Legal Document Analyzer: Project Documentation

Introduction

This documentation outlines the complete workflow and learnings from the Legal Document Analyzer project. The goal of the project was to automate and simplify the analysis of Indian legal case files using modern Natural Language Processing (NLP) techniques. The project includes preprocessing, extractive and abstractive summarization, Named Entity Recognition (NER), and a legal Q&A system—all designed with a focus on handling Indian legal texts.

---

1. Dataset and Preprocessing

I started the project with a large dataset of **11,970 Indian legal case files**. The preprocessing involved multiple steps to ensure the data quality and retention of contextual meaning:

- **Text Cleaning**: Removed unwanted characters, formatting issues, and non-legal terms.
- **Stopword Removal**: Carefully eliminated stopwords while maintaining the semantic context.
- **Embedding**: Used the **InLegalBERT** model to embed the cleaned case documents.
- **Batching**: Due to the size of the dataset, I divided it into **12 batches**.
- **Storage**: Stored the final vector embeddings in **ChromaDB** under the collection name **legal_docs**.

---

2. Extractive Summarization

For extractive summarization, the workflow included real-time processing of uploaded PDFs:

- **PDF Embedding**: The uploaded document was embedded and tokenized into sentences on the fly.
- **Cosine Similarity**: Compared sentence embeddings with document embeddings to select the **top 10 most relevant sentences**.
- **Similar Case Retrieval**: Using a RAG (Retrieval-Augmented Generation) approach:
    - Retrieved related cases from the legal_docs collection.
    - Displayed **metadata**, **summary sentences**, and the **Indian Kanoon links** for those cases.
    - Extracted and showed **relative sentences** from the retrieved documents.

---

3. Abstractive Summarization

Initially, I tried several large-scale models specifically trained on legal texts for abstractive summarization. However, hardware limitations required a shift in strategy:

- **Model Used**: Switched to **sshleifer/distilbart-cnn-12-6**, a lightweight yet accurate model.
- **Chunking Strategy**: As legal documents are lengthy:
    - I implemented **chunking with overlap** to preserve context.
    - Each chunk was individually summarized.
    - Final summary was constructed by combining and refining chunk-level summaries.

---

4. Named Entity Recognition (NER)

To identify legal entities, roles, organizations, and other important details:

- **Primary Model**: Used **MHGanainy/roberta-base-legal-multi-downstream-indian-ner** trained specifically on Indian legal text.
- **Comparative Benchmarking**: Also used **SpaCy's general NER model** for comparison.
- **Outcome**: The specialized legal NER model performed significantly better in identifying Indian legal entities accurately.

---

5. Legal Question Answering (Q&A)

The goal here was to build a robust Q&A system that understands Indian legal context:

- **Dataset**: Identified and used an **Indian Legal QnA dataset**.
- **Fine-Tuning Attempt**: Attempted to convert the data into **SQuAD format** for fine-tuning.
    - Due to hardware constraints, I was unable to complete the fine-tuning.

RAG-Based Q&A Implementation:

- **Embedding**: Embedded all questions and answers.
- **Storage**: Stored in **ChromaDB** under the collection **legal_qna**.
- **Retrieval Pipeline**:
    - **Primary Search**: Retrieved top 5 similar answers based on cosine similarity to user query.
    - **Fallback Mechanism**: If primary retrieval failed to meet a threshold, searched by **similar questions** and retrieved the corresponding answers.
- **LLM Integration**: Used **Gemini 2.5 API** for generating final answers due to its:
    - Robust performance.
    - Strong training on Indian Constitution and law.

Conclusion and Learnings

This project was a deep dive into the challenges of processing and analyzing Indian legal texts using NLP. Some of the key takeaways include:

- Effective use of **InLegalBERT** and **ChromaDB** for storage and retrieval.
- The importance of **chunking and overlap** in summarization.
- RAG methods as viable alternatives to **resource-intensive fine-tuning**.
- Domain-specific models significantly outperform general-purpose ones.
- **Gemini 2.5 API** proved to be highly effective for legal Q&A tasks.

The final Legal Document Analyzer provides a powerful pipeline to process, summarize, understand, and interact with legal case files from the Indian judicial system.
