Documentation for PDF-Based Question Answering System Using LangChain and HuggingFace

Overview

This system processes a PDF document to create a question-answering (QA) pipeline using LangChain and HuggingFace. The solution incorporates embeddings for text retrieval and a language model for generating accurate answers. It allows users to ask questions related to the PDF content, providing meaningful responses or appropriate fallback messages.

Workflow

1. PDF Processing:
    - Extracts and preprocesses text from a PDF file.
    - Converts the text into chunks for efficient retrieval.
2. Text Vectorization:
    - Converts chunks into embeddings using the sentence-transformers/all-MiniLM-L6-v2 model.
3. Language Model Integration:
    - Uses the Llama-2-7b-chat-hf model for text generation, leveraging HuggingFace's pipeline and BitsAndBytes quantization for memory efficiency.
4. Retrieval-Based QA Pipeline:
    - Combines embeddings and the language model to create a QA system capable of retrieving and generating contextually relevant answers.
5. Custom QA Function:
    - Extracts answers from the pipeline output and handles out-of-scope queries gracefully.

Code Breakdown

1. Dependencies

The solution relies on the following libraries:

- LangChain: For text processing, vectorization, and retrieval.
- HuggingFace Transformers: For the language model and text generation.
- PyPDFPlumber: To extract text from PDF files.
- Regex (re): For text preprocessing and answer extraction.
- Torch: For GPU-based operations.

2. PDF Text Extraction

The extract_text_from_pdf function uses pdfplumber to extract text from the PDF file:

3. Text Preprocessing

Prepares text for downstream processing:

- Removes special characters.
- Converts text to lowercase for normalization.

4. Text Splitting

Divides the preprocessed text into manageable chunks for vectorization using LangChain's CharacterTextSplitter:

5. Vectorization

Uses the sentence-transformers/all-MiniLM-L6-v2 model to generate embeddings, enabling similarity-based text retrieval:

6. Language Model Setup

The Llama-2-7b-chat-hf model is loaded with 4-bit quantization for efficient memory usage:

A HuggingFace pipeline is created for text generation:

7. Retrieval QA Pipeline

Combines vectorized text chunks and the language model into a retrieval-based QA pipeline:

8. Question Answering Function

Handles user queries and extracts meaningful answers from the pipeline:

Usage

1. Input: Provide a PDF file (input.pdf) containing relevant information.
2. Query: Ask questions related to the PDF content using get_answer().
3. Output: Receive contextually relevant answers or fallback responses for out-of-scope questions.
4.

Further Enhancements

- Deployment: Package the solution with Gradio or Streamlit for an interactive UI.
- Multi-File Support: Extend the pipeline to handle multiple files.
- Advanced QA: Incorporate additional prompts or models for more nuanced answers.