Mini Project: AI Multi-Document Chatbot with Memory using RAG

Project Title (IEEE Style)

“An Intelligent Multi-Document Conversational Assistant Using Retrieval Augmented Generation and Large Language Models”

Abstract

This project proposes an intelligent conversational chatbot capable of understanding and answering user queries based on multiple uploaded documents. Traditional search systems fail to provide contextual answers across large document collections. The proposed system uses Retrieval Augmented Generation (RAG) combined with Large Language Models (LLMs) to extract relevant information from documents and generate context-aware responses. The chatbot supports PDF document ingestion, semantic search, conversational memory, and natural language interaction. This system can be applied in education, enterprises, legal document analysis, and knowledge management systems.

Objectives

- To build a chatbot that can answer questions from multiple documents
- To implement semantic search using embeddings
- To enable conversational memory
- To provide accurate context-aware responses
- To create an interactive web interface

Problem Statement

Organizations and students often deal with large volumes of documents where extracting specific information manually is time-consuming and inefficient. Existing keyword-based search systems fail to understand context and intent. There is a need for an intelligent system that can understand natural language queries and retrieve precise answers from multiple documents efficiently.

Proposed Solution

The system uses a Retrieval Augmented Generation framework where documents are converted into embeddings and stored in a vector database. When a user asks a question, the system retrieves relevant document chunks and sends them to a Large Language Model to generate a contextual answer.

System Architecture

User → Web Interface → Document Loader → Text Splitter → Embeddings → Vector DB → Retriever → LLM → Response

---

Tech Stack

Frontend

- Streamlit

Backend

- Python

AI/ML

- LangChain
- LLM API (OpenAI / Gemini / Ollama)
- Sentence Transformers

Database

- FAISS / ChromaDB

Workflow

User uploads PDF documents
Documents are converted to text
Text is split into chunks
Embeddings are generated
Stored in vector database
User asks question
Relevant chunks retrieved
LLM generates response
Chat memory maintained

Key Features

Multi-document support
Context aware answers
Chat history memory
Fast semantic search
Simple UI
Offline support (with Ollama)

Applications

- Educational assistants
- Enterprise knowledge bots
- Legal document analysis
- Research paper summarization
- Customer support systems

Future Enhancements

- Voice interaction
- Multi language support
- Cloud deployment
- Fine tuned model
- Role based access

IEEE Paper References

You can cite these

Retrieval Augmented Generation for Knowledge Intensive NLP Tasks — Facebook AI Research

Dense Passage Retrieval for Open Domain Question Answering

Attention Is All You Need — Transformer architecture

Language Models are Few Shot Learners — GPT

Modules

- Document Processing Module

- Embedding Generation Module
- Vector Storage Module
- Query Processing Module
- Chat Interface Module