EchoDoc: Make Your Documents Speak

*

1st Dr. Priyanka Meel
Department of Information Technology
Delhi Technological University
Delhi, India
priyankameel@dtu.ac.in

2nd Sujal
Department of Information Technology
Delhi Technological University
Delhi, India
Sujalgupta09@gmail.com

3rd Vivek Patwal
Department of Information Technology
Delhi Technological University
Delhi, India
vivekpatwal543210@gmail.com

4th Sushant
Department of Information Technology
Delhi Technological University
Delhi, India
sushantnair31@gmail.com

***Abstract*—Using Streamlit, FastAPI, and React, this project presents an interactive, document-based chatbot driven by Retrieval-Augmented Generation (RAG). The system incorporates OpenAI's language models for intelligent question answering, Pinecone for vector storage, and Cohere for text embeddings. In order to facilitate semantic search and AI-driven responses, users can upload PDF documents that have been parsed, segmented, and indexed.**

The chatbot shows how to effectively use vector embeddings and language models to increase the accuracy of question answering over unstructured documents. Retrieval restrictions resulting from fixed chunking, inconsistent text extraction, and sporadic model hallucinations are among the main issues noted. Adaptive chunking, hybrid retrieval techniques, and OCR-scanned PDFs are the goals of future improvements. By enhancing accessibility and interaction with intricate information sources, this work advances document-focused conversational AI.

***Index Terms*—component, formatting, style, styling, insert**

I. Introduction

The need for intelligent systems that can draw pertinent conclusions from vast and intricate document repositories has increased due to the quick growth of digital documentation. The effectiveness of traditional keyword-based search techniques is often limited, especially when it comes to technical, legal, or academic PDFs, as they frequently miss semantic relationships and contextual nuances.

More sophisticated document interaction through natural language interfaces has been made possible by recent advancements in large language models (LLMs) and vector-based retrieval. By fusing semantic search with LLMs' generative powers, Retrieval-Augmented Generation (RAG) provides a potent remedy that enables users to ask natural language questions and get logical, context-aware answers. However, maintaining context in lengthy documents, reducing pointless retrievals, guaranteeing low-latency performance, and managing multi-document queries are all obstacles to efficient PDF interaction. By combining sophisticated parsing, chunking, vector storage, and LLM-powered response generation into a unified, RAG-based system, this paper presents a solution to these problems.

II. Related Work

A. Maintaining the Integrity of the Specifications

Traditional keyword-based approaches like TF-IDF and BM25 have given way to more semantically aware methods in document retrieval. Early models lacked contextual understanding, but they worked well for exact matches. Bidirectional context was introduced with the introduction of transformer-based models such as BERT, which greatly enhanced semantic comprehension.

Answer generation and retrieval accuracy have been further enhanced by recent large language models (LLMs), such as the GPT series and embedding-specialized variants. Vector databases like Pinecone, Weaviate, and FAISS are essential to contemporary document interaction systems because they complement these developments by enabling scalable, real-time similarity search over high-dimensional embeddings.

III. System Architecture and Methodology

A. Overall System Design

A modular three-stage pipeline that includes document preprocessing and indexing, semantic retrieval, and response generation is used in the suggested Retrieval-Augmented Generation (RAG) system. The architecture makes use of Streamlit to enable quick prototyping and demonstration, Flask for backend processing, and React for the frontend interface.

B. Document Ingestion and Preprocessing

To reliably extract text from machine-readable PDF files, the document ingestion pipeline makes use of LangChain's PDFPlumberLoader. Documents are divided into segments of roughly 1,000 characters with 200-character overlaps using a

