Project Proposal – Personal Document RAG System with Advanced Chunking and Indexing

Name: Sharad Talekar
StudentID: 23881300
Course: INFO556 – Text Retrieval and Web Search

Purpose

Traditional search approaches are often inadequate for managing personal document collections. Users accumulate hundreds of documents like research papers, course materials, technical manuals, personal financial documents etc. Retrieving specific information typically requires manual searching through each individual files. Keywords based search provides some assistance but returns the whole documents, not concise answers. Although, there are some cloud-based solutions, but they raise privacy concerns for sensitive personal data. These concerns are elevated in contexts where sovereign control over data storage is required.

The solution – A Retrieval Augmented Generation (RAG) system using Llama2 and LlamaIndex that enables interaction with private documents like PDF that look like a conversation. Unlike the LLM based chatbots, this refers user's personal documents and provides cited answers while maintaining local data control as the query and data does not leave the user's local system.

Key focus areas

I want to investigate the impact of chunking, context window on the search results, especially when there are diverse document types. Also, there are many LLM models available that can run on user's personal computer, I want to see which one is light weight such that it can do the job efficiently without the need for heavy compute.

Motivation

Privacy and trust – Sensitive documents like financial, medical or proprietary documents cannot use cloud APIs
IR Learning goal – Having worked in a RAG project on Azure infrastructure as an Infrastructure engineer, I am curious on how the whole system is implemented and deep dive on RAG systems.

Acceptance criteria

Users can query their documents conversationally like "What was my blood sugar level after last test?" for personal data

Main Functions

Document Ingestion and Preprocessing:

- PDF parsing with layout preservation
- Metadata extraction like title, author, date etc.
- Text cleaning and normalization

Chunking:

- Determining the right balance chunk size so that the context is not lost and explore the possibility of adaptive sizing.
- Preserving the semantics while chunking.

Indexing:

Use dense embeddings for semantic similarity search.

Retrieval:

- Retrieve top-k most relevant chunks
- Include surrounding context using neighboring chunks

Answer generation with LLMs:

- For every generated answer, the system must provide inline citations.
- Provide answers like "I don't know" when documents lack information (optional).

User Interface:

- Command line interface for queries.
- Display retrieved chunks along with generated answers and citation.
- (optional): UI for document upload or chat.

Implementing Approaches:

1. Document loading
    a. Use SimpleDirectoryReader to get the pdf files in the directory.
    b. Optional – get files from UI
2. Document Preprocessing and Chunking
    a. Tokenize, clean, split into manageable chunks
    b. Try overlap and variable size chunks
3. Query
    a. Create a system prompt and a wrapper formatted for Llama2 or other LLM
4. LLM
    a. Use HuggingFace CLI to access models
    b. Choose model like Llama2 or any other with parameters like context window, token generation, quantization
5. Service context and indexing configuration
    a. The LLM, embedding model, chunk size all combined into a service context.
    b. Service context bundles resources for indexing and querying
    c. Loaded documents are indexed using vector store index and service context.
6. Query
    a. Query the system via cli to generate the response

Note: *These steps are to get a minimum viable RAG implementation working, several modules might be modified with above goal in mind.*

Expected outcomes and timeline

Deliverables

1. Functional RAG system with CLI
2. Documentation and code repository
3. Final report analyzing results and lessons learned

Timeline

Week 1	10/05	Project proposal
Week 2	10/13	Environment setup, PDF parsing, minimal implementation.
Week 3	10/20	Chunking and indexing completion
Week 5	11/3	Ranking and answer generation
Week 7	11/17	Final touch and Demo preparation
Week 8	12/1	Project Demo
Week 9	12/8	Project report and code

References

LlamaIndex Documentation: https://developers.llamaindex.ai/python/framework/

LangChain Documentation: https://python.langchain.com/docs/tutorials/rag/

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, Sebastian Riedel, and Douwe Kiela. 2020. Retrieval-augmented generation for knowledge-intensive NLP tasks. In Proceedings of the 34th International Conference on Neural Information Processing Systems (NIPS '20). Curran Associates Inc., Red Hook, NY, USA, Article 793, 9459–9474. https://dl.acm.org/doi/pdf/10.5555/3495724.3496517

Asai, A., Wu, Z., Wang, Y., Sil, A., & Hajishirzi, H. (2023). Self-RAG: Learning to Retrieve, Generate, and Critique through Self-Reflection. *ArXiv, abs/2310.11511*. https://doi.org/10.48550/arXiv.2310.11511