Project Report - Part B
CS 242 : Information Retrieval and Web Search

Abhishek Ayachit, Gowtham Tumati, Lovepreet Singh Dhaliwal, Sudip Bala, Teja Vemparala
{aayac001, gtuma002, ldhal001, sbala027, tvemp001}@ucr.edu
Department of Computer Science, University of California - Riverside

Project Overview:

Information is key in today's world and the Internet plays a major role in providing adequate amounts of information about anything. Data is available in abundance over the Internet, which can be extracted easily by anyone. People rely on the information available on a large scale. Information could be of any form, articles, pictures, videos, and other well-known forms. Learning from the knowledge extracted on the internet, it can be used to get profitable productivity thereby prospering the exchange.

It is clear enough that we have adequate amounts of information about sports, latest news of events, leagues and all related stuff. Taking the data from Twitter into consideration, we make a search engine for the user, wherein the user can search for any sport-related term and have all the information about it, at hand.

Project Flow/Architecture:

We followed a simple architecture for building the web interface and indexing the crawled data. In the first phase of the project, we crawled the data using Twitter Streaming API. On completing the procedure, we used Lucene to index it. Following that, we now use Hadoop Map-Reduce to index the data. It would be the choice of the user, as to which procedure they would like, for retrieving the data, either by Hadoop or by Lucene.
Client is the user interface which is used to submit the query to the server. We implemented a crawler which used to fetch data from Twitter. We created a database using MongoDB(NoSql) and implemented an indexer using Hadoop and Lucene. Then we return the fetched data to the client through the server. We can refer to the figure for the architecture of the search procedure.

Technologies Used:

- MongoDB: Used to store the results, since it stores data in JSON type documents and that sharding(required for future scope) is already a feature in it.
- Apache Tomcat: used to run the server
- Jersey: A Java framework, has been used to develop backend APIs for RESTful web services.
- User Interface: HTML, CSS, JQuery
- AJAX: For requests to be made to the server
- Lucene: Used to generate index
- Hadoop: Used to generate Inverted Index

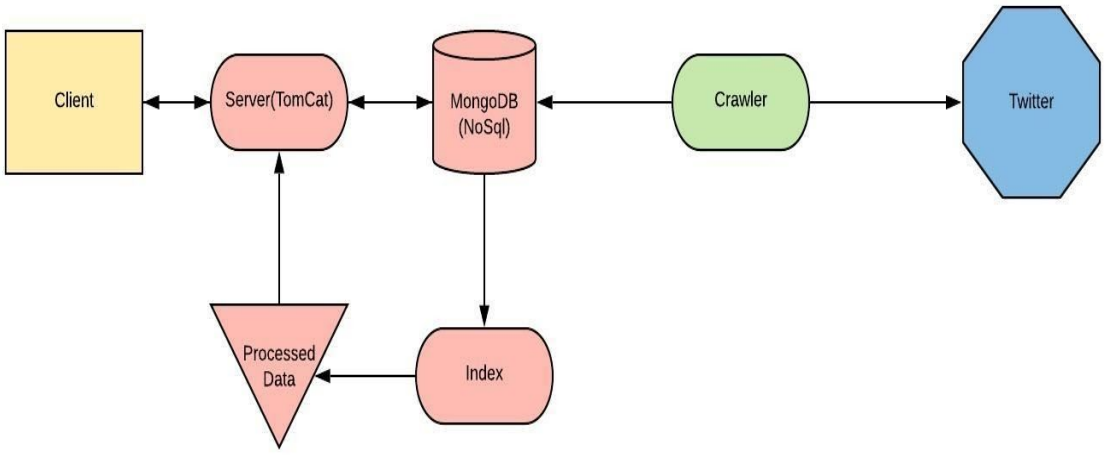


Figure. Architecture for the search procedure

Inverted Index generation using Map-Reduce

First, the Mapper reads the tweet, hashtag attributes of each tweet instance, and then, a key-value pair is generated, and similarly, the reducer counts the frequency of words.

Tasks for the Mapper:

1. Read tweets and then perform preprocessing
2. Compute term frequency for each word
3. Key value pair is generated where key is the word, and value is the documentID

Tasks for the Reducer:

1. Input is received as key, value pair
2. List is displayed in the decreasing order of the term frequency.

Hadoop Indexing and Search

After the crawling process is done, the tweet data must be indexed, in order to ensure quick retrieval. In case there is an implementation of a single machine program to index the data, it would become highly inefficient and non-scalable, with the reason being the huge amount of data processed. In this regard, we have implemented the concept of Inverted Index for the crawled data. In general, Hadoop MapReduce programs support a framework for storing and processing the data, in a distributed environment, and make an impact. But, in our case, we could exploit the advantage HDFS had, towards distributed storing and processing by the use of MapReduce for generation of an Inverted Index.