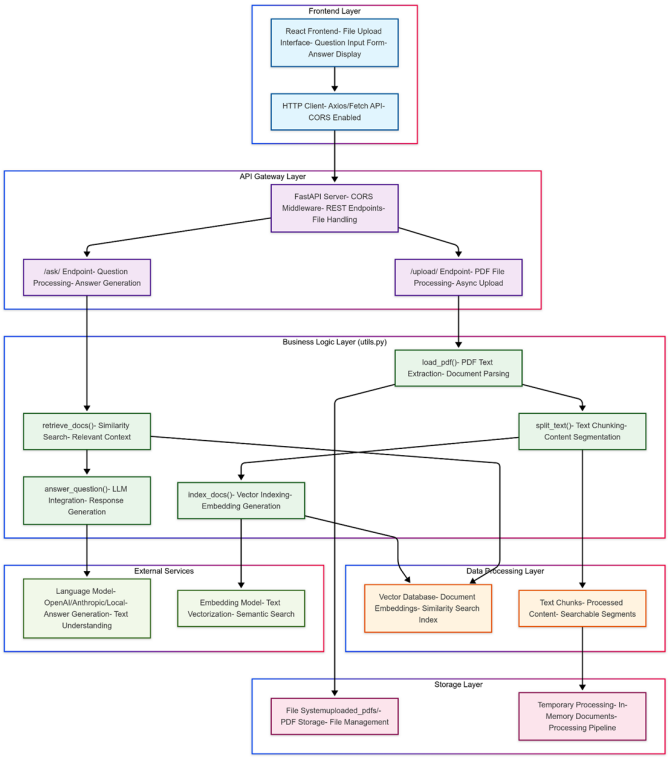

Fig. 1. Flowchart

recursive character-based splitting technique in order to maintain contextual coherence. By ensuring information continuity across chunk boundaries, this overlapping segmentation helps to mitigate context loss during retrieval.

Cohere's llama-text-embed-v2 model, which captures semantic representations essential for similarity matching, is used to encode each text chunk into a high-dimensional vector. Pinecone's vector database indexes these embeddings along with metadata like page numbers and document identifiers to facilitate effective similarity-based search.

C. Semantic Retrieval Mechanism

Using Cohere's embedding model, semantic retrieval is accomplished by projecting user queries into the same embedding space as the document chunks. Using cosine similarity metrics, the system chooses the five most pertinent chunks as part of a top-k retrieval strategy. This ensures that the system surfaces contextually relevant content instead of depending solely on surface-level keyword matching by striking a balance between retrieval accuracy and computational efficiency.

D. Response Generation Framework

The system uses Cohere's command-r-plus generative model for answer synthesis, producing accurate and contextually coherent responses based on the retrieved document segments. In order to enable targeted and organic conversational interactions, retrieved chunks are integrated into engineered prompt templates that blend user queries with pertinent context. This prompt design technique keeps the conversation flowing and improves the relevance of the responses.

IV. Experimental Results and Analysis

A. System Performance

Document Processing: Using LangChain's PDFPlumberLoader, the document pipeline quickly extracted clean text from structured PDFs. Contextual continuity was maintained with little processing overhead thanks to recursive chunking with 200-character overlaps.

Embedding and Indexing: Cohere's llama-text-embed-v2 was used to quickly generate embeddings, which provided condensed but semantically rich vectors. Pinecone's optimized vector index made it possible to store and retrieve large datasets with ease and low latency.

Retrieval Accuracy: The most contextually relevant chunks were consistently retrieved by top-5 cosine similarity. Particularly for indirect or complex queries, the semantic search consistently outperformed conventional keyword methods in terms of precision.

Response Generation: Clear, precise responses were provided by the command-r-plus model. In the technical, legal, and academic domains, responses handled both direct and cross-paragraph queries quickly and clearly while staying true to the original content.

B. Comparative Analysis

The suggested system was compared to well-known substitutes such as ChatPDF and conventional keyword search engines. The comparative results are summarized in Table I.

TABLE I
Performance Comparison of EchoDoc with Other Systems

Metric	EchoDoc (Proposed)	ChatPDF	Traditional Search
Semantic Understanding	High	Medium	Low
Response Coherence	High	Medium	Low
Processing Speed	Fast	Fast	Very Fast
Context Preservation	High	Medium	Low

According to these findings, EchoDoc provides a better user experience by handling lengthy or complicated documents more effectively and providing more accurate, insightful responses.

