CS 242 Project Report Part A - Group 14

CHENGKUN LYU, 862466258, clyu014@ucr.edu, UC Riverside, USA
HAIRU WEN, 862467599, hwen020@ucr.edu, UC Riverside, USA
JIAHUA XUE, 862464743, jxue041@ucr.edu, UC Riverside, USA
PO-CHU CHEN, 862465074, pchen243@ucr.edu, UC Riverside, USA
YI JIANG, 862469307, yjian221@ucr.edu, UC Riverside, USA

1 COLLABORATION DETAILS

In this project, we started teamwork based on the two parts of part A, crawler and index. For the work of crawler, we studied PRAW documentation, saved json files, multi-threading, and optimized crawling content. The index part is relatively as a whole. Learn and write code, and there are also some related python guidance and executable file guidance. Specifically, each team member has following contributions to our project:

- **Yi Jiang:** Yi Jiang participated in all code parts, provided support for python and various libraries, proposed in PRAW the need to use others because it is not thread safe, and discovered and used ThreadPool and related APIs. In the crawler part, Yi participated in designing the initial crawler structure, and at the same time changed the crawler from targeting specific subreddits to searching related subreddits based on specific topics based on the initial results. In the index part, Yi participated in writing the code for reading json file.
- **Po-Chu Chen:** Po-Chu developed a LaTeX template in line with ACM style guidelines, ensuring our report was professionally formatted. Additionally, he organized the project's structure and monitored its progress, keeping the team on track. Po-Chu also wrote about the crawling system's architecture and overview including the core crawling strategy and multi-threading and multi-processing approach in the report, providing clear explanations of complex concepts. Lastly, Po-Chu contributed advice on optimizing our data crawling approach, helping to improve the system's effectiveness.
- **Jiahua Xue:** In summary, Jiahua Xue's role in the project involved identifying key limitations, organizing the work, and suggesting improvements. Jiahua pointed out constraints in capturing discussions, hardware resource, and codebase dependencies, which were important for this project. By structuring the project and keeping track of progress, Jiahua helped maintain focus and efficiency. Jiahua's advice on data crawling optimization directly contributed to better system performance.
- **Chengkun Lyu:** In the indexing part of our project, Chengkun played a role within the PyLucene framework, focusing on the Title, Context, and URL fields. Chengkun was instrumental in devising a novel method for tokenizing and merging the title and body text of Reddit posts, a key factor for enhancing search accuracy and relevance ranking. Additionally, Chengkun's management of the URL Field was pivotal, as Chengkun ensured the efficient capture and storage of essential post metadata.
- **Hairu Wen:** Hairu Wen participated in organizing the project's structure and developing the implementation details. She gave a version of code to do the crawling and write down the files' links with spider, link finder, url parser and output writer. Hairu contributed to the crawling and indexing parts, came up with the way to realize multi-threading, and provided an initial applicable version to crawl with multi-processing. Hairu also gave advices on code optimization and helped improve the efficiency.

2 INTRODUCTION

For this project, our goal is to develop a sophisticated search engine that leverages modern technologies for web crawling, indexing, and query processing. For Part A(Collect your data and Index with PyLucene or PyElasticSearch), we plan to use PRAW to collect hot and controversial posts from Reddit. Our focus will be on extracting education content, specifically on the board "Education and Learning". We then decided to use PyElasticSearch to efficiently index the collected data by region. At the beginning we thought PyElasticSearch provides robust features for full-text search and can be more focused on keywords.So at the beginning our plan for part A is using PRAW on topic "Education and Learning" and PyElasticSearch, but during the working, we found that only education and learning with PRAW on Reddit can not get enough data for us, so we add several more topic about the education, such as “course”, “CSmajor”, “university” etc.. And during the indexing work, PyElasticSearch is a tool that needs some cloud space, and also our TA is not very familiar with it, so we changed the tool to PyLucene.

