Project Report

CS 6200

Focused Web Crawler for Apple Technology

Team 8: Parv Thakkar

Introduction

I am working on a project based on text acquisition part of the information retrieval task where I'm developing a focused web crawler that is designed to selectively gather documents from the internet to populate an Elastic Search cluster. My goal is to retrieve 4000 unique documents, and I prioritize them based on keyword and in link counts to ensure their relevance. I am also following the politeness policy to prevent overwhelming websites with requests and utilize robots.txt for proper crawling and not hurt the websites and avoid getting banned. I processed the documents and extracted them using Beautiful Soup and kept track of in links and out links for the crawled website.

The focused crawlers for a particular user or a group of users can be useful as it will work better for a specific task when compared to a general search engine as the documents stored for the purposes will be more relevant. The number of documents is also less in a focused crawler which may cause the overall computation power required to be less when compared to a general web crawler. It may be easier for a focused crawler to keep the web pages updated. Different methods can be applied to the focused search while crawling like prioritizing keywords and in link count which may improve crawling of a focused crawler.

In summary, my task streamlines the process of gathering, organizing, and accessing web-based information, making it valuable for users who are seeking specific, relevant, and easily retrievable data.

Query Analysis

Queries in this context, focused on Apple-related topics such as "Macbook Pro" or "Apple SDK," typically take the form of specific product names, software tools, or broader concepts associated with the Apple ecosystem. The queries cannot be very specific like “Macbook Pro Battery Life”, etc. The model due to limited number of documents crawled is not able to provide relevant documents to such specific queries so if the query is more general as mentioned in earlier examples it is able to retrieve relevant results. The nature of these queries suggests that users are seeking general information related to Apple’s product or related to Apple technologies, often with a technical or consumer-focused angle.

Relevant results for these queries would encompass a wide range of Apple-related content. For product-specific queries like "Macbook Pro," relevant results would include link to the document which is most relevant with respect to the query and the option to view the content that the document refers too. The queries might be able to return more specific results as we increase the number of crawled documents and summarize the content to avoid viewing the whole document.

The number of relevant results per query can vary greatly depending on the specificity and popularity of the query. Around 10-15 documents are retrieved for a particular query. If the query used is a general query related to Apple's products or technologies, then most of the results obtained will have higher chance of relevance and if a specific query is used depending on the query there may be only a few relevant results.

For general queries, a larger set of relevant results would be appropriate, whereas more specific queries might yield a smaller set of relevant but a large amount of non-relevant results. Ideally, there should be enough results to provide comprehensive coverage of the topic without overwhelming the user with redundant or irrelevant information.

The organization of results is performed by the Elastic Search cluster based on the query it will return the user relevant results based on the TF-IDF or BM25 to determine the relevance of the document based on the search query and displays them in that order.

The evaluation based on this problem is the management of the links to be crawled by the crawler as the links to be crawled are scored by the frontier and processed in the order based on the score calculated by the number of keywords and in links for a particular website which shows the importance and relevance of the website to prioritize the crawl of that website first.

Methodology

My implementation of the focused web crawler involves several key components designed to efficiently gather and store relevant documents in an Elastic Search Cluster. The process begins with a set of seed URLs, which form the starting point for the crawler. I've constructed a frontier, essentially a priority queue, which manages these URLs and subsequently adds more URLs to the queue. The primary goal is to crawl and store 4000 unique documents.

The frontier is optimized to prefer links with higher keyword and in link counts, assigning each link an importance score to prioritize them. This approach ensures that the most relevant and authoritative content is crawled first. To respect website protocols and avoid any potential harm, the crawler adheres to the robots.txt files of websites and follows a strict politeness policy, limiting requests to one per second. It also checks if the website allows crawl or not to avoid any issues in the process and avoid harming the websites.

For efficiency in data extraction and ease of processing documents, I focus exclusively on HTML files. As the crawler navigates a webpage, it collects out links which are then added back into the frontier, creating a cycle that continues until the goal of 4000 documents is reached. Regular expression techniques are employed to avoid crawling the same URL multiple times. This includes strategies like lowercasing the hosts and removing port numbers from URLs.

Once a document is crawled, it is parsed and stored using Beautiful Soup, and the in links and out links are stored as JSON files. Finally, the documents are added to the Elastic Search index. After, adding the documents to the Elastic Search Index Streamlit is used to create a web user interface where user can write the desired query and get the result relevant link and content extracted from the document.

