CSC 4700/HNRS 3035 Computer Science Homework #4

Learning Objective: To learn how to use leverage Retrieval Augmented Generation (RAG) with OpenAI embeddings and ChromaDB to enable LLMs to leverage external information to more accurately answer questions.

Assignment Objective: This assignment builds off Homework #3. The code you submit for this homework should be a **modified version of your code for Homework #3**. You can refer to the homework 3 solution, but your code must be your own. You will evaluate a variety of API-based LLMs on a subset of the Stanford Question Answering Dataset (SQuAD2.0), but this time, you will implement RAG, providing context to the model to answer the questions. You will then use GPT-4o-mini to score the correctness of the responses for each model.

Instructions and Questions:

1. **Download** the SQuAD2.0 Dev Set v2.0 here. You should **extract all of the context chunks** in the database and **store them in a Chroma database**. For the embeddings, you should use the following **OpenAI** embedding model: "**text-embedding-3-small**". Each context chunk in the SQuAD json file you downloaded are stored with the "context" key.
2. You should extract the **first 500 questions and potential answers** from the dataset. You will use these in the following steps. Some questions are considered as "is_impossible" in the dataset. **You should skip these when extracting the first 500 questions.**
3. Implement RAG into your GPT-4o-mini Q/A pipeline. Your code should take the question, query **5 chunks** from the ChromaDB using vector semantic search, and supply these to the model along with the question to generate the answer. Use the RAG pipeline to have the model answer the 500 questions, via the OpenAI **batch mode**.
    a. **Question 1**: What prompt did you use?
4. Repeat the last step, but this time, use the **Azure deployment of Llama 3.2 11B Vision Instruct** (**serially**, one question after another) to generate answers to all of the questions from step 1.
    a. **Question 2**: What prompt did you use?
5. Using the OpenAI API, use **GPT-4o-mini** (specifically, **gpt-4o-mini-2024-07-18**) in batch mode to score the responses of GPT-4o-mini and Llama as either **True** (correct) or **False** (incorrect). Save the outputs for each model in a Json file that you can use later, these files should be named appropriately, such as "gpt-4o-mini-RAG-DATE-hw4.json". You should use the **structured_outputs** feature to ensure the model outputs both an explanation (str) and a score (boolean). You should use the prompt shown below.
    a. **Question 3:** What was the average score for GPT-4o-mini? How does this compare to the result from Homework 3?
    b. **Question 4**: What was the average score for Llama? How does this compare to the result from Homework 3?

Code Submission: You should submit **only** your source code. You **do not** need to submit the batch files, model outputs, or database. Your code should be **commented** and **generally** styled in accordance with **PEP-8**. Use snake_case, not CamelCase/camelCase for function and variable names. Use CamelCase for class names.

Scoring Prompt:
"You are a teacher tasked with determining whether a student's answer to a question was correct, based on a set of possible correct answers. You must only use the provided possible correct answers to determine if the student's response was correct.

Question: {question}

Student's Response: {student_response}

Possible Correct Answers:
{correct_answers}

Your response should only be a valid Json as shown below:
{{
"explanation" (str): A short explanation of why the student's answer was correct or incorrect.,
"score" (bool): true if the student's answer was correct, false if it was incorrect
}}

Your response: "