FINAL PROJECT REPORT

AskMeDoc –
An AI product, based on a large language model (LLM)

Group 16:
Bandigoda Kariyawasam
100867743
Maria Paula Olaya Navarro
100904624
Zubeda Gooni
100841390

Contents

1.0 Introduction

In the age of information, companies often find themselves submerged in vast amounts of documentation. Extracting relevant data and insights from these documents is vital for decision-making and growth. This report details an AI-powered Document-Based Question-Answering System designed specifically for this purpose, highlighting the design, technical aspects, evaluation metrics, limitations, and ethical considerations.

The creation of an advanced document-based question-answering system using LangChain, along with large language models like OpenAI ChatGPT 3.5. LangChain, a specialized framework for language model-driven applications, forms the project's core, enabling an intelligent system for precise document-based inquiries. The project's focus is on generating accurate, context-aware answers using semantic search and state-of-the-art LLMs to create a cutting-edge Document QnA system.

2.0 AI Product Design

2.1 Overview

The system enables companies to upload their documents, including PDF and TXT files, and pose questions related to the content. By utilizing advanced technologies such as LangChain, OpenAI's LLMs, and Gradio, the system can provide precise and context-aware answers.

2.2 Target Audience

The system is designed for companies across industries that need to process and extract information from internal documents, contracts, reports, research, and more.

3.0 Technical Explanation

3.1 Document Loading and Processing

The system's initial step involves loading and preparing the documents. Using LangChain's `DirectoryLoader`, the system can effortlessly load a directory containing various document formats, adapting to diverse file types commonly found within corporate environments. Once the documents are loaded, the system utilizes a `RecursiveCharacterTextSplitter` to intelligently divide large documents into manageable chunks. This step is vital as it preserves the context within each chunk and ensures that the system can process the data efficiently. The flexibility to handle large and varied documents sets the stage for the subsequent stages of the process.

3.2 Embeddings and Query Handling

The core of the system's search capability lies in the transformation of user queries into meaningful embeddings and the subsequent similarity search within the document corpus. Utilizing OpenAI models, such as " text-embedding-ada-002," the system transforms queries into a form that can be used for semantic searches. This combination of embedding and similarity search ensures that the retrieved answers are not only contextually appropriate but also highly accurate.

3.3 Answer Generation and Question Answering Chain

Generating precise answers from the identified document sections is achieved using OpenAI's Large Language Model, specifically the "gpt-3.5-turbo." This model can understand the context of the query and the corresponding document sections to craft accurate and relevant answers. To ensure coherence and alignment with user expectations, the system employs a predefined question-answering chain. This

chain orchestrates the entire process, from embedding the query to retrieving the final answer, forming a seamless pipeline that delivers the desired information.

3.4 Interactive User Interface

An essential part of the system's design is its user interface, which has been crafted using Gradio. This interface allows users to upload documents and submit queries with ease, regardless of their technical expertise. By providing a user-friendly and interactive platform, the system enhances user engagement and ensures that accessing the information is as simple as asking a question. The integration of Gradio ensures that the complexity of the underlying technologies does not impede the user experience, offering a seamless interaction that makes the powerful capabilities of the system accessible to all.

Together, these technical components form a cohesive and robust system capable of navigating vast amounts of documentation to deliver precise and context-aware answers to user queries. Whether handling large volumes, varied formats, or complex queries, the system's design demonstrates a profound understanding of the challenges faced by businesses and offers an innovative solution leveraging cutting-edge technologies.

4.0 Evaluation Metrics

In our model, our primary focus is on the quality and correctness of the answers, rather than a traditional accuracy metric.

4.1 Quality and Response Time

Quality and response time are critical in a system that provides information extraction from documents. Evaluating the quality of the answers involves a close examination of the answers generated in response to queries, ensuring that they are contextually aligned with the documents and the query. The code leverages powerful models like "gpt-3.5-turbo" to ensure relevance and correctness. Rigorous testing with known answers, perhaps in a controlled environment, will enable a detailed evaluation of how precise the model is in different scenarios. In our tests, the model consistently provided correct answers within an acceptable time frame.

4.2 Efficiency

Efficiency encompasses both the speed of response and the optimal use of resources. The code reveals a well-designed pipeline. Benchmarks can be established for various document sizes and complexities to determine how well the system performs under varying loads. Understanding how the system scales with increasing data volumes and query complexity will be critical in assessing its suitability for different enterprise environments.

