CIS 620, Advanced Topics in Deep Learning, Spring 2024

Homework 5
Due: Monday, February 26, 11:59pm
Submit to Gradescope

Learning Objectives

After completing this assignment, you will:

- Be able to use Langchain
- Design a network and understand how LLMs like ChatGPT can write to and read from databases

Deliverables

This is a **pair** assignment for both the written and coding portions. - **One submission per pair** *(you may also do this individually if you wish)*

1. **A PDF with your name in the agreement**

   Copy and edit this googledoc to enter your names in the Student Agreement and answer the questions mentioned below. **Don't forget to add your team member on the gradescope submission!**

2. **hw5.ipynb file with the functional code**

   Complete the coding assignment in the Jupyter Notebook and upload the file to Gradescope. **Don't forget to add your team member on the gradescope submission!**

Note that there is a separate assignment for the papers, which will be handed in separately.

Homework Submission Instructions

Written Homeworks

All written homework must be submitted as a PDF to Gradescope. **Handwritten assignments (scanned or otherwise) will not be accepted.**

Coding Homeworks

All coding assignments will be done in Jupyter Notebooks. We will provide a .ipynb template for each assignment as well as function stubs for you to implement. You are free to use your own installation of Jupyter for the assignments, or Google Colab, which provides a Jupyter Environment connected to Google Drive along with a hosted runtime containing a CPU and GPU.

Questions

1. **Implement QA retriever (https://github.com/PratikSingh121/Langchain/blob/main/Langchain_QARetriever_and_Pinecone.py) and answer the following questions.**
    a. **How is OpenAI's ChatGPT called using langchain? What version and default settings of ChatGPT as used in this function call?**
    b. **What is the dimension of pinecone used in the implementation? What is meant by the *dimension* and *metric* parameters passed in when creating an index in pinecone?**
    c. **Run the code with necessary changes like API key and show the outputs obtained for 3 custom prompts passed to QA Bot.**
    d. **What is the impact of changing the RecursiveCharacterTextSplitter()'s chunk size and chunk overlap parameters? Support your answer with tabular or graphical comparisons obtained by making the relevant changes to the code implemented in part c.**
2. Design Question: Design and implement a program that utilizes Langchain and ChatGPT to provide users with personalized Greetings, everytime the user asks a question to the QABot.
    a. The first time the user asks a question, ChatGPT should be prompted to ask basic questions to the user. **Hint: An instruction like 'Gather basic information by talking to the user to be able to provide a personalized greeting' can be given to ChatGPT.**
    b. Use ChatGPT to store all the information in a relevant datastore.
    c. The next time the user starts a new conversation (where GPT does not have the previous conversation in context) and asks a question , the answer from ChatGPT should start with a personalized Greeting such as "Hi Lyle, I hope your day is going well!". Here there are two tasks: (1) ChatGPT must generate a greeting based on the stored data, and (2) ChatGPT must provide a response to the question. These two results need to be concatenated and given as a response to the user. **Again: The retrieval needs to be out of the context window and from the database, i.e., consider the user logging back in after logging out. See https://www.wired.com/story/chatgpt-memory-openai/ for inspiration.**

    For the HW you can do the design question considering a single user.

3. Describe one fun application you could make using the concepts covered this week (RAG, Toolformers, and Langchain). Elaborate on how you would design the system.

Bonus Question:

1. In the design question (Question 2 above) how would you handle multiple users.

Resources:

1. https://github.com/kyrolabs/awesome-langchain?tab=readme-ov-file
2. https://github.com/PratikSingh121/Langchain/blob/main/Langchain_QARetriever_and_Pinecone.py
3. https://medium.com/@joristechtalk/kickstart-your-ai-journey-with-langchain-10-exciting-project-ideas-62b27d67b743
4. https://www.wired.com/story/chatgpt-memory-openai/

ANSWERS:-

1. a) We import ChatOpenAI from langchain.chat_models.

