Project Report

Gopinath Robba
A20543707

Please provide the following regarding the project report (Note: Can be generated from the Project Codebase):

Abstract

We will use scrapy to download website data and images from books.toscrape.com, a demo website built for web scraping purposes, which contains data about 1000 books.

This report details the development of a web-based search system using Scrapy, Scikit-Learn, and Flask to crawl, index, and query web documents. The Objective is to build an efficient search system capable of processing and responding to text-based queries with relevance ranked results.

Overview

Solution involves a three-stage solution: crawling web data, indexing the data, and setting up a query processor.

Design and Architecture

Folder structure and details

books_crawler: scripts to crawl the webpages and get the requried html files
indexing: for indexing of crawled data using TF-IDF and FAISS
Query_processor: Real-time query handling with JSON input validation and ranked output results.

Operation

books_crawler:
pip install scrapy
scrapy startproject books_crawler
scrapy crawl books

Indexer:
pip install numpy scikit-learn faiss-cpu beautifulsoup4
pip install faiss-cpu

Execution after directing to Indexer directory
python preprocess.py
python build_index.py
python create_faiss_index.py

query_processor:
pip install flask
python app.py

Data Sources

https://books.toscrape.com/

Test Cases

Testing the application. Make an API call from Postmaster or a curl statement from the terminal. Example below:

curl -X POST -H "Content-Type: application/json" -d '{"query": "harry porter", "k": 5}' http://localhost:5000/search

Output:

```
gopinath@Gopinaths-MacBook-Pro spiders % curl -X POST -H "Content-Type: application/json" -d '{"query": "harry porter",
"k": 5}' http://127.0.0.1:5000/search
{
  "results": [
    "Document 79",
    "Document 107",
    "Document 17",
    "Document 12",
    "Document 6"
  ]
}
```

Source Code

Folder structure and details
books_crawler: scripts to crawl the webpages and get the required html files
indexing: for indexing of crawled data using TF-IDF and FAISS
Query_processor: Real-time query handling with JSON input validation and ranked output results.

Bibliography

1. https://docs.scrapy.org/en/latest/topics/spiders.html
2. https://chat.openai.com/
3. https://www.youtube.com/
4. https://www.elastic.co/
5. https://github.com/luigifilippochiara