

Laboratory file
on
AGENTIC AI



School of Engineering and Technology
Department of Computer Science and Engineering

Subject code – CSCR 3215

SUBMITTED BY:

Name : Rachna Singh
System ID: 2023371832

SUBMITTED TO:

Mr. Ayush Singh

Sharda University
Greater Noida, Uttar Pradesh

Lab 02: Chunking Methods

5 Levels Of Text Splitting MultiModal

Objective:

Efficient handling of long text for language models by splitting it into smaller, meaningful chunks. This improves retrieval, memory, and overall model performance.

Methodology:

Level 1: Character Splitting – Fixed-size chunks; simple but ignores structure.

Level 2: Recursive Splitting – Uses paragraph, line, and space separators; preserves logical boundaries.

Level 3: Document-Specific Splitting – Tailored for Markdown, Python, JS, PDFs; uses domain-specific separators.

Level 4: Semantic Chunking – Groups text by meaning using embeddings; keeps semantically related sentences together.

Level 5: Agentic Chunking – LLM-driven intelligent splitting; respects topic boundaries and context.

Tools & Libraries: LangChain Text Splitters, langchain_experimental + OpenAIEmbeddings, ChatOpenAI, unstructured (PDFs), evaluation with LangChain Evals / Llama Index Evals / RAGAS.

Working:

- 1. Input Text:** Text (or document) is loaded.
- 2. Split Strategy:** A splitting method is chosen based on file type, structure, or semantics.
- 3. Chunking:** Text is divided into chunks, optionally with overlap or semantic grouping.
- 4. Metadata:** Each chunk can store additional info (source, chunk number) for retrieval.
- 5. Usage:** Chunks are fed to LLMs for retrieval, summarization, or knowledge integration.

Outcomes:

1. Text splitting improves **LLM performance** for long documents.
2. Semantic and agentic chunking maintain context and reduce loss of meaning.
3. Custom chunking strategies can handle any file type: text, code, markdown, PDF, and multi-modal content.

Conclusion:

Effective text splitting is essential for handling long documents in LLM applications. By using strategies from simple character splitting to advanced agentic chunking, we can preserve context, maintain semantic meaning, and improve model performance. Tailored splitting methods allow working with diverse data types like code, Markdown, PDFs, and multi-modal content, making text processing more efficient and reliable for retrieval, summarization, and downstream AI tasks.