DS 4300 - Spring 2025

Practical 02

Due:

- Project needs to be functional for Exam on March 24.
- Final deliverable and repo to be submitted by March 26 @ 11:59pm.

Overview:

In this project, you and your team will build a local Retrieval-Augmented Generation system that allows a user to query the collective DS4300 notes from members of your team. Your system will do the following:

1. Ingest a collection of documents that represent material, such as course notes, you and your team have collected throughout the semester.
2. Index those documents using embedding and a vector database
3. Accept a query from the user.
4. Retrieve relevant context based on the user's query
5. Package the relevant context up into a prompt that is passed to a locally-running LLM to generate a response.

In this, you'll experiment with different variables - chunking strategies, embedding models, some prompt engineering, local LLM, and vector database options. You'll analyze how these changes affect the system's performance and output quality.

Teams and Corpus:

Each team should be composed of 2 - 4 members. They can be the same as Practical 01, but teams may switch up.

Each team should gather a collection of course notes taken by the team members. This can be the slide decks, personal notes taken throughout, and additional documentation for the tools/systems we have used.

Tools:

- Python for building the pipeline
- Ollama for running LLMs locally (you'll compare and contrast at least 2 different

models
- Vector Databases (Redis Vector DB, Chroma, and one other of your choosing)
- Embedding Models (you'll compare and contrast at least 3 different options)

Variables to Explore:

1. Text preprocessing & chunking
    a. Try various size chunks: 200, 500, 1000 tokens, for example
    b. Try different chunk overlap sizes: 0, 50, 100 token overlap for example
    c. Try various basic text prep strategies such as removing whitespace, punctuation, and any other "noise".
2. Embedding Models - Choose 3 to compare and contrast. Examples include:
    a. sentence-transformers/all-MiniLM-L6-v2
    b. sentence-transformers/all-mpnet-base-v2
    c. InstructorXL
    d. The model we used in class

Measure interesting properties of using the various embedding models such as speed, memory usage, and retrieval quality (qualitative).

3. Vector Database - At a minimum, compare and contrast Redis Vector DB and Chroma (you'll have to do a little research on this db) and one other vector database you choose based on research. Examine the speed of indexing and querying as well as the memory usage.
4. Tweaks to the System prompt. Use the one from the class example as a starting point.
5. Try at least 2 different local LLMs. Examples include Llama 2 7B and Mistral 7B. You aren't required to use these two specifically, however.

Suggested Steps:

1. Collect and clean the data.
    a. If you're using PDFs, review the output of whichever Python PDF library you're using. Is it what you expect?
    b. Do you want to pre-process the raw text in some way before indexing? Perhaps remove extra white space or remove stop words?
2. Implement a driver Python script to execute your various versions of the indexing pipeline and to collect important data about the process (memory, time, etc). Systematically vary the chunking strategies, embedding models, various prompt tweaks, choice of Vector DB, and choice of LLM.
3. Develop a set of user questions that you give to each pipeline and qualitatively

review the responses.
4. Choose which pipeline you think works the best, and justify your choice.

Deliverables:

As a team, you'll produce a slide deck communicating your findings as well as your final choice of pipelines with justification. Be specific. (Template for deck will be forthcoming.)

You will also include a public GitHub repository containing well-organized set of scripts related to the various pipelines your team tests. The README should describe how to execute your project.

More details on deliverables will be shared soon.

Areas for Evaluation:

1. Robustness of Experimentation (30%)
2. Analysis of collected data (30%)
3. Recommendation of pipeline organizations (20%)
4. Professionalism of Slide Deck (20%)