LangChain and Document Retrieval with TypeSense and FAISS

Mostafa Razavi Ghods

May 31, 2024

What is LangChain?

- LangChain is a framework for building applications powered by language models.
- It provides tools to manage and interact with language models effectively.
- LangChain simplifies the process of chaining multiple language model calls together.

What are Chains in LangChain?

- Chains are sequences of calls to language models or other computational steps.
  - Example: First, a text summarization model processes the input text, and then a translation model translates the summary into another language.
- They allow for complex workflows and tasks to be broken down into manageable steps.
  - Example: An email filtering system might first classify emails into different categories and then summarize the content of each email based on its category.
- Chains can include loops, conditionals, and parallel processing.
  - Example: A loop can be used to iterate through a list of documents, applying sentiment analysis to each one. Conditional statements can be used to handle different types of documents differently. Parallel processing can speed up tasks by processing multiple documents simultaneously.

Integration of TypeSense with LangChain

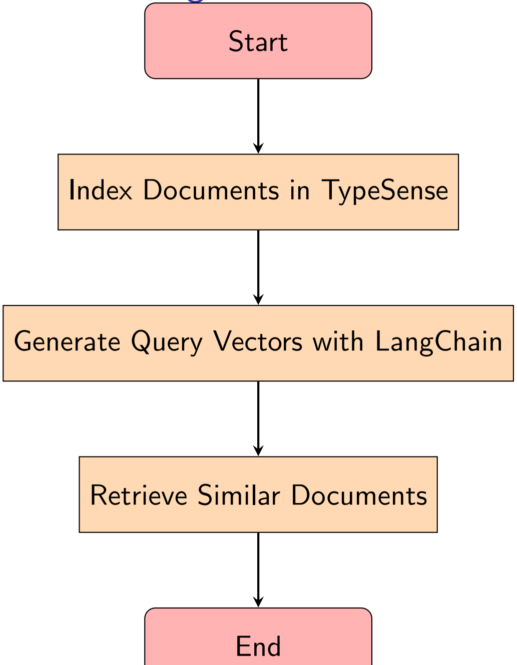
LangChain integrates with TypeSense to leverage its capabilities as a vector store while offering additional functionalities compared to a standalone FAISS library. Here's how it works:

- **Vector Embeddings:** LangChain first generates vector embeddings for your documents. These embeddings are mathematical representations that capture the semantic meaning of the text. You can use pre-trained models or custom methods to create these embeddings.
- **TypeSense Integration:** LangChain interacts with TypeSense through a wrapper class or library specifically designed for this integration. This wrapper class simplifies communication between LangChain and the TypeSense API.
- **Indexing Documents:** The LangChain tool using TypeSense adds the documents and their corresponding vector embeddings to a TypeSense collection. This collection acts like a specialized database optimized for fast vector similarity search.

Integration of TypeSense with LangChain

- **User Query:** When a user submits a query, LangChain sends it to the tool utilizing TypeSense.
- **Vector Similarity Search:** TypeSense performs a vector similarity search within the collection. It compares the user query's vector embedding to the embeddings of all the documents and retrieves the documents with the most similar vectors.
- **Additional Filtering (Optional):** A key advantage of TypeSense is the ability to combine vector search with attribute-based filtering. LangChain can leverage this feature to allow users to filter the search results based on document attributes (e.g., author, date, category) in addition to vector similarity.
- **Returning Results:** Finally, LangChain receives the most relevant documents from TypeSense and returns them to the user.

TypeSense Integration Workflow


Start
Index Documents in TypeSense
Generate Query Vectors with LangChain
Retrieve Similar Documents
End

Benefits of Using TypeSense with Vector Stores in LangChain

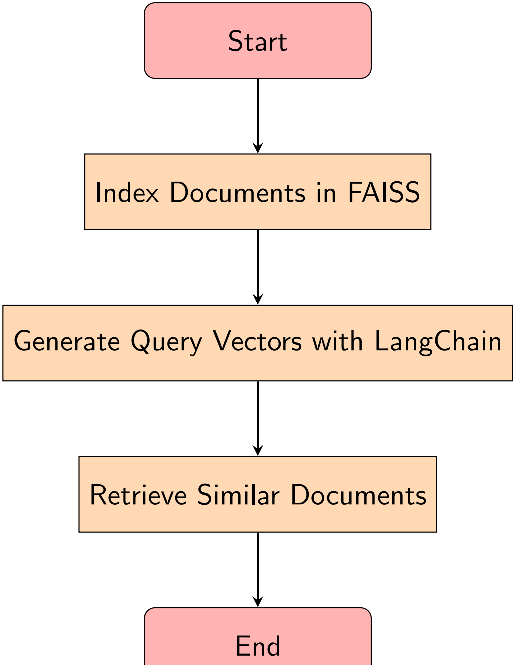
- **Faster Search:** TypeSense is optimized for fast in-memory search, making retrieval of similar documents efficient.
- **Richer Search Experience:** The ability to combine vector search with attribute filtering allows for more precise and user-friendly search experiences.
- **Easier Setup:** The LangChain integration with TypeSense simplifies the process compared to directly using FAISS.

In essence, LangChain uses TypeSense as a user-friendly wrapper around a vector store, enabling efficient vector similarity search with the added benefit of attribute-based filtering for a more comprehensive document retrieval experience.

Comparison: TypeSense vs FAISS

- **TypeSense:**
  - Designed for fast and relevant full-text search.
  - Provides real-time updates and is easy to integrate.
  - Better for small to medium-sized datasets.
- **FAISS:**
  - Developed by Facebook AI Research for efficient similarity search.
  - Optimized for large-scale datasets.
  - Supports various indexing methods for performance tuning.

Workflow with FAISS


Start
Index Documents in FAISS
Generate Query Vectors with LangChain
Retrieve Similar Documents
End

TypeSense vs FAISS in Finding Similar Documents

- **Performance:**
  - TypeSense excels in real-time, full-text search.
  - FAISS provides superior performance for large-scale vector searches.
- **Ease of Use:**
  - TypeSense is easier to set up and integrate for smaller projects.
  - FAISS requires more configuration but offers more control for large datasets.
- **Use Cases:**
  - TypeSense is suitable for applications needing instant search with frequent updates.
  - FAISS is ideal for applications dealing with extensive datasets requiring high retrieval speed.

Conclusion

- Both TypeSense and FAISS are powerful tools for document retrieval.
- Choice depends on dataset size, real-time requirements, and ease of integration.
- LangChain provides a flexible framework to leverage these tools effectively.