Step 2: Chunk Text for Embedding

The text needs to be broken down into smaller, manageable chunks. We use RecursiveCharacterTextSplitter to split each page's text into overlapping chunks.

Using the RecursiveCharacterTextSplitter to break text into smaller, manageable chunks with overlapping sections is important for several reasons, especially when dealing with natural language processing (NLP) tasks, large documents, or continuous text analysis. Here's why it's beneficial:

1. Improves Context Retention

- When text is split into overlapping chunks, each chunk retains some of the previous and following content. This helps preserve context, which is especially crucial for algorithms that rely on surrounding information, like NLP models.
- Overlapping text ensures that important details spanning across chunk boundaries aren't lost, which is critical for maintaining the coherence of the information.

2. Enhances Accuracy in NLP Tasks

- Many NLP models (such as question-answering systems or sentiment analysis models) can perform better when provided with complete context. Overlapping chunks help these models access more relevant information, leading to more accurate and reliable results.

3. Manages Memory and Processing Efficiency

- Breaking down large texts into smaller parts helps manage memory usage and processing time, making it feasible to handle extensive documents without overwhelming the system.
- Smaller chunks allow for parallel processing, improving the efficiency of tasks like keyword extraction, summarization, or entity recognition on large texts.

4. Facilitates Chunked Data Storage and Retrieval

- Overlapping chunks can be stored and retrieved more flexibly, making it easier to reconstruct portions of the text for further processing, such as when analyzing text in a sliding window approach for time series data or contextual searches.

5. Supports Recursive Splitting for Optimal Size

- RecursiveCharacterTextSplitter can recursively split text until the desired chunk size is achieved, allowing you to tailor chunk sizes according to model input limits or memory constraints while keeping context intact.