Laboratory File

on

AGENTIC AI

School of Engineering and Technology

Department of Computer Science and Engineering

Subject Code – **CSCR 3215**

SUBMITTED BY:

Name: Mohd Suhail Khan
System ID: 2023418892

SUBMITTED TO:

Mr. Ayush Singh

Sharda University

Greater Noida, Uttar Pradesh

Lab 02: Chunking Method

Project Documentation: Multi-Level Text Splitting for Multimodal Applications

Objective:
The objective of this project is to design, implement, and evaluate multiple text splitting (chunking) strategies to efficiently process large-scale textual and multimodal data. The project aims to enhance **context retention**, **semantic coherence**, and **information retrieval accuracy**, which are critical for AI systems such as chatbots, Retrieval-Augmented Generation (RAG), and document-based question answering systems.

Methodology:

1. Input Data Collection:
The system processes large and diverse data formats, including PDF documents, markdown files, source code files, and structured/unstructured text. This ensures the chunking strategies are tested on real-world and heterogeneous datasets.

2.Preprocessing:
Text preprocessing involves removing noise, normalizing formatting, handling special characters, and extracting meaningful content. This step ensures consistent input quality and reduces irrelevant data before chunking.

3. Chunking Strategies:
Five levels of chunking are implemented:

- **Character-based chunking:** Fixed-size text splits.
- **Recursive chunking:** Hierarchical splitting using logical separators.
- **Document-specific chunking:** Structure-aware splitting using headers, sections, or code blocks.
- **Semantic chunking:** Meaning-based chunks using embeddings.
- **Agent-based chunking:** Dynamic chunking guided by AI agents based on intent and context.

4. Tools and Frameworks:
Python is used as the primary programming language. LangChain is utilized for text loaders, chunking pipelines, and retrieval workflows. Embedding models convert text chunks into vector representations for semantic search and similarity matching.

Working:

- The input document is first analyzed to identify its structure and content type.

- A selected chunking strategy is applied to divide the document into logically meaningful segments.

- Each chunk is stored along with metadata such as source, position, and chunk type.
- Vector embeddings are generated and stored in a vector database.
- During inference, relevant chunks are retrieved based on semantic similarity to the user query, improving response accuracy.

Outcomes:

- Large documents are processed efficiently without exceeding context limitations.
- Contextual relevance of AI-generated responses is significantly improved.
- Semantic and agent-based chunking shows higher retrieval precision compared to basic methods.
- The evaluation provides insights into when and where each chunking strategy performs best.

Conclusion:

This project demonstrates that effective text splitting is a critical component of intelligent AI systems. Advanced chunking techniques—particularly semantic and agent-based methods—offer superior performance in terms of context understanding and retrieval accuracy. These approaches are highly suitable for scalable, data-intensive, and real-world AI applications.