   Upon reviewing the source code, the default model used is gpt-3.5-turbo.
   model_name: str = Field(default="gpt-3.5-turbo", alias="model").
   Also, in the similarity search, the OpenAI chatbot receives 4 document segments with the prompt to produce the output.

   b) The dimensions of pinecone used in this implementation are 1536. The dimension parameter we pass when creating the index refers to the size of the vector embeddings the index will store and query on. The metric used in our case is “cosine” because we intend to compare the query vector with the vectors in the vector database to find the most relevant texts for our query.

c) The three custom prompts passed to the QA bot are as follows,

Prompt/Question	Answer (chunksize=1000 & overlap=200)
Who is meta?	Meta is a company that presented a solution in a recent paper that allows large language models (LLMs) to use external tools via simple APIs, achieving the best of both worlds. They also developed Toolformer, a framework that integrates a range of tools, including a calculator, a Q&A system, two search engines, a translation system, and a calendar.
What is a toolformer?	A toolformer is a framework developed by Meta AI that integrates large language models with external APIs, allowing them to use various tools such as calculators, Q&A systems, search engines, translation systems, and calendars. This helps to solve problems with the accuracy and reliability of information generated by LLMs.
What is an llm?	An LLM is a large language model, a type of AI technology that is designed to generate natural-sounding text.

d)

Question	chunksize=1000
	overlap=200
		chunksize=500
		overlap=200
			chunksize=500
			overlap=50

Who is meta?	Meta is a company that presented a solution in a recent paper that allows large language models (LLMs) to use external tools via simple APIs, achieving the best of both worlds. They also developed Toolformer, a framework that integrates a range of tools, including a calculator, a Q&A system, two search engines, a translation system, and a calendar.	Meta is a company that has recently published a paper on a solution for LLMs using external tools via simple APIs. They have also developed Toolformer, a framework that integrates various tools to help solve crucial LLM problems.	I don't know.
What is a toolformer?	A toolformer is a framework developed by Meta AI that integrates large language models with external APIs, allowing them to use various tools such as calculators, Q&A systems, search engines, translation systems, and calendars. This helps to solve problems with the accuracy and reliability of information generated by LLMs.	Toolformer is a new approach to combining large language models with external APIs, allowing for more accurate and reliable information retrieval from the web.	I don't know.
What is an llm?	An LLM is a large language model, a type of AI technology that is designed to generate natural-sounding text.	An LLM, or large language model, is a type of AI system that is trained to understand and generate natural-sounding text. It is often used for tasks such as language translation, question answering, and text generation. However, LLMs have their own set of challenges and limitations that researchers are working to overcome.	I'm sorry, I cannot answer this question as it is not mentioned in the context.

2. a)
   For the first time the user asks a question, we will directly ask the user to describe themselves before entering the conversation session. We then pass the response to the prompt in ChatGPT to generate a greeting and answer consistent with the response.’

   b) We use SummaryChain to summarize a conversation into several sentences and store the summary for the next session.

   c) We use “id” to identify the users if there is a summary of the user in the storage (a Python dictionary). If a summary can be found, we pass the summary to the ChatGPT with the question. If not, we will go through the steps in 2a.

3. We can use langchain or toolformer to decorate the LLM as a personal assistant. Using vector datasets and APIs, we can ask the assistant to reserve a restaurant for us:
   - LLM can use a similarity search to find out what type of restaurants we like.
   - Then, using API from Yelp or Google Maps to collect restaurant information near us, and make a reservation through the API.
   - Finally, the assistant further uses API to set the schedule on the calendar like Google Calendar.

4. We handle multiple users by using a sign-up mechanism. We store information of every user in a dictionary. As a person exits a session and a new user enters, we just add his name and information to the dictionary.
   In every session, we continue to run the same code as in the second question and if the user types”q” or “quit”, they quit the session but if they type “exit,” they stop the whole cell from running.

   A supplemental is also included in the end that describes another way to handle multiple users in a stable manner, which we had implemented earlier and found it interesting and decided to include in the notebook.