Results

The crawler is successfully able to crawl 4000 unique documents as mentioned in the grade contract. Also, other relevant goals including having an approach to calculate the relevance of the link and prioritize it by maintaining a priority queue to prefer crawl of relevant links. The use of a priority queue ensures that resources are not wasted on less pertinent links. Also, the crawler works with robots.txt file to avoid harming website while crawling. The implementation of regular expressions to avoid revisiting the same URLs ensures the crawler's efficiency and prevents the unnecessary use of resources. This approach enhances the quality of the dataset by reducing duplicates.

I used Beautiful Soup for parsing and the structured storage of in links and out links in JSON format contribute to the organized and accessible nature of the data. This organization is vital for subsequent data retrieval and analysis. The crawler is efficiently able to crawl documents but takes longer to crawl 4000 documents and the number of documents seems less. Due to a smaller number of documents the search is not able to perform better on specific queries but is able to answer the general queries in a better way.

The integration of Streamlit for the user interface allows for a user-friendly and interactive way to query the Elastic Search index. This aspect is key to the overall usability of the system, making the information accessible to users in a practical and efficient manner.

Sample Query Analysis

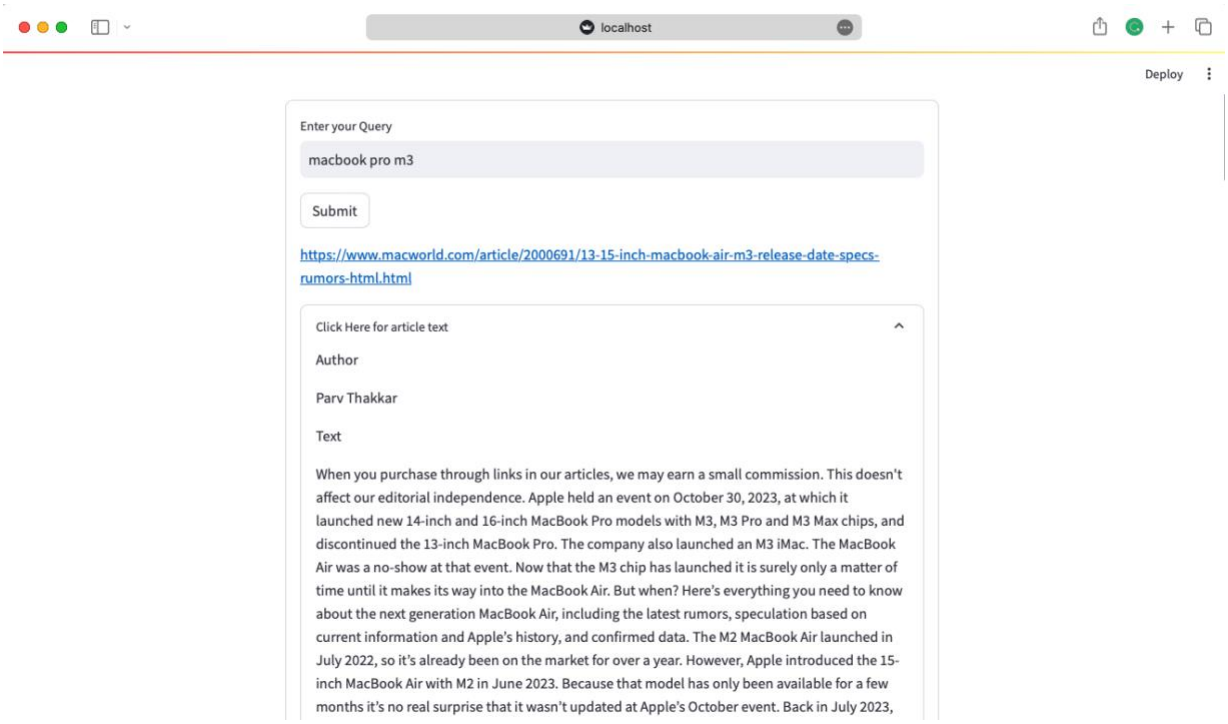
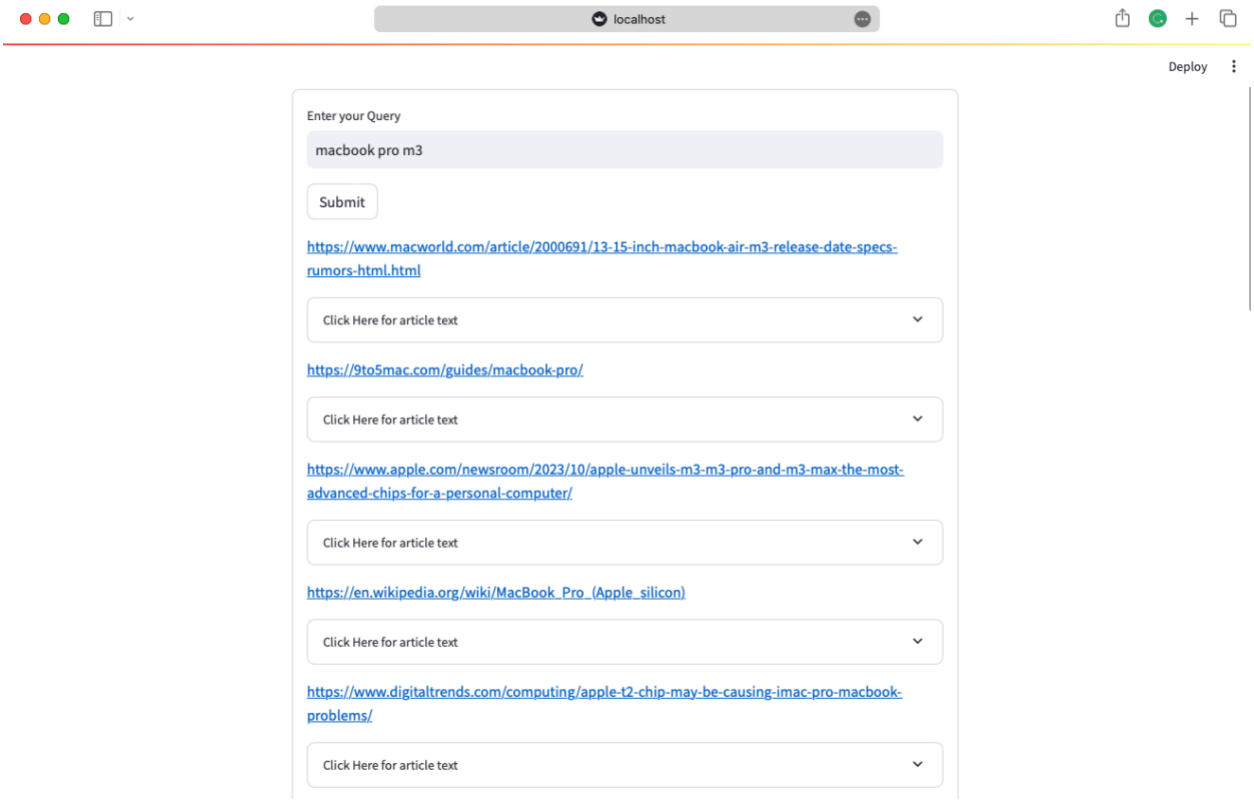


