International Journal of Scientific Research and Engineering Development-– Volume 7 Issue 6, Nov- Dec 2024
Available at www.ijsred.com

RESEARCH ARTICLE OPEN ACCESS

PDFgpt: Chat with your PDFs Using Generative AI

Dr. Archana Kumar*, Nandini Singh**, Ansh Varshney***
*(Dept. of AI&DS, ADGIPS, New Delhi, India
Email: profdrarchanakumar@gmail.com)
**(Dept. of AI&DS, ADGIPS, New Delhi, India
Email: nandiniisingh.work@gmail.com)
***(Dept. of AI&DS, ADGIPS, New Delhi, India
Email: anshvarshney1109@gmail.com)

----------**********----------

Abstract:

This project introduces a PDF-based Question Answering (QnA) system leveraging Large Language Models (LLMs) to enable efficient and accurate information retrieval from diverse document formats. Using PyMuPDF, PyPDF2, and OCR through PyTesseract, the system extracts and processes both text-based and image-based PDF content. It then utilizes Sentence Transformers for embeddings, stored in Chroma's vector database for high-performance similarity search. This architecture supports LLaMA3-8b-8192 for context-driven responses delivered via text and audio through gTTS. Future enhancements include multilingual support, multi-PDF querying, and real-time indexing, providing a versatile solution for complex document interaction across domains.

Keywords — **Retrieval-Augmented Generation, Large Language Models, Sentence Transformers, Chunking, Embeddings, Chroma Vector Database**

----------**********----------

I. INTRODUCTION

Document-based knowledge retrieval systems have evolved rapidly with the rise of Retrieval-Augmented Generation (RAG) and Large Language Models (LLMs), providing a level of depth and accuracy previously unavailable in traditional keyword-driven retrieval methods. The RAG architecture combines generative capabilities of Large Language Models (LLMs) with the precision of information retrieval. This approach has the potential to redefine how we interact with and augment both structured and unstructured knowledge in generative models to enhance transparency, accuracy, and contextuality of responses[1]. However, many of today's RAG and LLM-based applications are locked behind high usage costs, making them inaccessible to broader audiences, particularly in educational and non-commercial contexts.

Our system differentiates itself by offering a 100% free, open-source solution for advanced document-based question answering. By leveraging Sentence Transformers for semantic embeddings and the Chroma Vector Database for efficient, accurate information retrieval, our application is capable of processing both text-based and image-based PDFs. It also supports complex extraction with tools like PyMuPDF and PyPDF2, enabling robust text extraction even from scanned and non-textual PDF formats.

In contrast to other systems that impose subscription fees, usage limits, or premium tiers, our application is designed for accessibility, removing cost barriers that hinder many potential users. This democratization allows researchers, students, and professionals alike to engage with a powerful, contextually aware tool for document-based querying and knowledge extraction, making it a valuable resource across diverse fields.

1.1 Applications

The Retrieval-Augmented Generation (RAG) architecture coupled with document-based Large Language Model (LLM) systems like ours has a broad range of applications across various domains due to its ability to extract and generate contextually accurate information from both structured and unstructured data sources. Key applications include:

a. Academic Research and Education: Researchers and students can use the system to quickly extract relevant information from extensive academic papers, textbooks, and research articles. By asking specific questions, users gain insights without manually combing through large volumes of text,

making it especially useful for literature reviews and exam preparation.

b. Legal and Compliance: Legal professionals can leverage the system to retrieve precise information from lengthy legal documents, case files, and regulatory texts. This application helps in locating relevant case law, statutes, or compliance requirements, enhancing efficiency in legal research and analysis.
c. Corporate Knowledge Management: Companies can use this application to query organizational knowledge bases, employee handbooks, training materials, and policy documents. This enables employees to access specific information quickly, improving productivity and fostering better knowledge transfer within the organization.
d. Healthcare and Medical Research: Medical professionals and researchers can utilize the system to retrieve critical information from patient records, medical research papers, and pharmaceutical documentation. This application aids in improving patient outcomes by providing access to research-backed information and supporting evidence-based practices.
e. Finance and Investment: Financial analysts and investors can retrieve key data from reports, financial statements, and industry analyses to support investment decisions. By providing targeted, contextually accurate information, the application can significantly streamline the research process for financial professionals.
f. Customer Support and Service: Customer service teams can use the system to retrieve answers from knowledge bases, user manuals, and troubleshooting guides, thereby enabling more efficient and accurate responses to customer inquiries.
g. Multilingual Document Processing: With potential future integration of multilingual support, the application will expand to cater to non-English documents, allowing users to query texts across languages, making it useful for global enterprises and researchers working with diverse language sources.

These applications underscore the versatility and wide-reaching impact of a robust RAG-based document retrieval system, particularly one that is freely accessible.

