TEJIS.AI

Proof of Concept (POC) Documentation

Name : Kiruthika V G
Roll No : 21PT10

Overview:

This Proof of Concept (POC) demonstrates the **integration of a graph database (Neo4j) with a Large Language Model (LLM) (Llama2)** using LangChain. The objective is to enable natural language querying of the database, with responses formatted in a human-readable manner. This POC uses a publicly available movie dataset.

Requirements:

- Neo4j Database: For storing and querying movie data.
- Neo4j Python Driver: For connecting and interacting with the Neo4j database from Python.
- LangChain: A framework to facilitate interaction between the database and the LLM.
- Hugging Face Transformers: For using the Llama2 model.
- Llama2 Model: A pre-trained language model available on Hugging Face. (model used: https://huggingface.co/daryl149/llama-2-7b-hf)

Dataset:

The sample dataset used is a movie dataset, which includes the following fields:

- title
- director
- duration
- genres
- actor
- language
- country
- year
- IMDb

Implementation Steps:

1. Load Data into Neo4j:

Load the dataset into Neo4j using a suitable Cypher query to create Movie nodes, with each node containing relevant attributes like title, director, duration, genres, etc.

2. Set Up Neo4j Connection:

Create a Python script to set up and test a connection to the Neo4j database. Ensure that the connection is successful by running a simple test query.

3. Integrate LangChain with LLM and Neo4j:

- Establish Database Connection: Use the Neo4j Python driver to establish a connection to the Neo4j database.
- Query Execution: Implement functions to execute queries on the Neo4j database and fetch movie details.
- LLM Integration: The Hugging Face Transformers library loads and interacts with the Llama2 model. Authenticate and fetch the model information from Hugging Face.
- Prompt Template: Create a prompt template to structure the input for the Llama2 model. The template should format the movie query response in a natural language format.
- Response Generation: Implement functions to format the query results into a prompt, run the Llama2 model pipeline, and generate human-readable responses.

4. Execute the POC:

Run the integrated system to perform a sample query like "What is the Genre of Titanic?" Ensure the system returns a response like "Titanic is of Romance, Adventure genre."

Conclusion:

This POC successfully demonstrates the integration of a graph database with a Large Language Model to enable natural language querying. This approach can be extended to other datasets and queries, demonstrating the versatility and power of combining graph databases with advanced language models.