3 OVERVIEW OF CRAWLING SYSTEM

3.1 System Architecture

The architecture of this system is based on our design. It can discover related Subreddits from Reddit based on predefined topics, extract data from each Subreddit in turn, and use the Reddit API through the PRAW library to accurately identify Subreddits and crawl posts.

3.2 Targeting and Searching

The core of our strategy is to focus on specific topics, leading the system to search for relevant subreddits. Here we ensure the relevance and specificity of the sources of collected data through a targeted approach of "reddit.subreddits.search", after which we take advantage of the ".hot" and ".controversial" functions of the Subreddit instance, This allows extraction of posts and comments.

3.3 Crawling Strategy

The system uses a topic-driven search strategy that focuses on specific topics to ensure that the data collected is highly relevant. This targeted approach is valuable for conducting thematic analysis or research requiring specific types of data.

3.4 Subreddit Identification

By operating within subreddits related to designated topics, the system maximizes the relevance and quality of content collected, enriching the data set with diverse perspectives and insights.

3.5 Multithreading Approach

First of all, we have to mention the non-thread safety features of the PRAW library. In our tests when trying to use traditional Tread, there were conflicts between instances, instances crashed during crawling or became another instance. At the beginning, we tried to use other methods to accelerate search operations. First, we recorded the most original and common iteration method. This method is the slowest and will also trigger PRAW’s access restrictions, but it is more stable. Later, we tried to perform batch searches by superimposing keywords and searching for several topics at once, but PRAW did not support this method, which resulted in less data. Finally, according to the PRAW documentation,

we proposed to use "multiprocessing.pool.ThreadPool" "Replace traditional threads by using a separate instance for each thread through a multiple process thread pool. This approach ensures safety, concurrent operations, and efficient resource management. After our research, we found that based on the same idea, we thought of ThreadPoolExecutor, which is also one of the implementation methods of ThreadPool and is simpler and faster.

3.6 ThreadPool Implementation

Each thread in ThreadPool uses its own independent PRAW Subreddit instance to operate. Different threads obtain data from different subreddits in parallel, and significantly improve the efficiency of the crawler.

3.7 Data Storage

The folder where the data is saved is named "RedditData". The extracted data is organized and stored in JSON files and named after their respective subreddits. The overall logic is to create posts crawled in each Subreddit in the code. The corresponding Python dict saves the entire Subreddit's dict in a list, and then saves it to a json file named after the Subreddit through the json library. This way of organizing files facilitates data storage and subsequent processing, ensuring efficient data management. It also makes it easy to identify the approximate size of each Subreddit and to find relevant posts for a specific Subreddit.

4 OVERVIEW OF THE PYLUCENE INDEXING STRATEGY

4.1 Fields in the PyLucene index

In the code, documents are indexed with three main fields: Title Field, Context Field, and URL Field. For Title Field and Context Field: These two fields combine the title and the body of the post. It's tokenized, meaning it's broken down into individual words or tokens. Additionally, it stores positional information and term frequencies. This is crucial for accurate search results because it allows for proximity-based relevance ranking. Why we do this is because for Reddit posts, sometimes the title also plays the role of body text, but the body has no content. Therefore, because of this particularity of Reddit, our title cannot be considered the same as the title of an ordinary web page. It needs to play the same role as the context. URL Field: This field stores the URL of the post. It's a stored field and is likely intended to provide additional metadata about the document.

