Semantic Spotter – Project Report

1. Objective

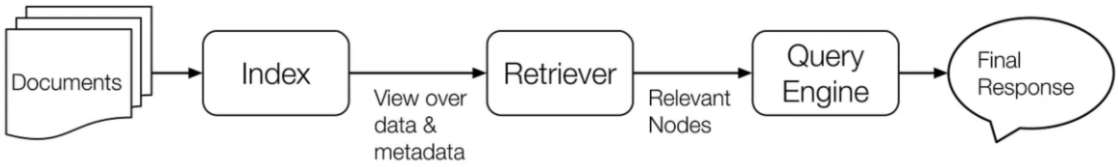
The objective of the Semantic Spotter project is to build a Generative Search (RAG) System for retrieving and answering queries from insurance documents using LlamaIndex. The system is designed to understand semantic context and generate relevant answers with source citations and confidence scores.

2. Data Source

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-Sanchay-Plus-Life-Long-Income-Option-101N134V19-Policy-Document.pdf
HDFC-Life-Smart-Pension-Plan-Policy-Document-Online.pdf
HDFC-Surgicare-Plan-101N043V01.pdf
uber_2021.pdf

3. System Design

The system consists of the following key components:



1. Document Loader: Reads and parses the PDF policy documents.
2. Indexer: Uses OpenAI embeddings to convert documents into vector representations.
3. Query Engine: Uses LlamaIndex's RefineQueryEngine with retriever and reranker.
4. Reranker: Cross-encoder-based reranker (MiniLM model) ranks retrieved chunks.
5. LLM Response Synthesizer: Generates final response using top-k relevant chunks.
6. Cache: Uses diskcache to store previously seen responses for performance.
7. Evaluator: GPT-4 based evaluators assess faithfulness, relevancy, and correctness.

4. Solution Strategy

1. Build a solution which should solve the following requirements:
2. Users would get responses from insurance policy knowledge base.

3. If user want to perform a query system must be able to response to query accurately.
4. If they want to refer to the original page from which the bot is responding, the bot should provide a citation as well.

5. Implementation

The solution was implemented in Python using Jupyter Notebook. Key technologies include:

- LlamaIndex (SimpleDirectoryReader, VectorStoreIndex, RefineQueryEngine)
- SentenceTransformer (cross-encoder reranker)
- OpenAI embeddings & GPT-4 for evaluations
- diskcache for query response caching

6. Why LlamaIndex ?

LlamaIndex is an innovative data framework specially designed to support LLM-based RAG framework application development. It offers an advanced framework that empowers developers to integrate diverse data sources with large language models. LlamaIndex includes a variety of file formats, such as PDFs and PowerPoints, as well as applications like Notion and Slack and even databases like Postgres and MongoDB.

The framework brings an array of connectors that assist in data ingestion, facilitating a seamless interaction with LLMs. Moreover, LlamaIndex boasts an efficient data retrieval and query interface.

LlamaIndex enables developers to input any LLM prompt and, in return, receive an output that is both context-rich and knowledge-augmementation.

Key Feature of LlamaIndex:
● Data connectors allow ingestion from various data sources and formats.
● It can synthesize data from multiple documents or heterogeneous data sources.
● It provides numerous integrations with vector stores, ChatGPT plugins, tracing tools, LangChain, and more.

7. Challenges Faced

- Faced compatibility issues with RAGAS evaluation module.
- gptcache did not integrate well; switched to diskcache.
- Version conflicts between dependencies (e.g., protobuf, requests).
- Some prompts resulted in overly long or verbose answers.

8. Lessons Learned

- Effective prompt design greatly affects the quality of LLM answers.
- Caching mechanisms are crucial for user experience.
- Fine-tuned reranking models significantly improve relevance.
- Robust evaluation using LLMs can validate generative system accuracy.

9. Tools & Libraries Used

- Python
- Jupyter Notebook
- LlamaIndex
- OpenAI API
- SentenceTransformers
- diskcache
- GPT-4 for Evaluation

10. System Flow Overview

The architecture involves document ingestion, vectorization, retrieval, reranking, response synthesis, and evaluation. Below is a simplified flow (diagram to be added if required):

User Query → Cache Check → Retriever → Reranker → LLM → Response + Scores + Source

11. Conclusion

The Semantic Spotter system successfully demonstrates the application of RAG systems in the insurance domain. Using LlamaIndex and GPT-4 based evaluation, the project achieves reliable, traceable, and semantically relevant responses.