Mr.PolicyHelperAI – LangChain RAG System for Insurance Policy Understanding

Abstract

This report presents Mr.PolicyHelperAI, a Retrieval-Augmented Generation (RAG) system built using LangChain to semantically process and query insurance policy documents. The system ingests unstructured PDF data, extracts and chunks content, embeds it using HuggingFace models, and stores it in a FAISS vector database. It leverages semantic search and GPT-based language models to generate accurate, context-aware responses with citations. Designed for scalability and modularity, Mr.PolicyHelperAI demonstrates the potential of combining document intelligence with generative AI for enterprise-grade applications in insurance and compliance.

Introduction

Insurance policy documents are often lengthy, complex, and difficult to interpret. Extracting relevant information from such documents requires not only keyword matching but also semantic understanding. Traditional search systems fall short in capturing the nuanced relationships between policy terms, exclusions, and benefits.

To address this challenge, Mr.PolicyHelperAI was developed as a LangChain-powered RAG system that enables users to ask natural language questions and receive precise, citation-backed answers from a corpus of insurance PDFs. The system integrates document parsing, semantic embeddings, vector search, reranking, and large language models to deliver an end-to-end intelligent retrieval experience. This report details the architecture, implementation, and evaluation of Mr.PolicyHelperAI, highlighting its effectiveness in transforming static insurance documents into interactive knowledge sources.

Project Overview

This project, titled **Mr.PolicyHelperAI**, aims to build a Retrieval-Augmented Generation (RAG) system using LangChain to semantically search and answer queries from **insurance policy documents**. The system processes **7 PDF files** from HDFC Life, extracts structured and unstructured data, embeds it using HuggingFace models, stores it in FAISS, and uses GPT-based LLMs to generate accurate, citation-backed responses.

System Architecture

The architecture consists of the following layers:

- **Data Ingestion Layer**: PDF parsing using pdfplumber.
- **Preprocessing Layer**: Text and table extraction, chunking.
- **Embedding Layer**: HuggingFace all-MiniLM-L6-v2 model.
- **Vector Store Layer**: FAISS for similarity search.
- **Retrieval Layer**: LangChain retriever with reranking.
- **LLM Layer**: ChatOpenAI for response generation.
- **Memory Layer**: LangChain's ConversationBufferMemory.

Modules and Techniques Used

PDF Parsing and Text Extraction

- **Library**: pdfplumber
- **Functionality**:
    - Extracts text and tables from each page.
    - Differentiates between table and non-table content using bounding box logic.
    - Clusters content to preserve layout chronology.

Chunking Strategy

- **Chunk Size**: 500 words
- **Method**: Fixed-size chunking per page using a custom function.
- **Metadata**: Each chunk retains page number, document name, and chunk index.

Embedding Generation

- **Model**: all-MiniLM-L6-v2 from HuggingFace
- **Tool**: HuggingFaceEmbeddings via LangChain

- **Output**: Dense vector representations of each chunk.

Vector Store

- **Tool**: FAISS
- **Functionality**:
    - Stores embeddings with metadata.
    - Enables fast similarity search.

Semantic Search

- **Retriever**: LangChain's retriever interface
- **Query Example**: “What is the policy on eye issues?”
- **Output**: Top-k semantically similar chunks with metadata.

Reranking and Compression

- **Tools**: ContextualCompressionRetriever, EmbeddingsFilter
- **Purpose**: Improves relevance by reranking compressed retrieved chunks.

Memory and Context

- **Tool**: ConversationBufferMemory
- **Purpose**: Maintains chat history for multi-turn interactions.

LLM Response Generation

- **Model**: ChatOpenAI (GPT-3.5-Turbo)
- **Prompt Template**: Custom prompt with citation formatting.
- **Output**: Direct answers with references to document chunks.

Data Description

Source Documents

- 7 PDF files from HDFC Life:

- HDFC-Life-Easy-Health-101N110V03-Policy-Bond-Single-Pay.pdf
- HDFC-Life-Group-Poorna-Suraksha-101N137V02-Policy-Document.pdf
- HDFC-Life-Group-Term-Life-Policy.pdf
- HDFC-Life-Sampoorna-Jeevan-101N158V04-Policy-Document (1).pdf
- HDFC-Life-Sanchay-Plus-Life-Long-Income-Option-101N134V19-Policy-Document.pdf
- HDFC-Life-Smart-Pension-Plan-Policy-Document-Online.pdf
- HDFC-Surgicare-Plan-101N043V01.pdf

Content Types

- Policy schedules
- Definitions
- Benefits
- Exclusions
- Terms and conditions

Data Volume

- 217 pages processed from 7 PDFs
- Around 1220 chunks generated

Implementation Details

Environment

- LangChain
- HuggingFace Transformers
- FAISS
- LangChain ChatOpenAI

Performance Optimizations

- Table-aware parsing
- Metadata tagging
- Chunk-level reranking
- Caching via LangChain memory