4.2 Text analyzer choices

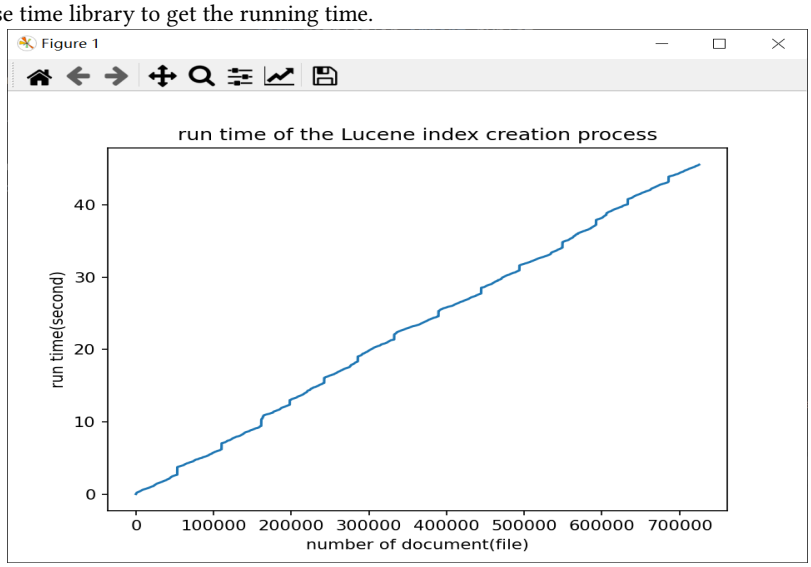
For our analyzer, the StandardAnalyzer is used for our project. We use this for several reasons:

- It's a built-in text analyzer provided by Lucene.
- It performs tokenization, which splits text into individual words or tokens based on whitespace and punctuation.
- It also handles lowercasing, removing common English stop words, and performs stemming (reducing words to their root form). However, stemming might not be explicitly mentioned in the code but is often included in standard tokenization.
- This analyzer is a good general-purpose choice for indexing and searching English text data.

So in conclusion, even though PyLucene provides other analyzers with specific characteristics tailored for different languages or use cases, we still chose the StandardAnalyzer.

4.3 Report the run time of the Lucene index creation process

For real run time, we test for about 500MB data, please see the graph, to get information about it, in our index code we use time library to get the running time.



5 LIMITATIONS

In the development of the Reddit crawler system, one of the fundamental challenges encountered pertains to the inherent limitations imposed by the dependency on the Reddit API. This dependency introduces significant constraints on the efficiency and stability of the crawler, primarily due to the rate limits and potential changes in the API.

The Reddit API enforces rate limits that restrict the number of requests a client can make within a given timeframe. This limitation poses a direct impact on the crawler's efficiency, as it caps the speed at which data can be retrieved. In scenarios where extensive data collection is required across multiple subreddits, the rate limits can lead to interruptions or delays in data fetching. Consequently, this constraint affects the continuity and completeness of the data collection process, potentially leading to gaps in the dataset and hindering the crawler's ability to provide a comprehensive snapshot of the discussions and trends within the Reddit community.

In light of these limitations, it is evident that the dependency on the Reddit API introduces considerable challenges to the crawler's operational efficacy. The constraints imposed by rate limits necessitate the implementation of robust error-handling and adaptive mechanisms within the crawler to mitigate the impacts on data collection and ensure the stability and reliability of the system.

Assessing the Impact on Data Quality and Diversity in Reddit Crawler Design The design of the Reddit crawler, with its focus on extracting 'hot' and 'controversial' posts, presents specific limitations regarding the quality and diversity of the data collected. This approach, while efficient for capturing trending and polarizing content, may inadvertently overlook valuable discussions characterized by less interaction but rich in content.

5.1 Limitations in Capturing Comprehensive Discussions

The crawler's strategy to prioritize 'hot' and 'controversial' posts is grounded in the assumption that these categories represent the most engaging and relevant content within a subreddit. However, this method may neglect posts that, despite having fewer interactions, offer substantial insights or present nuanced perspectives on the topic at hand. Such posts, often found beyond the immediate visibility of 'hot' and 'controversial' labels, contribute to a more holistic understanding of community discussions. The exclusion of these less interactive but content-rich posts from the dataset can result in a skewed representation of the subreddit's discourse, potentially omitting nuanced or emerging viewpoints.

