

About our client

Our client provides a customer support service with the help of a chatbot in three industries: the Travel and transportation industry, Restaurant, food and Beverage industry and Financial services industry. The primary vision is to provide instantaneous responses to user queries and reduce the need for human intervention in routine queries.

Scope

The project's scope involves the introduction of Retrieval Augmented Generation (RAG) to improve the effectiveness of customer support Chatbot in providing contextually appropriate and accurate responses to customer queries.

The problem

In traditional natural language generation systems like Chatbot, generating contextually relevant and coherent content can be challenging also users often encounter issues related to incomplete or inaccurate answers, leading to a less-than-optimal customer experience. Hence, the challenge is to enhance the system's capabilities by incorporating advanced techniques such as vector-based retrievals and state-of-the-art language models.

Solution

Langchain integration:

We have integrated Langchain, a language model known for its contextual understanding and natural language generation capabilities. By training Langchain on diverse datasets with industry-specific information, the system can easily be adapted to current trends and user preferences.

Vector Database integration:

Firstly, all the documents, articles or pieces of information are represented in vectors by using certain techniques such as word embedding (Word2vec), sentence embedding (Doc2vec) or pre-trained language model embedding (BERT embeddings).

A Vector Database has been set to store the Vector representations of the respective documents along with the document indexing in the vector database.

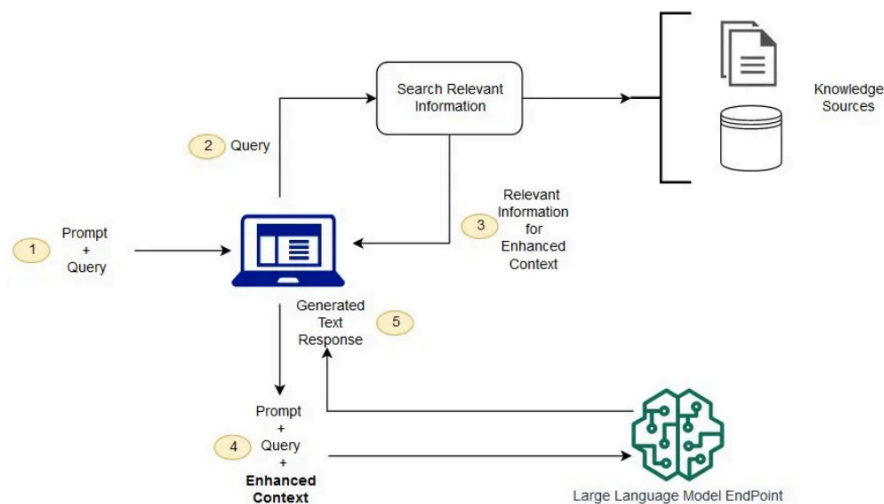
During the process of receiving a Query, the Query gets converted into a vector using an embedding technique which is a similar technique used for document vectors.

By using similarity metrics (cosine similarity, Euclidean distance, etc) vectorized user queries can be compared with vectors of documents stored in the vector database. This step is used to identify the documents that are most similar to the user's query.

After similarity scoring, we can retrieve the top-ranked documents based on similarity scores also these documents serve as the retrieved context for the RAG system.

Now, the retrieved document vectors should be integrated with the language model component of the RAG system and this involves passing the vectorized documents as input to the language model.

Using the language model and combined information from the retrieved documents and the user query, it is possible to generate context-aware and relevant content which constitutes the final response of the RAG system.



Results

The incorporation of Retrieval Augmented Generation (RAG) immensely improved the Chatbot's performance. Moreover, users experienced accurate, contextually relevant responses leading to higher customer satisfaction

Technologies used

Front End - HTML, CSS, ReactJS

Back End - NestJS, Python (Django)

Database - MongoDB

AI Prediction Model - UniLM (Unified Language Model), XLNet