Assignment 3

Part-I: Conceptual Understanding of RAG

Task 1: Short Answer Questions

1. What is the motivation behind Retrieval-Augmented Generation(RAG)?

The motivation behind RAG is to enhance the performance of language models by grounding their responses in external, domain-specific knowledge. This helps overcome limitations like hallucination, outdated knowledge, and lack of context by retrieving relevant documents during inference to generate more accurate and factual answers.

2. Explain the difference between RAG and standard LLM-based QA.

Standard LLM-based QA relies solely on the model's pre-trained internal knowledge, which may be incomplete or outdated. In contrast, RAG dynamically retrieves relevant information from external sources (like document databases) at runtime, combining retrieval with generation to provide more reliable and up-to-date answers.

3. What is the role of a vector store in a RAG pipeline?

A vector store stores the document embeddings, allowing for efficient similarity search. During retrieval, it helps find the most relevant chunks of text based on the semantic similarity between the user query and stored document vectors.

4. Compare “stuff”, “map_reduce”, and “refine” document chain types in LangChain.

- **“Stuff”** loads all retrieved documents into a single prompt, good for small data.
- **“Map_reduce”** processes each document individually (map) and combines results (reduce), scalable for large datasets.
- **“Refine”** iteratively builds an answer by refining an initial response with additional documents, offering a balance between completeness and token efficiency.

5. What are the main components of a basic LangChain RAG pipeline?

The main components include:

1. **Document Loader** (to ingest documents),
2. **Text Splitter** (to chunk large texts),
3. **Embeddings Model** (to convert text to vectors),
4. **Vector Store** (for similarity search),
5. **Retriever** (to fetch relevant chunks), and
6. **LLM Chain** (to generate responses using the retrieved context).

Task 2: RAG Pipeline Diagram

RAG Pipeline Flow Description

1. **User Query**

   → The user inputs a natural language question.

2. **Retriever**

   → The retriever takes the query and uses semantic similarity to search for relevant documents.

3. **VectorStore**

   → The retriever queries the vector store (which holds embeddings of document chunks) to find the top-N relevant documents.

4. **LLM(LanguageModel)**

   → The retrieved documents and the original query are passed to the LLM, which uses them as context to generate a grounded response.

5. **FinalAnswerGeneration**

   → The LLM outputs the final answer based on both the query and the retrieved context, reducing hallucination and improving accuracy.

[User Query]

↓

[Retriever]

↓

[Vector Store] ←→ [Embedded Documents]

↓

[Retrieved Documents + Query]

↓

[LLM]

↓