Example Query Flow

1. User asks: “What is the policy on eye issues?”
2. Retriever fetches top chunks.
3. Reranker scores relevance.
4. LLM generates response:

   “Blindness is covered if corrected visual acuity is 3/60 or less in both eyes or field of vision is less than 10 degrees. Diagnosis must be confirmed and not correctable by aids or surgery. [HDFC-Life-Easy-Health, Page 7]”

Evaluation and Observations

The Mr.PolicyHelperAI system was evaluated based on its ability to accurately retrieve and respond to user queries from a corpus of insurance policy documents. The evaluation focused on semantic relevance, citation accuracy, system responsiveness, and overall user experience.

Evaluation Criteria

- Semantic Accuracy: How well the retrieved chunks match the intent of the user query.
- Citation Precision: Whether the response includes correct document references (name, page number).
- Response Quality: Clarity, completeness, and correctness of the generated answers.
- Retrieval Latency: Time taken to fetch and rerank relevant chunks.
- Scalability: Performance with increasing number of documents and queries.

Observations

- The system consistently retrieved highly relevant chunks for queries involving definitions, exclusions, and benefits.
- Responses generated by the LLM were contextually accurate and included chunk-level citations, improving traceability.
- The use of FAISS enabled fast similarity search even with over 1200 chunks.
- Reranking and Compression via LangChain's compression retriever scoring improved precision, especially for queries with overlapping semantics.
- The system handled multi-turn conversations effectively using LangChain's memory module.
- Table-heavy pages were parsed successfully, but semantic interpretation of tabular data remains limited.
- The system architecture proved modular and extensible, allowing easy integration of additional documents or components.

Limitations and Challenges Faced

While Mr.PolicyHelperAI demonstrates strong performance in semantic retrieval and response generation, several limitations and challenges were encountered during development:

Limitations

- Fixed-Size Chunking: The current chunking strategy uses a fixed word count, which may split semantically related content across chunks, reducing retrieval coherence.
- Table Interpretation: Although tables are extracted using pdfplumber, they are stored as raw JSON strings, limiting their interpretability during semantic search.
- No Fine-Tuning: The system uses pre-trained embeddings and LLMs without domain-specific fine-tuning, which may affect precision in highly specialized queries.
- Citation Granularity: Citations are based on chunk-level metadata, which may not always point to the exact sentence or clause relevant to the answer.

- Scalability Constraints: As the number of documents grows, FAISS indexing and retrieval latency may increase without sharding or distributed search.

Challenges Faced

- PDF Layout Variability: Insurance documents vary in structure, making consistent extraction of text and tables difficult.
- Semantic Overlap: Similar policy terms across documents led to overlapping embeddings, requiring careful reranking to avoid redundancy.
- Memory Management: Maintaining conversational context across multiple queries required careful use of LangChain's memory modules.
- Prompt Engineering: Designing prompts that balance answer quality with citation accuracy involved iterative tuning and testing.
- Evaluation: Measuring the quality of semantic answers and their alignment with source documents required manual validation.

Recommendations and Future Enhancements

Based on the current implementation and evaluation of Mr.PolicyHelperAI, the following recommendations are proposed to improve system performance, scalability, and user experience:

Recommendations

- Improve Chunking Strategy: Replace fixed-size chunking with semantic-aware chunking using LangChain's RecursiveCharacterTextSplitter or SemanticChunker to preserve context boundaries.
- Citation Accuracy: Enhance citation formatting by including document name, page number, and chunk index in a standardized reference format.
- Query Expansion: Integrate query expansion techniques to improve recall for ambiguous or domain-specific queries.
- User Interface: Develop a lightweight front-end interface for non-technical users to interact with the system via chat or search bar.

Future Enhancements

- Agent Integration: Extend the system with LangChain Agents to support multi-step reasoning, policy comparison, and summarization tasks.
- Fine-Tuned Reranker: Replace the default retriever scoring with a fine-tuned cross-encoder reranker (e.g., Cohere or OpenAI embeddings) for better relevance.
- Multi-Modal Support: Enable image and diagram parsing from PDFs to support richer document types.
- Feedback Loop: Implement user feedback collection to continuously improve retrieval quality and LLM responses.
- Enterprise Deployment: Containerize the system using Docker and deploy via FastAPI or Streamlit for scalable enterprise use.

Conclusion

Mr.PolicyHelperAI demonstrates a robust LangChain-based RAG system tailored for insurance policy understanding. It combines PDF parsing, semantic embeddings, vector search, reranking, and LLM-based response generation to deliver accurate, contextual answers with traceable citations. The system is modular, scalable, and ready for deployment in enterprise document intelligence workflows.