194.207-2025W Generative AI

Project Plan

“ChatPDF - Generative AI Research Assistant for Academic Users”

Aya Wahbi (01427598)
Babak Bayani (12347302)
Benedikt Widy (01026787)
Muhammad Awais Riaz (12443585)

The idea is to implement a tool that is capable of taking multiple, locally stored text based documents (mainly research papers), analyze them and then be capable of chatting and answering queries about those documents’ details.

Users

Our primary users are students and academic researchers, a group that is highly motivated to engage with academic content while constantly seeking more efficient ways to manage and extract insights from research literature.

Students (Undergraduate, Graduate, PhD)

Goals:

- Complete assignments and writing tasks (essays, theses).
- Prepare for exams and conduct literature reviews.

Current Workflow:

- Download research papers (PDFs) or other necessary documents from university libraries, databases, or journal websites.
- Organize papers locally (by course or topic).
- Read, highlight,and write notes.

- Manually search through collections for specific facts or arguments.

Key Challenges:

- Feeling overwhelmed by the volume of available papers.
- Difficulty recalling specific details from previous readings.
- Challenges in synthesizing information across multiple sources.
- Excessive time spent on searching rather than analysis.

Academic Researchers (Post-docs, Professors, Research Scientists)

Goals:

- Stay current with new research and identify gaps in the literature.
- Design impactful experiments, write grant proposals, publish papers, and teach.

Current Workflow:

- Maintain a large, specialized collection of research papers.
- Engage in deep critical analysis.
- Rely on manual retrieval processes despite increasingly complex research collections.

Key Challenges:

- The “needle in a haystack” problem when searching through hundreds or thousands of papers.
- Rapid synthesis of interdisciplinary findings and comparison of methodologies/results.
- Balancing the constant influx of new research with the need for in-depth study.

Data

Given our target users and their challenge of extracting specific facts from numerous research PDFs, the data landscape is as follows:

Information Sources and Formats:

Primary Data are academic research papers in PDF format (with a focus on text-based PDFs).

Focus for the Content Format of the Prototype:

- Abstracts.
- Introductions.
- Main body text (methods, results, discussion).
- Conclusions.

Storage and Organization:

The text based files are to be saved and organized.

Storage Locations:

- Local folders organized by course, topic, or project.
- Potential cloud storage (e.g., Google Drive, OneDrive, Dropbox).

Organization Dimensions:

- Granularity: Research papers are lengthy documents that contain many discrete ideas. Our system will extract these atomic insights for easier access.
- Connections: Semantic relationships among papers (e.g., citations or thematic links) are implicit and can be inferred by the AI.
- Completeness: The dataset consists of polished, peer-reviewed articles, easing the extraction process.
- Context: Metadata (title, authors, publication year, journal, abstract) and file organization provide valuable context.
- Heterogeneity: For the prototype, a focused dataset from a single academic domain (e.g., computer science, biology, social sciences) will maintain consistency.

The Problem

Academic users face significant challenges when trying to efficiently extract, synthesize, and leverage information from a vast collection of research papers. Major issues include:

Retrieval Burden:
“I know I read something about X, but I can’t quickly locate it.”
Manual searches (e.g., Ctrl+F) are time-consuming and may not capture nuanced phrasing.

Synthesis Block:
“I have many papers but cannot easily see the big picture or compare findings.”
Synthesizing and comparing methodologies, results, and conclusions manually is demanding and error-prone.

Information Overload & Cognitive Strain:
Continuous new research combined with extensive archives forces users to spend excessive time managing information rather than generating insights, thereby reducing overall productivity.

Lack of Granular Access:
Critical insights and specific data points are buried within lengthy documents, and current tools offer minimal help in isolating these elements across diverse content.

The Solution

Our generative AI application will serve as an intelligent research assistant, transforming a vast collection of research papers into an organized, queryable, and insightful knowledge base. It will empower academic users to extract, synthesize, and understand information more effectively.

The Core Concept: **“Chat with Your Research Papers”**
Users can pose natural language queries across their entire document repository, essentially consulting with an extremely knowledgeable colleague.

Key Functionalities and Their Impact:

Intelligent Cross-Document Search & Question Answering

> Functionality: Users upload their Document Collection and ask natural language queries (e.g., “What are the common challenges in applying machine learning to medical imaging?”). The system retrieves concise, directly cited information from relevant sections.

Impact: Reduces manual search time and provides granular access to the information needed.

Automated Summarization & Key Point Extraction

Functionality: Generate concise summaries for individual papers or clusters (including main arguments, methodologies, and findings) while extracting key facts.

Impact: Allows users to quickly grasp the essence of a document, aiding in prioritization and recall.

Comparative Analysis & Synthesis Assistance

Functionality: Enable queries like "Compare the methodologies of Author A and Author B" or "What conflicting findings exist on topic X?" to synthesize comparative insights from multiple documents.

Impact: Facilitates a deeper understanding of relationships and trends across research papers, helping users identify research gaps.

Prototype Focus

We plan to use the Retrieval-Augmented Generation (RAG) Approach, this includes the following steps:

1. Text extraction from Documents.
2. Chunking the text into manageable segments.
3. Transforming chunks into vector embeddings.
4. Indexing these embeddings in a vector database.
5. Retrieving relevant chunks via similarity search.
6. Generating contextually grounded answers using a large language model, complete with source citations.

Metrics

We will assess the effectiveness of our research assistant using metrics that capture efficiency, accuracy, and user experience:

Time-to-Information Retrieval (Efficiency)

Measures:

- Task completion time for search queries compared with baseline manual methods.

Answer Relevance and Accuracy (Quality)

Measures:

- User ratings on a standardized relevance/accuracy scale.
- Verification of source citations aligning with provided answers.

User Satisfaction and Perceived Productivity (User Experience)

Measures:

- Qualitative feedback about overall experience and reduced information overload.

Prototype Development Plan

Our objective is to build a working prototype that demonstrates the core value of an intelligent, queryable research PDF repository. The project will be executed in the following phases:

Phase 1: Data Ingestion & Preprocessing

PDF to Text Extraction: in this phase we will use **Unstructured** Python library to automatically detect and extract text, tables, and metadata from many file types (`.txt`, `.md`, `.csv`, `.json`, `.rtf`, `.pdf`, `.docx`, `.pptx`, `.xlsx`, `.html`, `.htm`, `.eml`).

Text Chunking: then we divide the extracted text into manageable segments (e.g., 500–1000 words per chunk, with overlapping context).

Phase 2: Embedding & Indexing

Text Embedding: we will use pretrained models (e.g., sentence-transformers like all-MiniLM-L6-v2) to convert text chunks into vector embeddings.

Vector Database Indexing: Store and index embeddings using vector databases (e.g., FAISS or ChromaDB) for rapid similarity searches.

Phase 3: Querying & Generation

User Query Processing: Convert natural language queries into vector embeddings using the same model.

Relevant Chunk Retrieval: Perform similarity searches in the vector database to obtain

pertinent text segments.

Answer Generation with LLM: Construct prompts that integrate the user's query with retrieved chunks and use a large language model (via local solutions like Ollama or API-based models like OpenAI) to generate coherent, contextually grounded answers with source citations.

Phase 4: User Interface Development

professional GUI, native desktop application using **PyQt5 / PySide6.**

Phase 5: Testing & Iteration

Internal Testing:

Validate the prototype using a controlled set of Documents and queries, 2-3 types of research paper subject (e.g., computer science, biology, social sciences)

User Feedback:

Collect detailed input from a small group of students and researchers

Refinement:

Iteratively improve text chunking strategies, prompt engineering, embedding quality, and LLM parameters based on feedback.