Comparative Analysis of Chunking Strategies for a Physics Textbook-Based RAG LLM Chatbot

Saif Mohammed 2121913042
Humayra Rahman Nipa 2121128042

September 2024

1 Introduction

In developing a physics textbook-based chatbot, the choice of chunking strategy plays a crucial role in ensuring the chatbot can accurately retrieve information and provide contextually relevant answers. Several chunking strategies have been considered, each with its advantages and trade-offs. This document presents a comparative analysis of five chunking strategies applied to a physics textbook in PDF format, focusing on their efficiency, contextual accuracy, and overall performance in a chatbot environment.

2 Strategies Considered

- Character Text Splitter
- Split by Token
- Recursive Chunking
- Markdown Header Chunking
- Semantic Chunking

3 Character Text Splitter

3.1 Overview

The **Character Text Splitter** divides the document based on a specific number of characters, ensuring that the chatbot doesn't process overly large chunks of text. This method is simple and can be easily applied to any document.

3.2 Pros

- **Speed**: Fast to implement and process.
- **Uniformity**: Ensures chunks of relatively uniform size.

3.3 Cons

- **Context Loss**: Can cut off sections of a sentence or paragraph, leading to a loss of context.
- **Lack of Structure**: Does not respect the natural structure of the document.

3.4 Use Case

This strategy is suitable for smaller, simpler documents where context loss is minimal but not ideal for textbooks.

4 Split by Token

4.1 Overview

Split by Token divides the text based on the number of tokens (words or subwords), rather than characters, ensuring a more granular and controlled splitting.

4.2 Pros

- **Fine Control**: Better control over chunk size, aligning better with language models.
- **Performance**: Optimizes model performance based on token limits.

4.3 Cons

- **Context Fragmentation**: Risk of breaking context mid-sentence or paragraph.
- **Complexity**: Additional layers may be required to avoid important section splits.

4.4 Use Case

Useful for models with token limits, but risk of context loss makes it less effective for textbooks.

5 Recursive Chunking

5.1 Overview

Recursive Chunking breaks the document down into larger sections based on natural breaks, then recursively splits these into smaller chunks.

5.2 Pros

- **Preserves Structure**: Maintains the logical flow of content.
- **Context Retention**: Retains more meaningful information by minimizing context loss.

5.3 Cons

- **Slower Processing**: More complex and slower than simpler methods.

- **Variable Chunk Size**: May lead to inefficiencies in systems requiring uniform input sizes.

5.4 Use Case

Works well for structured content like textbooks but may require more computational resources.

6 Markdown Header Chunking

6.1 Overview

Markdown Header Chunking splits the document based on headings and subheadings, ideal for textbooks with a clear hierarchical structure.

6.2 Pros

- **Structure Preservation**: Respects the document's natural structure.
- **Contextual Integrity**: Ensures each chunk represents a logical section, retaining context.

6.3 Cons

- **Not Universally Applicable**: Only works well with documents having proper headers.
- **Chunk Size Variability**: Can result in inconsistencies in chunk size.

6.4 Use Case

Effective for textbooks with clear sections and headings, making it ideal for structured educational content.

7 Semantic Chunking

7.1 Overview

Semantic Chunking splits text based on meaning, using machine learning models to create coherent and contextually rich chunks of information.

7.2 Pros

- **Contextual Relevance**: Preserves meaning and context, ensuring accurate responses.
- **Ideal for Q&A Systems**: Works well for systems that need precise and meaningful responses.

7.3 Cons

- **Higher Computational Cost**: Requires more computational resources.
- **Slower Chunking Process**: Can be slower, especially for large documents.

7.4 Use Case

Best for systems prioritizing context and accuracy, such as educational chatbots for physics textbooks.

8 Summary of Comparative Analysis

To summarize, the following table presents a comparative analysis of the discussed chunking strategies.

Strategy	Pros	Cons	Best Use Case
Character Text Splitter	Fast and uniform chunk size	High context loss, unsuitable for structured documents	Simple documents with minimal context needs
Split by Token	Fine control, optimized for token limits	Context fragmentation, requires token count management	Token-limited systems
Recursive Chunking	Preserves structure, good context retention	Slower processing, variable chunk size	Structured documents, content-heavy systems
MarkdownHeader Chunking	Respects structure, ideal for textbooks	Only works with proper headers, chunk size variability	Textbooks and hierarchical documents
Semantic Chunking	High context retention, meaningful segmentation	Slower and computationally expensive	Q&A chatbots, systems needing accurate and coherent chunks

Table 1: Comparative Analysis of Chunking Strategies

9 Conclusion

The choice of chunking strategy depends on the specific needs of the chatbot system. For a physics textbook-based chatbot, **Semantic Chunking** and **Markdown Header Chunking** are the most suitable options due to their ability to retain context and respect the document's structure. Combining both strategies can capture the textbook's structure and provide meaningful, contextually relevant responses, ensuring an accurate and efficient chatbot experience.