Problem Statement:-The primary challenge addressed by this project is to develop an intelligent system capable of understanding and processing the content of PDF documents to provide accurate and contextually relevant answers to user queries. This involves leveraging advanced natural language processing (NLP) techniques to go beyond basic keyword search and understand the semantics of both the questions and the document content.

Objectives:-
1).**Develop a Robust Text Extraction System**:

- Implement methods to accurately extract text from PDF documents, including handling various formats and styles of text within the PD

2)Build an Interactive User Interface:

- Design a user-friendly web application using Streamlit, allowing users to upload PDF documents, input their queries, and receive answers in real-time.

3).Leverage NLP Techniques for Understanding:

- Utilize pre-trained word embeddings (such as GloVe) and other NLP tools to understand the semantic meaning of the text in PDFs and the user queries

Approach

Methodology

The PDF Answering AI project employs a systematic methodology leveraging advanced natural language processing (NLP) techniques and user-friendly web technology to address the problem of extracting relevant answers from PDF documents. The approach can be broken down into several key steps:

1. **Text Extraction from PDFs:**

   Library Used: PyPDF2

   Process:

   - Load the PDF document using PyPDF2.
   - Iterate through each page of the document.
   - Extract the text content from each page and concatenate it into a single string

2. **Interactive Web Interface with Streamlit:**

Create a user-friendly web interface for uploading PDFs, entering queries, and displaying answers.

- **Library Used**: Streamlit
- **Process**:
    - Build a Streamlit app that allows users to upload a PDF file.
    - Extract text from the uploaded PDF using the previously defined function.
    - Provide an input field for users to enter their query.
    - Display the most relevant sentence from the PDF as the answer to the query.

3. BERT Embeddings for Sentence Representation

Convert sentences to embeddings using BERT for semantic understanding.

- **Library Used**: Transformers (by Hugging Face)
- **Process**:
    - Load a pre-trained BERT model and tokenizer.
    - Then we trained the pre-trained model and fine tuned it based on question-answering approach.
    - Convert sentences from the PDF and user queries into BERT embeddings.

Failed Approaches

(1) We cannot take random dataset to train the bert model because we need very specific dataset to train this model based on context and their answers.
(2) Many time i got pipe broken [error no-32] which take long time for me to rectify.
(3) We can't directly use simple neural network because they will got so much difficulty to converge.

Analysis of Results

The PDF Answering AI project achieved its primary goal of creating an efficient and accurate system for retrieving relevant information from PDF documents based on user queries. The analysis of the results reveals several significant insights into the system's performance and its impact on information retrieval.

1. **Accuracy and Relevance of Answers**:

    - The use of BERT embeddings significantly improved the system's ability to understand the context and semantics of both the user queries and the PDF content. This led to highly accurate and relevant answers, particularly for complex and nuanced queries that traditional methods struggled with.
2. **Efficiency in Text Extraction**:
    - The text extraction process using PyPDF2 proved to be both accurate and efficient, handling a wide variety of PDF formats and layouts. This step was crucial in ensuring that the subsequent NLP processing had a reliable and clean text input.
3. **User Experience**:
    - The Streamlit-based web interface provided a simple and intuitive platform for users to interact with the system. Users could easily upload PDF documents, enter their queries, and receive answers in real-time, which greatly enhanced the overall user experience and accessibility.
4. **Handling of Synonyms and Variants**:
    - BERT's contextual embeddings allowed the system to recognize synonyms and different variants of expressions, which traditional keyword-based approaches like TF-IDF could not handle effectively. This capability ensured that the system could understand and respond accurately even when different words were used to express th

Conclusion

Effective Text Extraction:

- Using PyMuPDF, the system reliably extracted text from various PDF documents, handling different formats and styles.

Rich Semantic Understanding:

- Leveraging BERT embeddings allowed the system to understand the contextual meaning of words and sentences, significantly improving the accuracy of the answers provided.

User-Friendly Interface:

- The Streamlit-based web application provided a simple and intuitive interface for users to upload PDFs, enter queries, and receive answers, enhancing user experience and accessibility.

References:-

(1).BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding
https://arxiv.org/abs/1810.04805

(2).Streamlit: Turn Python Scripts into Beautiful ML Tools

- Streamlit Documentation. Available at: https://docs.streamlit.io/

(3) https://www.youtube.com/playlist?list=PLam9sigHPGwOBuH4_4fr-XvDbe5uneaf6

(4) https://thinkinfi.com/fine-tune-bert-for-question-answering/ for reference regarding fine tuning