Project Report: Web Document Crawler and Query Processor

Abstract

The Web Document Crawler and Query Processor project aims to develop a system for crawling web documents, constructing an inverted index, and providing a query processing interface. The objectives include implementing a Scrapy-based crawler to download web documents, building a TF-IDF-based indexer for search indexing, and creating a Flask-based processor for handling free text queries. The next steps involve testing the system, deploying it for production use, and exploring potential enhancements such as query expansion and distributed crawling.

Overview

The project involves developing a solution to crawl web documents, index them for search, and provide a user-friendly interface for querying the indexed documents. Relevant literature includes studies on web crawling, search indexing techniques, and query processing systems. The proposed system utilizes Scrapy for web crawling, Scikit-Learn for TF-IDF indexing, and Flask for query processing.

Design

The system capabilities include downloading web documents, constructing an inverted index, and processing user queries. Interactions involve the Scrapy crawler collecting web documents, the indexer building the TF-IDF matrix, and the Flask processor handling user queries. Integration is achieved through data exchange between components and adherence to defined interfaces.

Architecture

The software components include the Scrapy crawler, the TF-IDF indexer, and the Flask query processor. Interfaces include HTTP endpoints for query processing and file I/O for index storage. Implementation details cover the usage of libraries such as Scikit-Learn, Gensim, and NLTK.

Operation

To operate the system, execute the Scrapy crawler to collect web documents, run the indexer to build the TF-IDF index, and start the Flask processor to handle queries. Installation involves setting up Python environment, installing dependencies from `requirements.txt`, and running the provided scripts.

Conclusion

The project successfully implements a web document crawler, TF-IDF indexer, and query processor. Results include the ability to download and index web documents, process user queries, and retrieve relevant search results. Caveats include the need for further testing and optimization to improve performance and scalability.

Data Sources

- Web documents are sourced from [books.toscrape.com](http://books.toscrape.com/)

- Additional data sources can be integrated as needed for testing and evaluation.

Test Cases

Test cases involve validating the functionality of the crawler, indexer, and query processor. Frameworks such as Scrapy's testing tools and unit testing libraries for Python can be utilized. Test coverage includes scenarios for crawling, indexing, and querying.

Source Code

The source code for the project is available in the provided directories:

- `scrapy\_crawler` contains the Scrapy crawler implementation.

- `indexer` contains the TF-IDF indexer implementation.

- `processor` contains the Flask query processor implementation.

**Documentation and dependencies are included in the source code repositories.**

Bibliography

- [Scrapy Documentation](https://docs.scrapy.org/en/latest/)

- [Scikit-Learn Documentation](https://scikit-learn.org/stable/documentation.html)

- [Flask Documentation](https://flask.palletsprojects.com/en/2.0.x/)

- [NLTK Documentation](https://www.nltk.org/)

- [Gensim Documentation](https://radimrehurek.com/gensim/index.html)