4.3 Usability

The Gradio interface indicates a strong focus on user interaction and accessibility. Usability testing would involve real-world users and various user personas to determine how well the system meets the needs of different types of users. Feedback on the interface's intuitiveness, error handling, and overall user satisfaction would offer valuable insights into potential improvements.

5.0 Limitations

While the code implementation includes various efficient handling methods tailored to process documents, there are certain limitations that must be taken into consideration.

5.1 Volume Handling

When dealing with an extremely large number of documents, the system's resources may become a bottleneck, leading to potential delays or failures in processing. Optimization techniques may be required to handle a substantial volume of data.

5.2 Tokenization

The utilized text embedding model, "ada," restricts processing to a maximum of 4096 tokens. Documents that exceed this token limit will either need to be truncated or split into smaller segments for processing. This constraint may affect the comprehensiveness of the analysis, particularly for complex or verbose documents, and it underscores the importance of considering volume and token limitations in the utilization of this code.

5.3 Language and Contextual sensitivity

The use of OpenAI models indicates that the system may have sensitivities to different languages or highly specialized terminologies. Depending on the training data and configuration of the underlying models, performance might vary significantly across different linguistic contexts or domains of expertise.

5.4 Complex Query Handling

While the code is designed to handle queries and generate answers using state-of-the-art models, there may still be challenges with highly complex or ambiguous queries. The system's ability to interpret and respond to such queries might need further tuning or additional logic to ensure that the answers are meaningful and accurate.

5.5 Dependency on External Services

The system relies on specific versions and services, including OpenAI models. Any changes or updates in these dependencies may affect system behavior and performance.

6.0 Ethical Considerations

Ethical considerations are paramount in the design and implementation of the AI-powered document-based question-answering system. The system's ability to process potentially sensitive information requires a meticulous approach to data privacy and security. This includes adhering to privacy regulations, obtaining proper user consent, and implementing robust encryption and secure storage methods. Bias and fairness are additional areas of concern; it's crucial to ensure that the system does not carry or perpetuate biases that could affect the quality or equity of its responses. The design must also prioritize accessibility and inclusivity, avoiding inadvertent favoritism towards certain groups of users.

6.1 Transparency and Accountability

Transparency and accountability further play a vital role in the system's ethical standing. This entails offering insights into the decision-making process and maintaining comprehensive logs for accountability. The potential environmental impact, particularly in terms of resource efficiency, warrants consideration, especially in the context of the significant computational resources required by large language models.

6.2 Compliance with Law and regulations

Compliance with applicable laws and regulations, including intellectual property laws and industry-specific standards, is a fundamental ethical obligation. This compliance ensures that the system's operation is in alignment with legal requirements. Finally, careful consideration of human interaction and dependency is essential. Balancing automation with human oversight, understanding potential over-reliance on AI, and providing adequate user training encapsulates a responsible approach to deployment.

7.0 Instructions to Run the Application

GitHub Link: https://github.com/goonizubeda/AIDI-2001---KAES/tree/main/AskMeDoc-Final_Project

1. Download Code from GitHub or Clone the repository from GitHub and navigate into the repository directory.
2. Set Up OpenAI API Key: Open the notebook and locate the line where the OpenAI API key is set. Replace "your_api_key" with your actual OpenAI API key.
3. Modify the path to the directory according to the environment where the code is being executed.
4. Execute the entire code.
5. Launch the Gradio Interface.
6. Upload Relevant Document: Once the Gradio interface is launched, you'll see an interface that allows you to upload a document and enter a question. Follow these steps:
    a. Click to choose the file or drag/drop the file to upload a relevant document (TXT or PDF format) from your local computer.
    b. Enter a question related to the uploaded document in the provided text box.
    c. Click the "Submit" button to initiate the Q&A process and get the answers as output.

By including these steps, users will be able to interact with the Gradio interface, upload the relevant document, enter questions, and get answers based on the content of the document.

***Please note: This works optimally when executed on Google Colab.**

8.0 Conclusion

The AI-powered Document-Based Question-Answering System offers an innovative solution for companies to efficiently extract vital information from their extensive documentation. By integrating state-of-the-art technologies and designing the system with the specific needs of businesses in mind, it opens doors to enhanced decision-making and insight-driven growth. Its capabilities are a testament to the potential of AI in transforming the way companies handle and derive value from their information assets.