In summary, while the focused approach of extracting 'hot' and 'controversial' posts enables the crawler to efficiently gather content of immediate relevance and high engagement, it introduces significant constraints on the quality and diversity of the collected data. These limitations underscore the need for a more inclusive data collection strategy that extends beyond the prevailing metrics of engagement to ensure a more representative and nuanced dataset.

5.2 Limitation of Hardware Resource

5.2.1 Hardware Limitations and Performance Bottlenecks in Parallel Processing with PRAW. The integration of ThreadPool for parallel processing in the Reddit crawler, while addressing the thread safety concerns associated with PRAW, introduces a new dimension of challenges related to hardware limitations. Despite the software-level optimizations, the actual performance of the system remains heavily contingent upon the underlying hardware resources, particularly the CPU. This dependency becomes critically evident in scenarios demanding high concurrency, where the hardware's capabilities can significantly constrain the system's throughput and efficiency.

5.2.2 CPU as a Performance Bottleneck. The utilization of ThreadPool facilitates the execution of multiple threads concurrently, leveraging the multicore architecture of modern CPUs to enhance the data collection process. However, the effectiveness of this parallel processing is inherently limited by the CPU's core count and its ability to handle multiple threads simultaneously. In the context of the Reddit crawler, where extensive data extraction tasks are executed in parallel, the CPU's computational capacity and threading efficiency emerge as pivotal factors determining the system's overall performance. The constraints imposed by the CPU are particularly pronounced in high-concurrency scenarios, where the demand for computational resources exceeds the available supply. Under such conditions, the system may experience throttling, where the execution of tasks is significantly slowed down, or in extreme cases, some tasks may be queued or dropped, leading to inefficiencies and potential data loss. This scenario underscores the critical role of the CPU as a bottleneck in the parallel processing framework, where the hardware's limitations directly impact the system's ability to scale and perform under heavy loads.

5.2.3 Addressing Hardware-Induced Limitations. To mitigate the performance bottlenecks induced by hardware constraints, it is essential to adopt a holistic approach that encompasses both software optimizations and strategic hardware upgrades. On the software side, optimizing the crawler's algorithmic efficiency, implementing efficient data structures, and fine-tuning the ThreadPool configuration can yield significant performance improvements. Concurrently, investing in hardware upgrades, such as increasing the CPU core count or opting for CPUs with better multi-threading capabilities, can provide a more direct and impactful solution to the performance challenges.

5.2.4 Conclusion. In summary, while the use of ThreadPool in the Reddit crawler's design addresses the thread safety issues associated with PRAW, the system's performance remains tightly bound to the capabilities of the underlying

hardware. Recognizing and addressing the limitations imposed by the CPU is crucial for ensuring the scalability and efficiency of the crawler in handling data-intensive tasks in a parallel processing environment.

6 OBSTACLES AND SOLUTIONS

The crawler handles duplicate pages through the PRAW feature and the Pythonlist feature. First of all, based on our communication with TA, for this project, we only consider the same URL as a duplicate page. Therefore, according to the PRAW feature, the same post will not be returned multiple times in each Subreddit instance. For example, in .hot, all Posts will only appear once, so in order to avoid duplicate posts we only need to care about two points, 1. Avoid duplicate Subreddit instances, 2. Avoid the same post appearing hot and controversial. For Subreddit instances, we obtain them through Python set The instance list is deduplicated once, and for the post part, before we add the postdict to the postlist, we will check the postdict to see if it is in the postlist. For multiple threads, we have already talked about it in the Overview of Crawling System part. For other questions, please also see our Crawling System part and Indexing part.

7 INSTRUCTION ON HOW TO DEPLOY THE CRAWLER AND LUCENE INDEX

We have three executable file, setup.sh, runcrawl.sh runindex.sh, before everything, please use chmod +x to give permission for each, then use./setup.sh to set up the environment, then use ./runcrawl.sh to run the crawler, use ./runindex.sh to run the index.