1.2 Role of different fields

The development of a Retrieval-Augmented Generation (RAG)-based system is driven by contributions from multiple fields, each providing critical components to its functionality. Technology and software development play a foundational role by providing the necessary frameworks and libraries to handle documents, process data, and create interactive interfaces. Tools like PyMuPDF and PyPDF2 enable efficient extraction of text from PDFs, while platforms like Streamlit help in developing the user interface. These efforts ensure the system is efficient, scalable, and user-friendly.

Natural Language Processing (NLP), Machine Learning, and AI are integral to the system's ability to understand and generate contextually relevant responses. NLP techniques, including the use of transformer models like Sentence Transformers and deep learning models like LLaMA, are employed to process user queries and extract meaning from large text datasets. Information retrieval techniques and vector databases like Chroma store and retrieve data, improving the precision and relevance of the system's responses. These fields enable the RAG architecture to efficiently combine generative capabilities with retrieval precision.

User Experience (UX) Design, Cloud Computing, and Data Science ensure the system is not only efficient but also accessible and user-friendly. Cloud infrastructure handles the computational load required for processing large datasets, while UX design focuses on creating an intuitive interface. Additionally, academic research contributes to refining algorithms, and business partnerships help apply the technology across various industries, making the system versatile for real-world applications. Together, these diverse fields create a robust, scalable solution for document processing and querying large datasets.

1.3 Challenges in RAG for PDF Processing

Despite the significant potential of Retrieval-Augmented Generation (RAG) models, several challenges need to be addressed, particularly in processing PDFs. One key issue is the retrieval mechanism, which can struggle with ambiguous queries or specialized domains, often retrieving irrelevant or off-topic documents. Improving query expansion and contextual disambiguation techniques is necessary to enhance retrieval accuracy.

The integration between the retrieval and generation components also presents challenges. While the processes are well-aligned in theory, the generative model may not always effectively incorporate retrieved information, leading to inconsistencies. Research into better alignment techniques, such as advanced attention mechanisms, is needed to address this issue.Additionally, RAG models are computationally expensive, requiring both retrieval and generation steps for each query. This can be resource-intensive, especially with large datasets or real-time applications. Techniques like model pruning or knowledge distillation could help reduce the computational burden without compromising performance.

Finally, ethical concerns like bias and transparency must be addressed. While retrieval can mitigate biases by sourcing diverse documents, there is still a risk of amplifying existing biases in the retrieved content. Ensuring transparency in the selection and integration of documents is critical to maintaining trust, especially in sensitive applications like legal or medical domains.

II. LITERATURE REVIEW

Recent developments in Retrieval-Augmented Generation (RAG) have significantly advanced the processing capabilities for tasks involving PDF documents. Prior to RAG, NLP models typically focused on either retrieval or generation. While retrieval-based systems were efficient in fetching relevant documents, they lacked the ability to synthesize or create new information. Conversely, generative models, known for their fluency and creativity, often struggled with factual accuracy. The fusion of retrieval and generation in hybrid systems has revolutionized document processing,

enabling models to generate more accurate and contextually relevant responses.

One notable breakthrough is the agentic RAG systems, such as the one proposed by Ravuru [2] which employ a multi-agent hierarchical architecture. These systems use specialized sub-agents fine-tuned for specific tasks, showcasing improved flexibility and performance, especially in time-series analysis. Similarly, the RULE framework [3] addresses the challenge of factual accuracy in medical Vision-Language Models by introducing strategies to calibrate selection and optimize preferences, thereby improving the factuality of RAG systems in the healthcare domain. The METRAG framework [4] enhances RAG by incorporating multi-layered, thought-enhanced generation techniques, which refine the outputs based on task utility and document similarity, demonstrating improved performance in knowledge-intensive tasks.

Additional advancements, such as RAFT [5] and FILCO [6], focus on enhancing the retrieval process by eliminating irrelevant documents and minimizing over-reliance on retrieved passages. These approaches help to improve the quality and relevance of the information provided to the generative model, reducing errors like hallucinations and improving the overall reliability of RAG systems. Furthermore, Self-RAG [7] introduces reflection tokens for adaptive response generation, while CommunityKG-RAG [8] incorporates community structures within Knowledge Graphs to improve fact-checking and contextual relevance.

Finally, the RAPTOR model [9] introduces a hierarchical retrieval process, summarizing information at different abstraction levels to improve performance in complex reasoning tasks, such as question answering. Studies comparing RAG with long-context (LC) LLMs[10], show that while LC models might outperform RAG in certain contexts, RAG remains more cost-efficient, making it an optimal choice for real-time document processing tasks, especially in applications like PDF-based querying.