Ranking based on Index created using Hadoop

The inverted index generated using Hadoop is maintained in-memory on the server, in order to retrieve the tweet data stored in MongoDB faster. As soon as the server gets a request from the

user, it reads the string, splits the input phrase into words and uses the inverted index which is stored in-memory to get the posting list for all the words. When the list is retrieved, ranking function (tf x idf), is used wherein tf is retrieved from the posting list, and idf is the logarithmic value of the fraction total tweets and the length of the posting list.

$$idf = \log_{10}(\text{total tweets}/ \text{length of posting list})$$

The lists are maintained in a map with documentID being the key and the ranking score being the value. Sorting is done based on the scores. The server retrieves the complete tweet instance from MongoDB using documentID and returns them as JSON type file, for all the top ranked documents

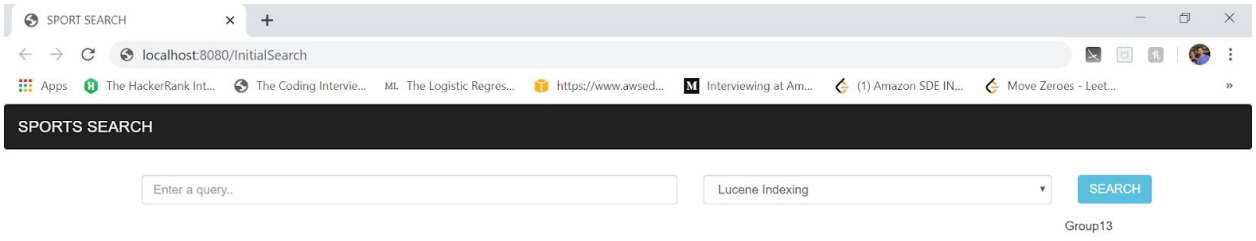
Lucene Indexing and Search

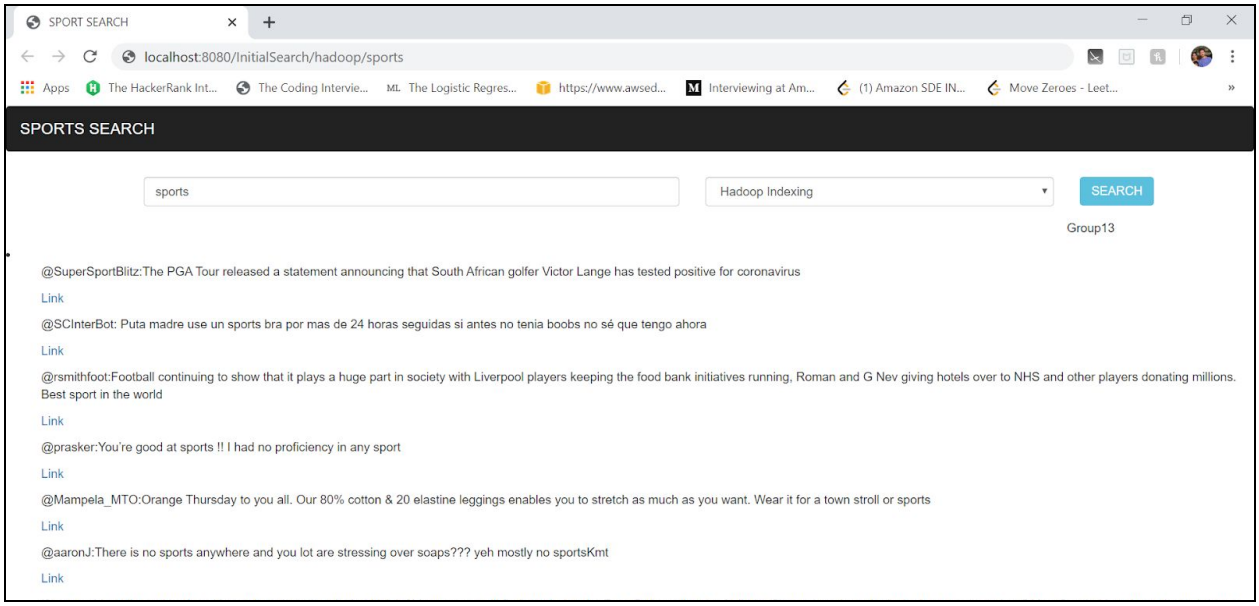
In Part A, we indexed the files using Lucene, in order to generate the search results. The string is read, parsed and Lucene parses those words to search, after splitting into individual words. On passing this to the search, relevant documents are returned, without any order of the rank. These results are the response to the query by the user, and in JSON type document.

