Understanding Retrieval-Augmented Generation (RAG) with Chunking

1. Introduction to Retrieval-Augmented Generation (RAG)

Retrieval-Augmented Generation (RAG) is a framework that combines pre-trained language models with external knowledge bases to enhance the accuracy and relevance of generated text. Instead of relying solely on the model's internal knowledge, RAG retrieves relevant chunks of information from external sources, such as document collections, to assist in generation.

2. What is Chunking?

Chunking refers to the process of breaking down large documents or text data into smaller, manageable pieces (or chunks). Each chunk is designed to be semantically coherent and easily retrievable. Chunking is a critical step in RAG, as it ensures that only the most relevant parts of the document are retrieved for answering a query.

3. Step-by-step Chunking for PDF Documents

Step 1: Extracting Text and Metadata

Use tools like PyPDF2, Marker, or Docling to extract text and metadata from PDF documents. Metadata such as titles, authors, and section headers can assist in chunk labeling.

Step 2: Chunking the Text

Split the extracted text into smaller chunks. This can be done using:

- Fixed-length splitting (e.g., every 500 words).
- Semantic chunking (split based on paragraphs, sentences, or topics). Tools like spaCy or NLTK can assist in semantic chunking.

Step 3: Storing Chunks

Store the chunks in a vector database like Chroma, Pinecone, or Weaviate. These databases enable efficient similarity searches for retrieving relevant chunks during inference.

Step 4: Retrieving Chunks for RAG

When a query is made, retrieve the most relevant chunks from the database using vector search.

These chunks are then fed into the language model to generate responses.