III. METHODOLOGY

The methodology focuses on the efficient extraction, processing, and retrieval of information from PDF documents. The system ensures seamless integration of document understanding, query handling, and response generation, providing a user-friendly interface and leveraging state-of-the-art AI technologies for enhanced performance and accuracy.

3.1 General Design

The PDFgpt application is designed to facilitate question-answering and content extraction from PDF documents. Built using a Retrieval-Augmented Generation (RAG) model, the application leverages the LLaMA3-8b-8192 model, Langchain, Hugging Face, Groq, and Chroma for efficient text processing, similarity search, and multimodal responses. The system integrates various components such as PDF text extraction, embedding creation, vector databases for fast querying, and a generative model for generating context-aware responses. It provides users with an interactive interface built using Streamlit, enabling seamless document upload and text-to-speech functionalities.

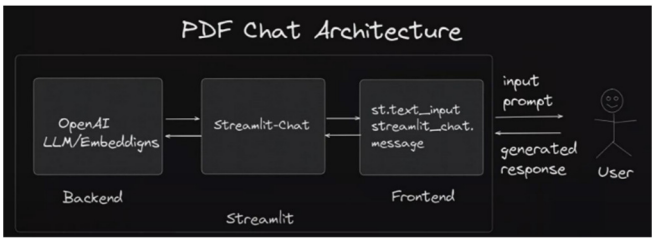

Fig. 1 PDFgpt – System Architecture

3.2 Prerequisites

Before using the PDFgpt application, the following prerequisites are required:

1. Software and Libraries: Python 3.8+, Streamlit, Langchain, Hugging Face Transformers, Groq, Chroma, PyMuPDF, PyPDF2, gTTS (Google Text-to-Speech).
2. Hardware: A system with sufficient RAM (minimum 8GB) and processing capability. For faster execution, an environment with GPU support is recommended.
3. File Requirements: Users should upload PDF documents in .pdf format. The system can process PDFs up to 200MB in size and can handle various orientations (even non-upright PDFs).
4. Internet Connection: Required for fetching the LLaMA model and other dependencies via external libraries and APIs.

3.3 Flow of the Application

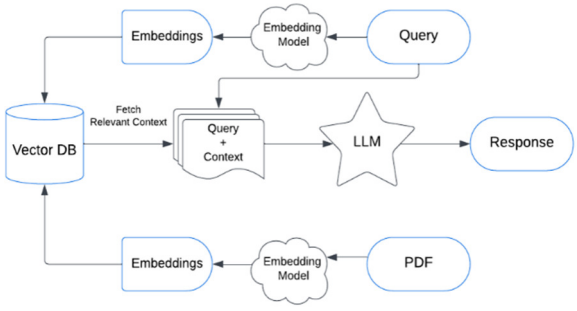

Fig.2 Flowchart of the designed system

i. Document Upload: The user uploads a PDF document through the Streamlit interface.
ii. Text Extraction: The uploaded PDF is processed using PyMuPDF or PyPDF2 for text extraction, which converts the document into machine-readable text.
iii. Text Chunking & Embedding: The extracted text is split into smaller chunks for efficient processing. These chunks are then embedded using transformer models (e.g., Sentence Transformers) and stored in a vector database (Chroma) for fast retrieval.
iv. Query Handling: Users can enter questions or queries related to the document. The system

retrieves the most relevant text chunks from the vector database using similarity search.

v. Response Generation: The generative model (LLaMA3-8b-8192) processes the retrieved information and generates context-aware responses.

vi. Text-to-Speech (Optional): If enabled, the generated response is converted to speech using Google Text-to-Speech (gTTS), providing users with an audio response.

vii. Result Display: The application returns the answer in both text and audio (if enabled), displaying the response to the user in the interface.

3.4 Testing

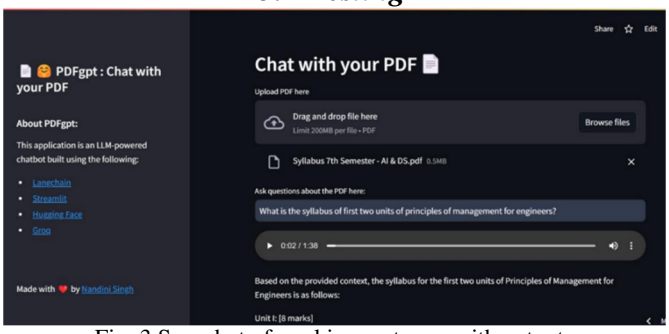

Fig. 3 Snapshot of working system – with output.

