Project codebase

Your name

Institution

04-22-2024

**Abstract**

The project aims to create a web-crawling and the extracting documents system by Scrapy library for web crawling, Scikit-Learn library for document indexing with TF-IDF and Flask library for a search query processing. The goals consist of making a scalable system to crawl webdocuments, the indexing of them so that document retrieval can be carried out efficiently applying TF-IDF, and interface that is end-user friendly for the user to search the relevant documents based on a text input. Follow-up plans involve further refinement of the indexing system, application of alternative retrieval methods, and possibly even utilization of the system in an actual world application.

**Overview**

The project integrates web crawling, document indexing and query processing in order to provide pertinent information organizing it from the internet. The literature that is related covers researches on using web crawling techniques, TF-IDF indexing, and cosine similarity to improve the relevancy of the search results. Through Scrapy for web crawling, Scikit-Learn for document representation, and Flask for creation of a query interface based on the web, the proposed system will work.

**Design**

The system is designed to handle the following capabilities:

* Web crawling: Utilizes Scrapy to download web documents based on specified criteria.
* Document indexing: Uses Scikit-Learn to compute TF-IDF scores and construct an inverted index.
* Query processing: Implements a Flask-based processor to handle user queries and return relevant search results.

Interactions within the system include:

* Crawling module interacts with web servers to download HTML content.
* Indexing module processes crawled documents to compute TF-IDF vectors.
* Query processor module receives user queries, matches them against the indexed documents, and returns relevant results.

Integration involves combining the output of the crawling module with the indexing and query processing modules to create a functional information retrieval system.

**Architecture**

The software components include:

* **my\_crawler.py**: Scrapy spider for web crawling.
* **indexer.py**: Script to compute TF-IDF scores and build the inverted index.
* **processor.py**: Flask application for query processing and search results presentation.

Interfaces include HTTP endpoints for query submission and response retrieval.

**Operation**

To operate the system:

1. Install Python dependencies (**scrapy**, **scikit-learn**, **flask**).
2. Run the web crawler (**my\_crawler.py**) to download web documents.
3. Execute the indexer (**indexer.py**) to compute TF-IDF scores and build the inverted index.
4. Start the Flask server (**processor.py**) to handle user queries and return search results via HTTP.

**Conclusion**

The project achieved success in developing a functional web crawling and information retrieval system. Results include the ability to crawl web content, index documents using TF-IDF, and retrieve relevant documents based on user queries. Further optimization and enhancements are required for scaling the system and handling larger datasets.

**Data Sources**

Web sources are dynamically crawled using the Scrapy framework. Specific URLs, downloads, or access information are not applicable as the system retrieves data from publicly accessible websites such as [www.wikipedia.com](http://www.wikipedia.com) .

**Test Cases**

Test cases involve:

* Verifying the correctness of web crawling results.
* Ensuring accurate computation of TF-IDF scores and indexing.
* Validating query processing and search result relevance.

Coverage includes testing various scenarios, edge cases, and performance evaluations.

**Source Code**

The source code listings are available in the project repository:

* **my\_crawler.py**: Defines the web crawler using Scrapy.
* **indexer.py**: Computes TF-IDF scores and builds the inverted index.
* **processor.py**: Implements the Flask-based query processor.
* Dependencies are documented in the **README.MD** file.

**Bibliography**

Yin, Fulian, Xiating He, and Zhixin Liu. "Research on scrapy-based distributed crawler system for crawling semi-structure information at high speed." In *2018 IEEE 4th International Conference on Computer and Communications (ICCC)*, pp. 1356-1359. IEEE, 2018.

Fugkeaw, Somchart, Lyhour Hak, and Thanaruk Theeramunkong. "Achieving Secure, Verifiable, and Efficient Boolean Keyword Searchable Encryption for Cloud Data Warehouse." *IEEE Access* (2024).

Dwyer, Gareth. *Flask By Example*. Packt Publishing Ltd, 2016.