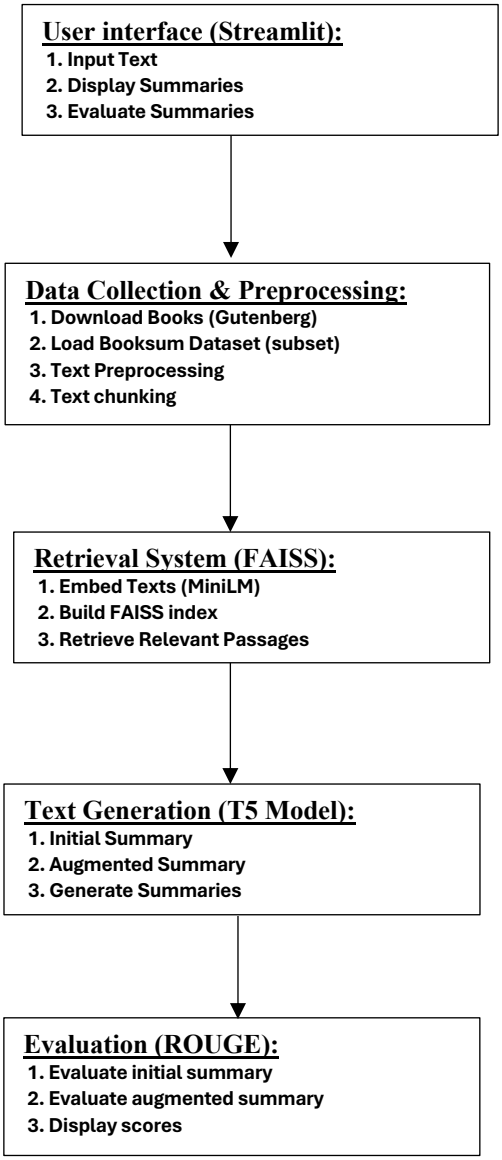
RAG-based Book Summary Solution

1. Overview

This documentation provides a comprehensive guide to the Retrieval-Augmented Generation (RAG) book summary solution implemented using the Hugging Face `T5` model, the `FAISS` retrieval system, and the `Streamlit` web interface. The solution leverages a combination of classic books from Project Gutenberg and the `booksum` dataset to generate enhanced summaries of user-provided text inputs. This document details the entire process, including data collection, preprocessing, model architecture, hyperparameters, and the evaluation process.

2. Architecture Diagram

Below is a simplified architecture diagram of the system:



3. Process Description

3.1 Data Collection & Preprocessing

- **Downloading Books:** The system starts by downloading selected classic books from Project Gutenberg. The books are stored as text files for further processing.
- **Loading the Booksum Dataset:** The booksum dataset from Hugging Face is loaded asynchronously. A subset of 1000 entries is selected, excluding the first entry used a test input.
- **Preprocessing:** The downloaded books and the booksum data undergo preprocessing where all text is converted to lowercase, and excessive whitespace is removed.
- **Text Chunking:** The preprocessed text is chunked into smaller segments (500 words each) for easier processing during retrieval.

3.2 Setting Up the Retrieval System

- **Embedding Texts:** The system uses all-MiniLM-L6-v2 from the Hugging Face sentence-transformers library to embed the text chunks into vector space.
- **Building FAISS Index:** The embedded vectors are stored in a FAISS index, which allows for efficient similarity search when retrieving relevant passages based on user queries.

3.3 Text Generation

- **Loading T5 Model:** The T5-small model is used for generating summaries. This model is pre-trained and fine-tuned for text generation tasks.
- **Generating Initial Summary:** Given a user-provided input, the T5 model generates an initial summary based solely on the input text.
- **Generating Augmented Summary:** To enhance the summary, relevant passages are retrieved from the FAISS index and appended to the input before generating a new, augmented summary.

3.4 Evaluation

- **ROUGE Metrics:** The ROUGE (Recall-Oriented Understudy for Gisting Evaluation) metric is used to evaluate the quality of the generated summaries by comparing them to a reference summary provided by the user.
- **Evaluation Process:** Both the initial and augmented summaries are evaluated, and the ROUGE scores are displayed to the user through the Streamlit interface.

4. Model Architecture and Hyperparameters

4.1 Model Architecture:

- **T5 Model:** A transformer model pre-trained on a large corpus of text data. The model has an encoder-decoder architecture, where the encoder processes the input text, and the decoder generates the output summary.
- **FAISS:** A library optimized for fast similarity search and clustering of dense vectors. It stores the embedded vectors and performs quick lookups to retrieve relevant passages.

4.2 Hyperparameters:

- **Tokenization and Generation:**
  - **Max Input Length:** 512 tokens
  - **Max Output Length (Initial Summary):** 150 tokens
  - **Max Output Length (Augmented Summary):** 250 tokens
  - **Number of Beams:** 4 (Beam search to improve generation quality)
  - **Temperature:** 0.7 (Controls the randomness in generation; lower values make it more deterministic)
  - **Top_k Sampling:** 50 (Limits the sampling pool to the top 50 tokens for diversity)
- **Embedding Model:**
  - **Model:** `all-MiniLM-L6-v2` (A lightweight, high-performance model for generating sentence embeddings)
- **FAISS Parameters:**
  - **Index Type:** Flat L2 (basic FAISS index, ideal for small to medium datasets)

5. Conclusion

This documentation outlines the complete process for building and running a RAG-based book summarization system. The combination of retrieval-augmented generation, modern transformer models, and efficient similarity search makes it a powerful tool for summarizing and analysing text. By following these steps outlined, any users can easily set up the system and generate summaries. Main challenges faced was the computational resources necessary for building this solution, hence reducing the number of downloaded books and a subset of main dataset, here below a minimum configuration necessary: