Rufus: A Tool for Web Data Preparation in RAG Agents

Summary of Approach

The goal of Rufus is to streamline the process of collecting relevant web data for use in Retrieval-Augmented Generation (RAG) agents. To achieve this, I designed a system that intelligently crawls websites, extracts relevant content, and synthesizes structured documents ready for downstream processing by language models.

Key Features

- **Dynamic Web Crawling**: Rufus utilizes Selenium to navigate web pages, allowing it to retrieve content from dynamically loaded sites, which are often challenging to scrape with traditional methods.
- **Text Chunking and Embedding**: The extracted content is split into manageable chunks, which are then converted into embeddings using a pre-trained model. This allows for efficient retrieval based on semantic similarity.
- **Answer Generation**: Utilizing a text generation model, Rufus can generate answers based on the retrieved context, enhancing the relevance and accuracy of responses.
- **Output Options**: Results can be saved in JSON or DOCX formats, providing flexibility for users in terms of how they want to store or present the data.

Challenges Faced and Solutions

1. **Dynamic Content Retrieval**: Many websites use JavaScript to load content, making it difficult for traditional scrapers to extract data.
   Solution: Implemented Selenium to allow for dynamic scraping. This enables Rufus to load web pages fully before extracting the text.
2. **Data Relevance and Duplication**: Extracted content can often contain irrelevant information, making it crucial to filter only the most pertinent data.
   Solution: Developed a mechanism to chunk text and store embeddings in FAISS for efficient retrieval based on user queries. This allows the system to focus on relevant information when answering questions.
3. **Rate Limiting and Blocking**: Rapid requests to a single domain can lead to IP blocking.
   Solution: Introduced a mechanism for rate limiting and adjustable wait times between requests to mimic human behavior and avoid being flagged as a bot.
4. **User Experience**: Initially, the tool required users to modify code for configuration changes.
   Solution: Future enhancem