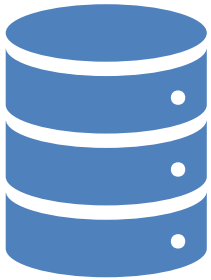
Introduction to LlamaIndex and Retrieval Augmented Generation (RAG)

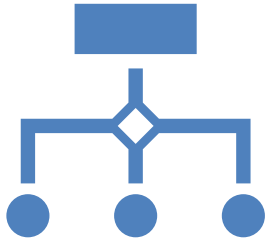
Enhancing Large Language Models with External Knowledge

Presented by: Mohammad Arshad

What is LlamaIndex?

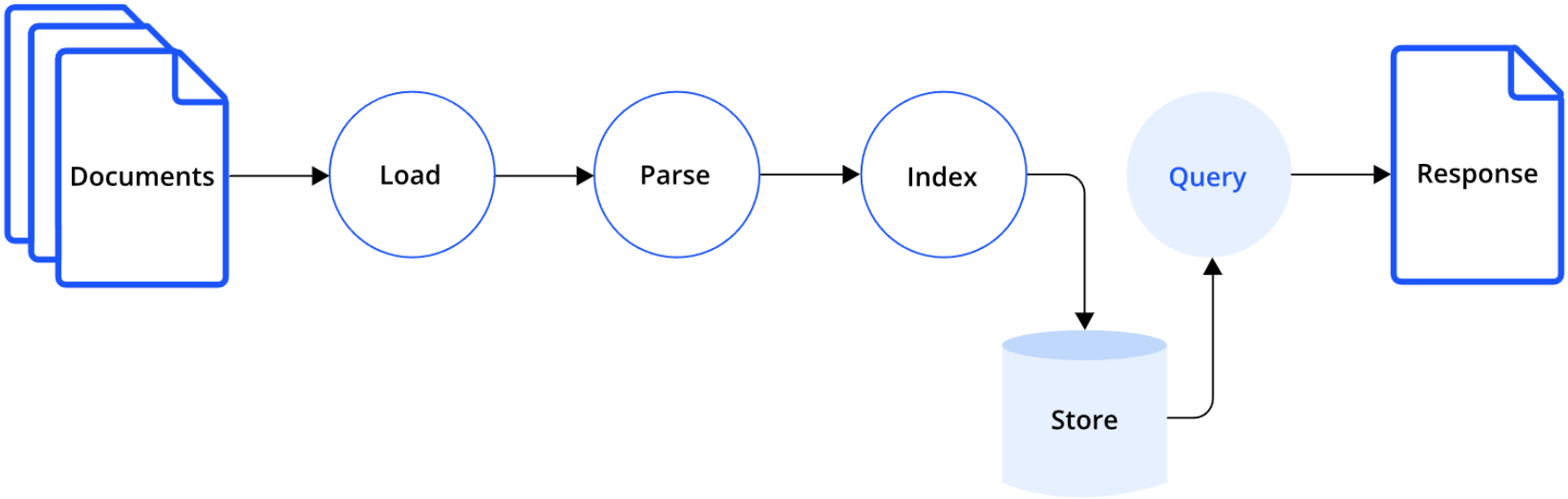


LlamaIndex is an open source project that allows users to build and query indexes over external data sources.



It enhances the capabilities of Large Language Models (LLMs) by providing them with access to structured data.

Designed to work with Retrieval Augmented Generation (RAG), a technique that improves LLMs by retrieving relevant information from external data.



What is Retrieval Augmented Generation (RAG)?

RAG is a framework that combines a retriever with a generator (e.g., a large language model).

The retriever selects relevant documents from a corpus or data source.

The generator (LLM) then uses this information to produce more accurate and context aware responses.

This approach helps overcome limitations of LLMs by enhancing their knowledge with external, up to date information.

Why Use LlamaIndex with RAG?

LlamaIndex helps bridge the gap between large language models and external databases.

LLMs can be limited to the information they were trained on (e.g., knowledge cutoff dates).

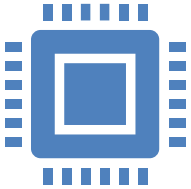
RAG enables LLMs to retrieve relevant, updated, and specific information from large external data sources.

Improves the accuracy and contextual relevance of generated responses by enhancing access to real time data.

Key Components of LlamaIndex

1**. Indexes:** Build indexes on external data (e.g., documents, databases).

2. **Query Engine**: Allows querying the indexed data using natural language queries.



3. **Retrieval Interface**: The interface between the LLM and the external data source to retrieve relevant information during query generation.

How LlamaIndex Works

1. **Data Ingestion :** LlamaIndex connects to external data sources (e.g., text files, databases) and builds an index.

2. **Query Processing :** When a question or query is posed to the LLM, LlamaIndex retrieves relevant information from the index.

3. **Response Generation :** The retrieved data is used by the LLM to generate a more informed and contextually accurate response.

Practical Applications of RAG and LlamaIndex

Customer Support : RAG can retrieve up to date support documents to answer user questions.

Legal and Compliance : Access to external legal databases to retrieve specific clauses or laws.

Healthcare : Use external medical research papers to generate accurate, evidence based responses.

Finance : Access to real time financial reports and data to generate market insights.

Hands On: Setting Up LlamaIndex with RAG

1. Install LlamaIndex : pip install llama index

2. Connect to a Data Source : Use LlamaIndex to ingest and index data from text files, databases, or APIs.

3. Query the Index : Use the LlamaIndex API to perform natural language queries on the indexed data.

4. Integrate with LLM : Combine the LLM with LlamaIndex to perform Retrieval Augmented Generation.

LlamaIndex Query Example

```
from llama_index.core import VectorStoreIndex, SimpleDirectoryReader
documents = SimpleDirectoryReader("data").load_data()
index = VectorStoreIndex.from_documents(documents=documents)
query_engine = index.as_query_engine()
response = query_engine.query("who is mentioned about in document")
print(response)
```

Benefits of Using LlamaIndex with RAG

Enhanced Knowledge : Access to a broader knowledge base beyond the LLM's training data.

Real Time Updates : Ability to fetch and retrieve the most up to date information.

Improved Accuracy : More contextually relevant and accurate responses to user queries.

Scalability : Works with large, external datasets or documents without overloading the LLM.

Conclusion

LlamaIndex enhances the capabilities of LLMs by integrating with external data sources.

Retrieval Augmented Generation (RAG) enables LLMs to produce more informed and accurate responses.

Together, LlamaIndex and RAG offer powerful solutions for data intensive and real time applications.

Questions?

Any questions on LlamaIndex or Retrieval Augmented Generation?

Further resources available on GitHub and official LlamaIndex documentation.