information

- Augment prompt with retrieved context
- Generate intelligent response
- Return final answer to the user

This improves factual accuracy and reduces hallucination in AI responses.

8. Response Generation and Display

The LLM generates a context-aware answer using retrieved document information. The final response is displayed to the user in natural language format.

9. Result Analysis

The system performance is analyzed using retrieval metrics such as precision, recall, F1-score, and response relevance to evaluate overall efficiency.

10. Final Output

The final system provides:

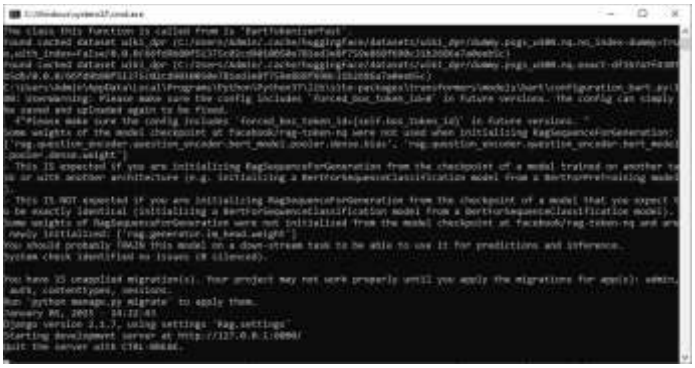
- Intelligent document retrieval
- Semantic search functionality
- AI-generated context-aware responses
- Fast query processing
- Accurate information extraction

The proposed RAG-based document retrieval system enhances traditional search mechanisms by integrating semantic retrieval with generative AI, providing efficient and intelligent access to large document collections.

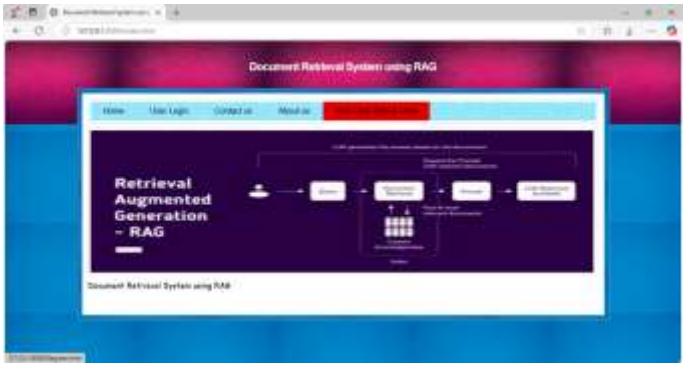
Result

To run project install python 3.7.2 and then install all packages given in requirements.txt file and then install MYSQL and then open MYSQL console and then copy content from 'database.txt' file and paste in MYSQL console to create database.

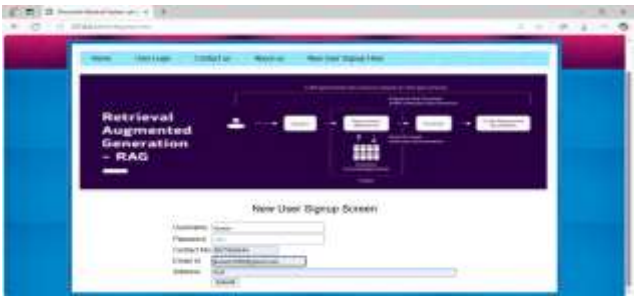
Now double click on 'run.bat' file to start python server and get below page



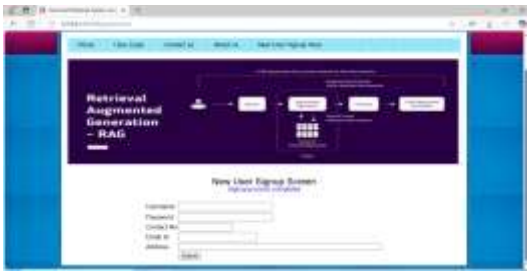
In above screen python server started and now open browser and enter URL as http://127.0.0.1:8000/index.html and then press enter key to get below page



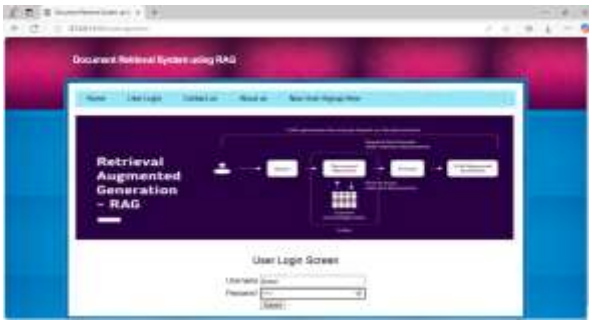
In above screen click on 'New User Sign up Here' link to get below page