Throughout the development of PDFgpt, we implemented a variety of testing techniques to ensure the system functioned as intended and met user expectations:

- Unit Testing: Key functions, such as text extraction and embedding generation, were unit tested to ensure that each component performed correctly in isolation, handling edge cases and producing the expected results.
- Integration Testing: Following unit testing, integration testing was performed to ensure that the system components—such as PDF processing, embedding creation, and query handling—worked seamlessly together.
- User Acceptance Testing (UAT): To confirm the system met user needs, UAT was conducted by allowing users to upload different PDFs and ask questions. This helped identify usability issues and refine the interface.

These comprehensive testing methods ensured PDFgpt's reliability, robustness, and readiness for deployment.

IV. Conclusion

In conclusion, PDFgpt successfully integrates advanced natural language processing techniques with efficient PDF handling, offering users an intelligent, context-aware platform for querying custom PDFs. By leveraging state-of-the-art models such as LLaMA3 and Chroma, PDFgpt delivers high-quality, accurate, and contextually relevant responses, improving document interaction efficiency. The robust testing strategy ensured that the system functions seamlessly, meeting user expectations and delivering a reliable, user-friendly experience. PDFgpt proves to be a valuable tool for various use cases where document querying and information retrieval are essential.

V. Future Scope

The current version of PDFgpt integrates text extraction, machine learning, and NLP, but there are several opportunities for future enhancements:

- Multilingual Support: Adding support for multiple languages, including non-Latin scripts, through multilingual OCR and models like mBERT would expand global applicability.
- Multiple PDF Merging and Querying: Allowing users to merge multiple PDFs into a single document for unified querying would improve its use in research, legal, and business contexts.
- Real-time Collaborative QnA and Feedback: Real-time collaboration features would enable multiple users to interact with the system simultaneously, improving teamwork in research or corporate environments.
- Advanced Search Filters and Customization: Offering more granular search options like date range and document category filtering would enhance result precision for specialized users.
- Support for Richer Document Types and Formats: Supporting additional formats like Word, Excel, PowerPoint, and scanned images would increase the system's versatility.
- AI-Driven Document Summarization: Implementing an AI-driven summarization module would allow users to quickly access concise summaries of large documents, improving efficiency.

VI. References

1. Ayman Asad Khan, Md Toufique Hasan, Kai Kristian Kemell, Jussi Rasku, Pekka Abrahamsson, "Developing Retrieval Augmented Generation (RAG) based LLM Systems from PDFs: An Experience Report," Oct 2024, https://arxiv.org/abs/2410.15944
2. Ravuru, C., Sakhinana, S. S., & Runkana, V. (2024). Agentic Retrieval-Augmented Generation for Time Series Analysis. ArXiv. https://arxiv.org/abs/2408.14484
3. Peng Xia, Kangyu Zhu, Haoran Li, Hongtu Zhu, Yun Li, Gang Li, Linjun Zhang, Huaxiu Yao, "RULE: Reliable Multimodal RAG for Factuality in Medical Vision Language Models," Oct 2024, https://arxiv.org/abs/2407.05131
4. Gan, C., Yang, D., Hu, B., Zhang, H., Li, S., Liu, Z., Shen, Y., Ju, L., Zhang, Z., Gu, J., Liang, L., & Zhou, J. (2024). Similarity is Not All You Need: Endowing Retrieval Augmented Generation with Multi-Layered Thoughts. ArXiv. https://arxiv.org/abs/2405.19893
5. Zhang, T., Patil, S. G., Jain, N., Shen, S., Zaharia, M., Stoica, I., & Gonzalez, J. E. (2024). RAFT: Adapting Language Model to Domain Specific RAG. ArXiv. https://arxiv.org/abs/2403.10131
6. Zhiruo Wang, Jun Araki, Zhengbao Jiang, Md Rizwan Parvez, Graham Neubig, "Learning to Filter Context for Retrieval-Augmented Generation," Nov 2023, arXiv:2311.08377v1
7. Akari Asai, Zeqiu Wu, Yizhong Wang, Avirup Sil, Hannaneh Hajishirzi, Self-RAG: Learning to Retrieve,

Generate, and Critique through Self-Reflection, Oct 2023, arXiv:2310.11511
8. Rong-Ching Chang, Jiawei Zhang, "CommunityKG-RAG: Integrating Knowledge Graph Community Structures in RAG," Aug 2024, https://arxiv.org/abs/2408.08535
9. Parth Sarthi, Salman Abdullah, Aditi Tuli, Shubh Khanna, Anna Goldie, Christopher D. Manning, RAPTOR: Recursive Abstractive Processing for Tree-Organized Retrieval, Jan 2024, https://arxiv.org/html/2401.18059v1
10. Zecheng Tang, Keyan Zhou, Juntao Li, Baibei Ji, Jianye Hou, Min Zhang, L-CiteEval: Do Long-Context Models Truly Leverage Context for Responding?, Oct 2024, https://arxiv.org/abs/2410.02115