Figure 1 The images display the web version and the results for the query "macbook pro m3"



1.) Query: “Macbook pro M3”
    a. The user is trying to find articles related to the macbook pro m3 and get more information about the new chip products that were recently released and get more information on the products like review, features, specifications, etc. The user might be considering an upgrade, seeking to compare models, or simply staying up to date with the latest Apple hardware.
    b. The user may check how relevant and accurate the returned article is it a lates information or an outdated version. The user will also check the authenticity of sources considering as the internet can be prone to wrong information. The user would look for articles that specifically mention the "MacBook Pro" with an "M3" chip, not earlier models or other products.
    c. Relevant Results include (2 results) examples are:
        i. https://www.macworld.com/article/2000691/13-15-inch-macbook-air-m3-release-date-specs-rumors-html.html
        ii. https://www.apple.com/newsroom/2023/10/apple-unveils-m3-m3-pro-and-m3-max-the-most-advanced-chips-for-a-personal-computer/
    d. Non-Relevant Results include (10 results) examples are:
        i. https://www.techradar.com/reviews/apple-macbook-air-m12020
        ii. https://en.wikipedia.org/wiki/MacBook_Pro_(Apple_silicon)
        iii. Most of the non-relevant documents are related to previous generation chips or macbook.
    e. Notes:
        i. The crawler was able to scan latest files related to the M3 chip, but it did not scan a good number of relative documents the issue may be based on the seed urls. While trying the same search on google it returns mostly urls related to buying.

