

Retrieval-Augmented Generation (RAG) Model for QA Bot

Overview

This document outlines the implementation of a Retrieval-Augmented Generation (RAG) model designed for a Question Answering (QA) bot that retrieves information from a given dataset and generates coherent answers. The model leverages a vector database to store and retrieve document embeddings efficiently and utilizes a generative AI model to create responses based on the retrieved information.

Model Architecture

Components

1. **Data Loader:** Responsible for loading documents here we are using PDFs and extracting text for processing.
2. **Text Cleaner:** Cleans the extracted text to remove unnecessary characters and whitespace.
3. **Text Splitter:** Splits the cleaned text into manageable chunks suitable for processing and embedding.
4. **Embedding Generator:** Creates embeddings for the text chunks using a generative AI model.
5. **Vector Database:** A storage solution for document embeddings that enables efficient retrieval. In this implementation, we use Chroma as the vector store.
6. **Retrieval Mechanism:** Retrieves relevant document embeddings based on the user's query.
7. **Response Generator:** Utilizes the google generative AI model to produce coherent answers from the retrieved information.

Flow of Information

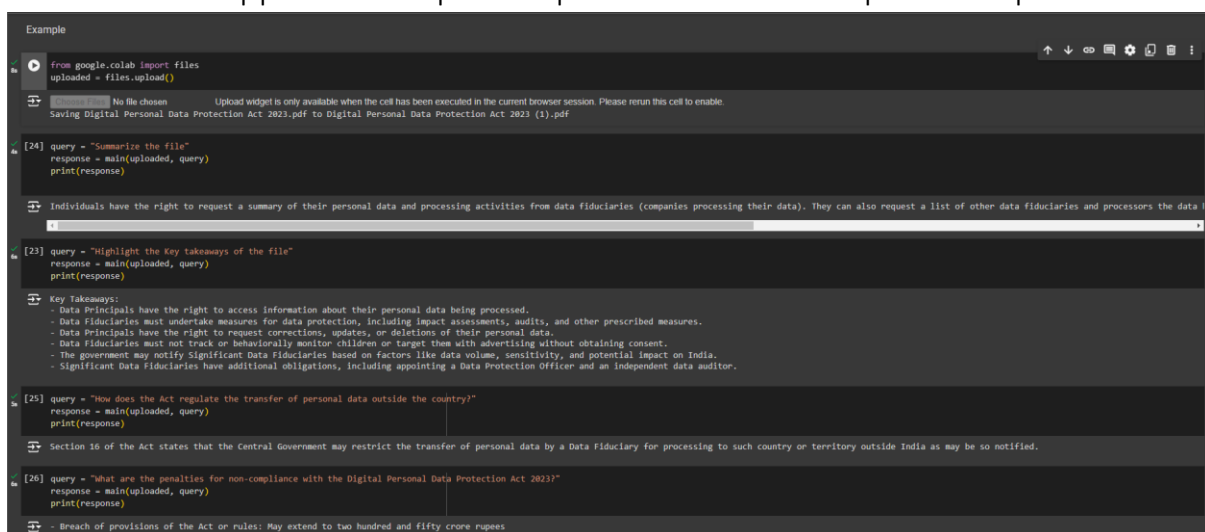
1. **Data Loading:** Load PDF files and extract text.
2. **Text Processing:** Clean and split the text into chunks.
3. **Embedding Creation:** Generate embeddings for each text chunk and store them in the vector database.
4. **Query Handling:** Receive user queries, retrieve relevant embeddings, and generate responses using the generative model.
5. **Output:** Display the generated response to the user.

Implementation Steps

1. **Data Loading and Preprocessing:** In the Colab notebook, PDF files are uploaded, and the text is extracted using the `PyPDF2` library. The text is then cleaned to remove any unwanted characters.
2. **Text Chunking:** The cleaned text is split into chunks using the `RecursiveCharacterTextSplitter` class, which helps manage the length of the text segments for efficient processing.
3. **Embedding Generation:** Using `GoogleGenerativeAIEmbeddings` from the `langchain_google_genai` library, embeddings for the text chunks are created. The embeddings are stored in **Chroma**, a vector database designed for efficient similarity searches.
4. **Query Handling and Response Generation:**
When a user submits a query:
 - The relevant document chunks are retrieved based on the embeddings.
 - The generative model processes the retrieved text to generate a coherent response.

Examples:

Below is the snippet of examples of queries and there response output



```
Example
from google.colab import files
uploaded = files.upload()

[24] query = "Summarize the file"
response = main(uploaded, query)
print(response)

[23] query = "Highlight the Key takeaways of the file"
response = main(uploaded, query)
print(response)

[25] query = "How does the Act regulate the transfer of personal data outside the country?"
response = main(uploaded, query)
print(response)

[26] query = "What are the penalties for non-compliance with the Digital Personal Data Protection Act 2023?"
response = main(uploaded, query)
print(response)
```

Individuals have the right to request a summary of their personal data and processing activities from data fiduciaries (companies processing their data). They can also request a list of other data fiduciaries and processors the data is shared with.

Key Takeaways:

- Data Principals have the right to access information about their personal data being processed.
- Data Fiduciaries must undertake measures for data protection, including impact assessments, audits, and other prescribed measures.
- Data Principals have the right to request corrections, updates, or deletions of their personal data.
- Data Fiduciaries must not track or behaviorally monitor children or target them with advertising without obtaining consent.
- The government may notify Significant Data Fiduciaries based on factors like data volume, sensitivity, and potential impact on India.
- Significant Data Fiduciaries have additional obligations, including appointing a Data Protection Officer and an Independent data auditor.

Section 16 of the Act states that the Central Government may restrict the transfer of personal data by a Data Fiduciary for processing to such country or territory outside India as may be so notified.

Breach of provisions of the Act or rules: May extend to two hundred and fifty crore rupees
Breach in respecting the obligation of Data Fiduciaries to take reasonable security safeguards: May extend to two hundred crore rupees

Conclusion

The implemented RAG model for the QA bot efficiently retrieves and generates answers from provided documents using a combination of text processing, embedding generation, and generative AI techniques. This solution can be adapted for various business use cases where quick and accurate information retrieval is required.