Documentation for Question-Answering System with LangChain and Open-Source LLM

Objective

The goal of this project is to develop a question-answering system that utilizes LangChain and an open-source language model, such as GPT-2 or GPT-J. The system will take user input and provide answers based on a specified context.

Task 1: Setting up the Environment

Python Virtual Environment

A virtual environment was created to isolate the project dependencies. This ensures that the required packages do not conflict with other projects.

Required Packages

The following libraries were installed:

LangChain: A framework designed for building applications powered by language models.

Transformers: A library for working with various pre-trained language models, including GPT-2 and GPT-J.

Torch: A deep learning framework essential for running the language models.

Verification

The installation was verified by checking the list of installed packages to confirm that all necessary libraries were successfully added.

Task 2: LLM Integration with LangChain

Library Imports

The project begins by importing the necessary libraries, including LangChain's components for managing language models, prompts, and document loading.

Loading Context

A sample context was prepared, which provides relevant information that the language model will use to answer questions. This context serves as a foundation for generating responses.

Language Model Initialization

An open-source language model (e.g., GPT-2) was initialized to handle the question-answering tasks. This model serves as the core engine for generating answers based on the input questions and context.

Creating the QA Chain

A question-answering chain was established using LangChain. This chain integrates the language model with a prompt template that formats the input data.

Answer Retrieval Function

A function was defined to process user questions and return answers by utilizing the QA chain. This function takes user input and leverages the language model to generate a relevant response.

User Interaction

A simple loop was implemented to allow users to ask questions interactively. Users can input their questions, and the system responds with answers based on the context provided.

Task 3: Working with LangChain Prompt Templates

Custom Prompt Template

A custom prompt template was created to improve the interaction between the language model and the input data. This template specifies how the context and question should be presented, guiding the model to generate more accurate and concise answers.

Integration of Custom Prompt

The custom prompt template was incorporated into the QA chain, enhancing the model's ability to focus on the context and produce high-quality responses.

Performance Improvement

The custom prompt template improves the system's performance in several ways:

1.Clarity: By clearly separating the context and question, the prompt reduces ambiguity.

2.Guidance: It instructs the model to generate concise and accurate answers, steering it towards better performance.

3.Focus on Context: The structured approach emphasizes the importance of the provided context, ensuring that the model remains relevant to the user's query.

Conclusion

The project successfully demonstrates the creation of a question-answering system using LangChain and an open-source language model. By following the outlined steps, users can engage with the system to ask questions and receive context-based answers. The implementation of a custom prompt template significantly enhances the quality and relevance of the responses generated by the model.