C. Identified Limitations

Despite its advantages, a number of drawbacks were noted:

- **OCR Support:** Without OCR integration, the system is unable to process scanned or image-based PDFs.
- **Chunking Problems:** Sometimes, context-sensitive data is broken up by fixed-length chunking.
- **Hallucinations:** In rare cases, response hallucinations occurred when the context was unclear or insufficient.
- **Multi-turn Context Drift:** The model occasionally lost context during lengthy discussions, which had an impact on coherence.

V. Challenges and Future Enhancements

A. Current System Challenges

- **Scalability:** It takes a lot of resources to process and store embeddings for big document collections.
- **Retrieval Noise:** Retrieval noise is introduced when similar but unrelated chunks sporadically surpass retrieval thresholds.
- **Latency:** Large prompt sizes and the computational expense of embedding generation affect real-time responsiveness.
- **Cross-Document Queries:** Without sophisticated context merging techniques, it is still challenging to synthesize responses from several documents.

B. Proposed Enhancement Strategies

- **Hybrid Retrieval:** To increase recall and precision, combine semantic search with conventional keyword techniques (BM25 + vector search, for example).
- **Clickable Highlights:** Give users the ability to pinpoint specific document sections from responses.
- **Source Tracking:** To improve transparency, automatically add citations with page numbers.
- **Domain Adaptation:** Adjust prompts and embeddings for documents that are scientific, legal, or medical.
- **Modular Support:** Add additional models (OpenAI GPT, FAISS, and Weaviate) to the backend compatibility list.

C. Technical Improvements

- **OCR Integration:** Use programs like Tesseract or PaddleOCR to enable scanned or image-based PDFs.
- **Intelligent Chunking:** To maintain logical document structure, use topic- or sentence-based splitting.
- **Context Summarization:** To improve efficiency and token management, condense the retrieved chunks.
- **User Feedback Loops:** To continuously enhance response quality, implement interactive feedback systems and evaluation metrics (BLEU, ROUGE).

VI. Conclusion

A RAG-based system for interactive PDF processing that combines large language models, vector retrieval, and semantic embeddings was presented in this paper. The system manages contextual understanding, document ingestion, and response generation with great accuracy. Its robustness for structured documents and semantic relevance are demonstrated by experimental results.

Notwithstanding drawbacks like context fragmentation, sporadic hallucinations, and a lack of OCR, suggested improvements like hybrid retrieval and OCR integration provide obvious avenues for advancement. The framework's modular design and assessment offer a solid foundation for further advancement and implementation across a range of fields, meeting the increasing demand for effective document-based knowledge extraction.

Acknowledgment

The LangChain, Cohere, Pinecone, and OpenAI platforms, among other development frameworks and APIs used in this study, facilitated the deployment and assessment of the suggested system, for which the authors are grateful.

