Project Report: Intelligent Document Processing and Retrieval System

Introduction

This project focuses on creating an intelligent document processing and retrieval system capable of handling multiple file formats, such as PDFs, CSVs, and JSON files. The system processes uploaded files, extracts text, generates embeddings using a Hugging Face model (mistralai/Mistral-7B-Instruct-v0.3), and stores them in Pinecone for efficient semantic search and retrieval. A question-answering (QA) feature allows users to query the processed content and receive accurate responses.

Objectives

1. Enable processing of multiple file formats (PDF, CSV, JSON).
2. Leverage advanced embedding techniques for semantic text representation.
3. Implement efficient vector storage for scalable search and retrieval.
4. Provide a user-friendly interface for question-answering over processed content.

System Architecture

The system is implemented using FastAPI, and the main components include:

- File Upload and Text Extraction: Handles multiple file types and extracts text for further processing.
- Text Embeddings: Generates dense vector representations of text using Hugging Face's mistralai/Mistral-7B-Instruct-v0.3.
- Vector Storage: Pinecone is used for storing and managing vector embeddings, enabling fast similarity searches.
- Question Answering (QA): Uses LangChain's RetrievalQA chain for intelligent responses based on uploaded content.

Key Features

1. File Handling:
   - PDF: Extracts text from pages and splits it into manageable chunks.
   - CSV: Processes the text column for embeddings.
   - JSON: Extracts values from the text key of JSON objects.
2. Semantic Search: Efficient semantic similarity search through Pinecone.
3. Question Answering: Provides precise answers to queries based on processed document content.

Technologies Used

- FastAPI: For building the API.
- LangChain: For text splitting, embeddings, and QA chains.
- Hugging Face: For generating embeddings with the mistralai/Mistral-7B-Instruct-v0.3 model.
- Pinecone: A scalable vector database for efficient storage and retrieval of embeddings.
- PyPDF2: For PDF text extraction.
- Pandas: For CSV data handling.

Why Pinecone Over FAISS

While FAISS is a robust vector database, Pinecone was chosen for this project due to several reasons:

1. Ease of Integration: Pinecone provides a serverless, fully managed environment with minimal setup and maintenance.
2. Scalability: Pinecone scales seamlessly with growing data volumes, which is essential for large datasets.
3. Advanced Features: Pinecone offers cloud-native capabilities, such as automatic sharding and distributed querying, that FAISS lacks in its local setup.
4. Error Resolution:

    - FAISS with SentenceTransformer: The integration of FAISS and SentenceTransformer resulted in the error: "Error processing file: too many values to unpack (expected 2)". This occurred due to a mismatch in the expected input/output of FAISS's from_embeddings method and the sentence transformer-generated embeddings.
    - Despite attempts to resolve the issue, the workflow consistently broke when integrating FAISS for high-dimensional vector search.
5. OpenAI Embeddings Compatibility: Using OpenAI embeddings with FAISS avoided the unpacking error, but the project aimed to use open-source Hugging Face embeddings for cost-efficiency and flexibility.

Challenges and Resolutions

1. SentenceTransformer with FAISS

- Issue: The integration led to compatibility errors during embedding storage. FAISS expected specific formats for embeddings and metadata, which weren't directly aligned with the SentenceTransformer-generated embeddings.
- Resolution: Replacing FAISS with Pinecone solved the compatibility issues and provided additional benefits like cloud-native scalability.

2. Multi-format File Support

- Issue: Handling diverse file formats required implementing distinct logic for each.
- Resolution: Modularized text extraction for each format (PDF, CSV, JSON), making the system extensible.

Results

The project successfully achieves:

1. Text processing for PDFs, CSVs, and JSONs.
2. Semantic embeddings using Hugging Face's Mistral model.
3. Scalable vector storage with Pinecone.
4. Robust question-answering based on processed document content.

Conclusion

This project demonstrates the integration of state-of-the-art language models with modern vector databases to build an intelligent document retrieval and QA system. By leveraging Pinecone's advanced capabilities and Hugging Face's open-source embeddings, the system is efficient, scalable, and cost-effective.