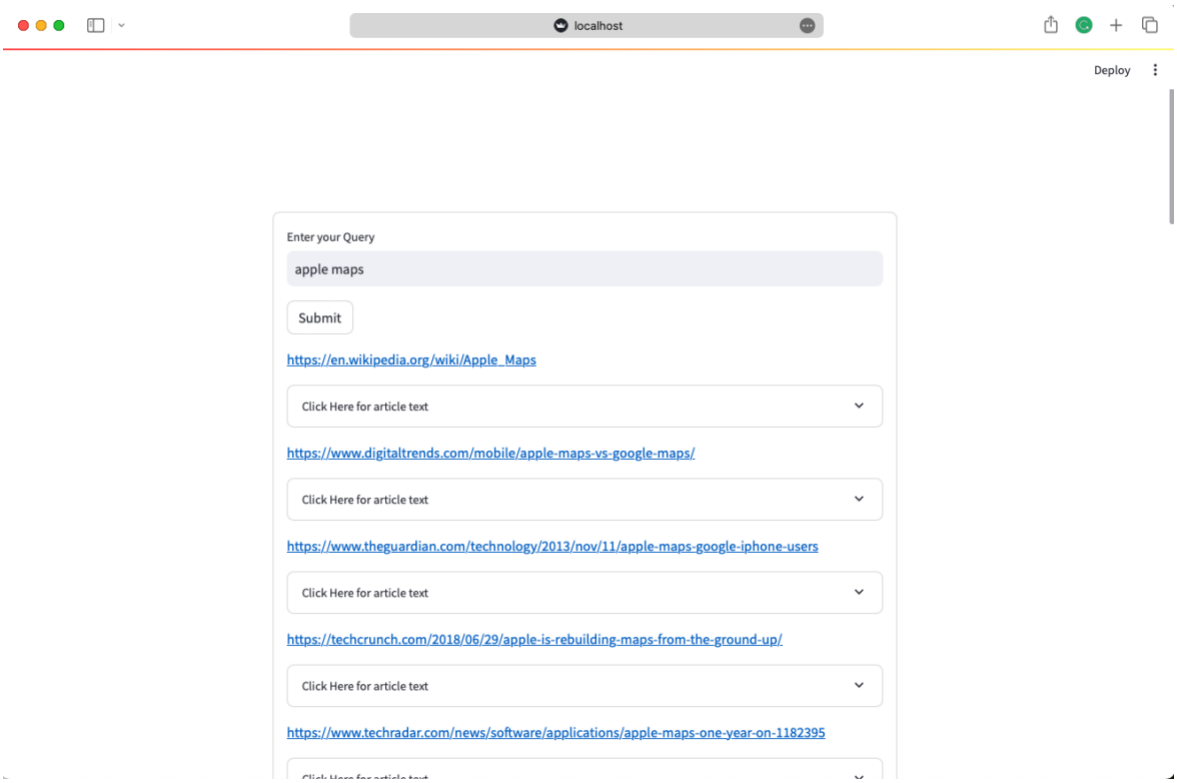


Figure 2 The above image shows the output for the query "apple maps"

2.) Query: “Apple Maps”

a. The user is trying to find articles related to the apple maps technology provided by apple. The user is likely looking to gather information on Apple's mapping service. This could include a range of motivations such as learning about the app's features, understanding its navigation capabilities, finding out about recent updates or improvements, comparing it with competitors like Google Maps, or troubleshooting issues.

b. The user would likely prefer the most up-to-date information. Since digital mapping services frequently update, articles or pages that discuss the latest version of Apple Maps would be considered more relevant. The authenticity of the sources is also important. The user would look for content specifically discussing the features, updates, or comparisons of Apple Maps rather than general information about Apple or its other services.

c. Relevant Results include (15 results) examples are:
   i. https://en.wikipedia.org/wiki/Apple_Maps
   ii. https://www.digitaltrends.com/mobile/apple-maps-vs-google-maps/

d. Non-Relevant Results include (10 results) examples are:
   i. https://www.cnn.com/2019/06/03/tech/wwdc-2019-apple-keynote/index.html
   ii. https://www.techradar.com/news/software/applications/apple-maps-one-year-on-1182395
   iii. The non-relevant results for this query mostly include old information related to apple maps.

e. Notes:
    i. The issue with this was that the crawler did not scan documents based on freshness. So, the query can return latest documents related to the query. While doing the same search on google it fetches latest articles though both are able to give the Wikipedia page as output. The crawler would have crawled this url as it was one of the seed urls.

References

1.) Project Link: https://github.com/parv212/Topic-Focused-Crawler-and-Search
2.) https://www.elastic.co/guide/en/elasticsearch/client/python-api/current/index.html
3.) https://www.geeksforgeeks.org/beautifulsoup-scraping-link-from-html/
4.) https://www.tutorialspoint.com/what-are-focused-web-crawlers