References

1) A. Vaswani *et al.*, "Attention is All You Need," in *Advances in Neural Information Processing Systems*, 2017.
2) S. Li, L. Stenzel, C. Eickhoff, and S. A. Bahrainian, "Enhancing Retrieval-Augmented Generation: A Study of Best Practices," *arXiv preprint arXiv:2501.07391*, Jan. 2025. [Online]. Available: https://doi.org/10.48550/arXiv.2501.07391
3) X. Li, Y. Cao, Y. Ma, and A. Sun, "Long Context vs. RAG for LLMs: An Evaluation and Revisits," *arXiv preprint arXiv:2501.01880*, Jan. 2025. [Online]. Available: https://doi.org/10.48550/arXiv.2501.01880
4) Y. Gao *et al.*, "Retrieval-Augmented Generation for Large Language Models: A Survey," *arXiv preprint arXiv:2312.10997*, Dec. 2023. [Online]. Available: https://doi.org/10.48550/arXiv.2312.10997
5) S. Gautam, "Bridging Multimedia Modalities: Enhanced Multimodal AI Understanding and Intelligent Agents," in *Proc. 25th Int. Conf. Multimodal Interaction (ICMI)*, ACM, 2023, pp. 695–699. [Online]. Available: https://doi.org/10.1145/3577190.3614225
6) S. Krishna *et al.*, "Fact, Fetch, and Reason: A Unified Evaluation of Retrieval-Augmented Generation," *arXiv preprint arXiv:2409.12941*, Jan. 2025. [Online]. Available: https://doi.org/10.48550/arXiv.2409.12941
7) Y. Liu *et al.*, "RA-ISF: Learning to Answer and Understand from Retrieval Augmentation via Iterative Self-Feedback," *arXiv preprint arXiv:2403.06840*, Jun. 2024. [Online]. Available: https://doi.org/10.48550/arXiv.2403.06840
8) A. Balaguer *et al.*, "RAG vs Fine-tuning: Pipelines, Tradeoffs, and a Case Study on Agriculture," *arXiv preprint arXiv:2401.08406*, Jan. 2024. [Online]. Available: https://doi.org/10.48550/arXiv.2401.08406
9) J. Chen, H. Lin, X. Han, and L. Sun, "Benchmarking Large Language Models in Retrieval-Augmented Generation," *arXiv preprint arXiv:2309.01431*, Dec. 2023. [Online]. Available: https://doi.org/10.48550/arXiv.2309.01431
10) M. Kang *et al.*, "Knowledge Graph-Augmented Language Models for Knowledge-Grounded Dialogue Generation," *arXiv preprint arXiv:2305.18846*, 2023. [Online]. Available: https://doi.org/10.48550/arXiv.2305.18846
11) S. Nakano *et al.*, "WebGPT: Browser-assisted question-answering with human feedback," OpenAI, 2021. [Online]. Available: https://openai.com/research/webgpt

12) S. Izacard and E. Grave, “Leveraging Passage Retrieval with Generative Models for Open Domain Question Answering,” *arXiv preprint arXiv:2007.01282*, Jul. 2020.
13) J. Karpukhin *et al.*, “Dense Passage Retrieval for Open-Domain Question Answering,” in *Proc. Empirical Methods in Natural Language Processing (EMNLP)*, 2020, pp. 6769–6781.
14) A. Lewis *et al.*, “PaLM: Scaling Language Modeling with Pathways,” *arXiv preprint arXiv:2204.02311*, Apr. 2022.
15) S. Borgeaud *et al.*, “Improving Language Models by Retrieving from Trillions of Tokens,” in *Proc. Int. Conf. Machine Learning (ICML)*, 2022.
16) T. Yu *et al.*, “Generate Rather than Retrieve: Large Language Models are Strong Context Generators,” *arXiv preprint arXiv:2306.11644*, Jun. 2023.
17) A. Mialon *et al.*, “Augmented Language Models: A Survey,” *arXiv preprint arXiv:2302.07842*, Feb. 2023.
18) C. Shi *et al.*, “Long-Context Retrieval-Augmented Generation for Complex Question Answering,” *arXiv preprint arXiv:2305.06983*, May 2023.
19) L. Pang *et al.*, “Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks,” *arXiv preprint arXiv:2005.11401*, May 2020.
20) R. Nogueira and K. Cho, “Passage Re-ranking with BERT,” *arXiv preprint arXiv:1901.04085*, Jan. 2019.
21) S. Wang *et al.*, “Augmenting Large Language Models with Long-Term Memory,” *arXiv preprint arXiv:2305.08355*, May 2023.
22) J. Lin *et al.*, “A Survey on Retrieval-Augmented Language Models,” *arXiv preprint arXiv:2306.13406*, Jun. 2023.
23) R. Menon *et al.*, “Do Retriever-Augmented Language Models Need Reasoning?” *arXiv preprint arXiv:2305.12191*, May 2023.
24) B. Yang *et al.*, “Improving LLMs’ Faithfulness with Retrieval-Augmented Generation,” *arXiv preprint arXiv:2310.02268*, Oct. 2023.
25) H. Sun *et al.*, “Memory-Augmented Large Language Models: A Survey,” *arXiv preprint arXiv:2402.08341*, Feb. 2024.