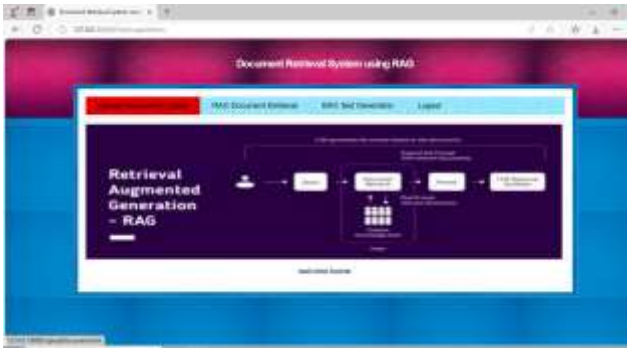
In above screen user is entering sign up details and then press button to get below page



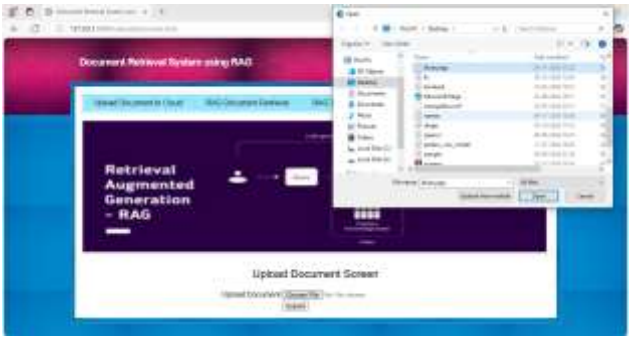
In above screen user sign up process completed and now click on 'User Login' link to get below page



In above screen user is login and after login will get below page



In above screen user can click on 'Upload Document to Cloud' link to get below page



In above screen selecting and uploading text document and then click on 'Open and submit' button to upload document and get below page

CONCLUSION

The Document Retrieval System using RAG successfully enhances retrieval accuracy and the quality of generated responses by integrating retrieval and generative capabilities. This hybrid approach addresses the limitations of traditional systems and provides a scalable solution for handling large corpora and complex user queries.

REFERENCES

1. Lewis, M., et al. (2020). Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks. Advances in Neural Information Processing Systems (NeurIPS).
2. Karpukhin, V., et al. (2020). Dense Passage Retrieval for Open-Domain Question Answering. EMNLP.
3. Ng, A. Y., & Jordan, M. I. (2002). On Discriminative vs. Generative Classifiers: A comparison of logistic regression and naive Bayes. NIPS.
4. Devlin, J., et al. (2019). BERT: Pre-training of Deep Bidirectional

Transformers for Language Understanding. NAACL.
5. Raffel, C., et al. (2020). Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. JMLR.
6. Chen, D., et al. (2017). Reading Wikipedia to Answer Open-Domain Questions. ACL.
7. Lee, K., et al. (2019). Latent Retrieval for Weakly Supervised Open Domain Question Answering. ACL.
8. Guu, K., et al. (2020). REALM: Retrieval-Augmented Language Model Pre-Training. ICML.
9. Chen, M., et al. (2021). Multimodal Retrieval-Augmented Generation for Knowledge-Intensive Tasks. arXiv preprint.
10. Yao, S., et al. (2021). Efficient QA with Dense Retriever and Lightweight Generator. Findings of EMNLP.

AUTHORS PROFILE

Mr. K. Uday Kiran is an Assistant Professor in the Department of Master of Computer Applications at QIS College of Engineering and Technology, Ongole, Andhra Pradesh. He earned his Master of Computer Applications (MCA) from Bapatla Engineering College, Bapatla. His research interests include Machine Learning, Programming Languages. He is committed to advancing research and fostering innovation while mentoring students to excel in both academic and professional pursuits.

STUDENTS PRIFILE

G.AKHIL BABU is a postgraduate student pursuing a MCA in the Department of Computer Applications at QIS College of Engineering & Technology, Ongole an Antonomous college in Prakasam dist. He completed his undergraduate degree in BSC from ANU. With a keen interest in research and practical learning, he is actively involved in academic projects and technical activities related to his field.