RAG-Based Agent

Using LangChain with a ReAct agent for reasoning, it integrates Qdrant as a vector database for context retrieval via a custom retrieve_context tool. It employs RedisChatMessageHistory for persistent conversational memory, allowing the agent to maintain context across turns.

Interview Booking Process:

A dedicated FastAPI endpoint captures the information, stores it in SQLite database and sends a confirmation email using smtplib to a configurable email address. An agent tool book_interview_tool is also provided for agent driven booking.

Qdrant Similarity Search Algorithm:

The algorithm demonstrates and compares Cosine similarity and Euclidean distances for vector search in Qdrant. It creates separate Qdrant collections for each metricand populates them with same documents.

Findings on Similarity Search Algorithms:

Cosine Similarity:

Measures the angle between two vectors, score closer to 1 indicates higher similarity while -1 indicates opposite directions.

It generally is more effective for text and high-dimensional data because it focuses on the orientation of vectors, representing semantic meaning, rather than magnitude. This makes it robust to differentiate in document length or word frequency.

Observation: We can find that Cosine similarity often returns more semantically relevant results for text queries, as its less influenced by the absolute “size of embeddings”.

Euclidean distance:

Measures the straight-line vector distance between two points in a multidimensional vector space, smaller distance indicates higher similarity. It is often preferred in scenario where the absolute difference between features is important, such as image processing and numerical dataset.

Observation: For text data Euclidean distance can sometimes be misleading. A long document can have very large vector magnitude, making it far from a short, semantically similar document in Euclidean space, even if their semantic direction is same.