API Details:

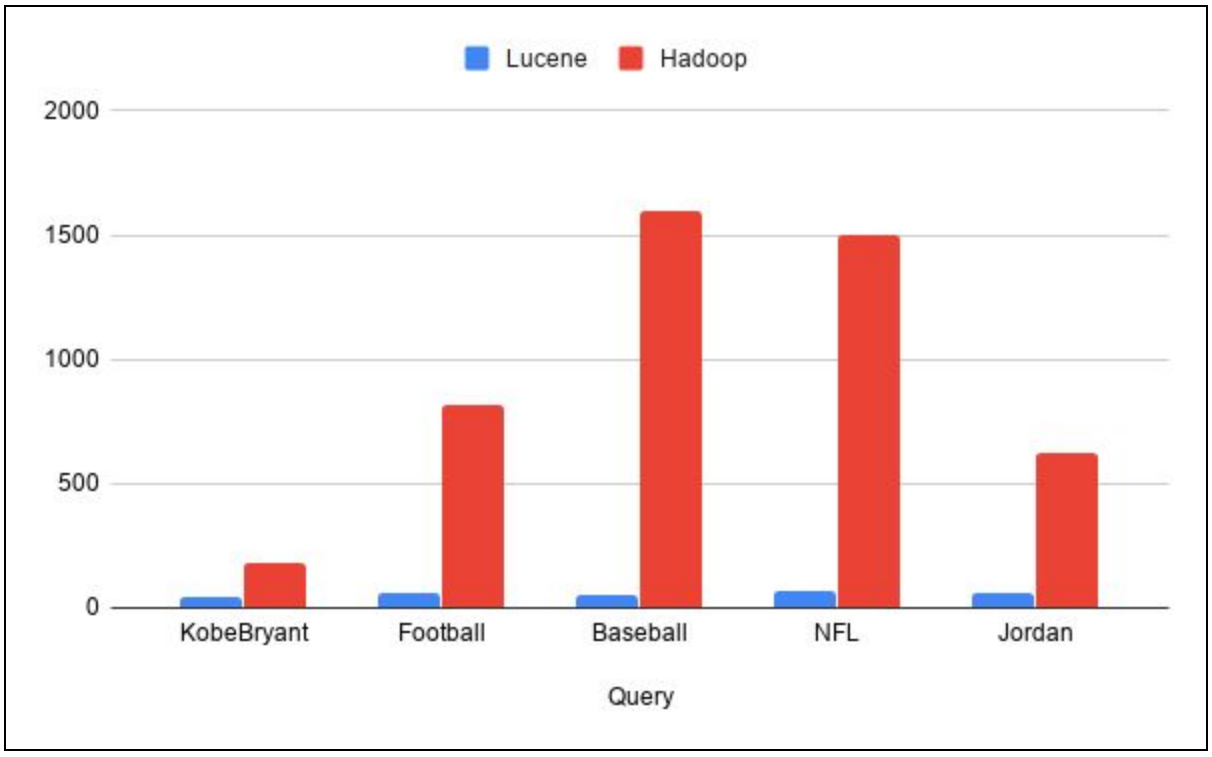
Web Interface:

We kept the design of the web interface at a minimalist level. It is a simple thing, containing a search box for the query and a dropdown, to select between Lucene and Map-Reduce. Following this is a search button. On searching for a term, we populate the results retrieved from the server in a table.




SPORT SEARCH
localhost:8080/InitialSearch/hadoop/sports
Apps
The HackerRank Int...
The Coding Intervie...
The Logistic Regres...
https://www.awsed...
Interviewing at Am...
(1) Amazon SDE IN...
Move Zeroes - Leet...
SPORTS SEARCH
sports
Hadoop Indexing
SEARCH
Group13
@SuperSportBlitz:The PGA Tour released a statement announcing that South African golfer Victor Lange has tested positive for coronavirus
Link
@SCInterBot: Puta madre use un sports bra por mas de 24 horas seguidas si antes no tenia boobs no sé que tengo ahora
Link
@rsmithfoot:Football continuing to show that it plays a huge part in society with Liverpool players keeping the food bank initiatives running, Roman and G Nev giving hotels over to NHS and other players donating millions. Best sport in the world
Link
@prasker:You're good at sports !! I had no proficiency in any sport
Link
@Mampela_MTO:Orange Thursday to you all. Our 80% cotton & 20 elastine leggings enables you to stretch as much as you want. Wear it for a town stroll or sports
Link
@aaronJ:There is no sports anywhere and you lot are stressing over soaps??? yeh mostly no sportsKmt
Link

Results:


Lucene
Hadoop
2000
1500
1000
500
0
KobeBryant
Football
Baseball
NFL
Jordan
Query

Problems faced:

1. Search delay, due to repeated load of inverted index, in each iteration
   -Solution: Inverted Index is loaded in the initial load of the server
2. Space Complexity
   -Solution: MongoDB has been used to store the tweets, and generate the documentID. In the list, we stored only documentID and term frequency.
3. In Part A of the project, we made a mistake by storing the crawled tweets in a text file. Doing so, there may be some parsing errors while extracting data.
   -Solution: Stored all the crawled tweets in JSON type file, so that the above situation is rectified.

Future Scope:

1. Real-time indexing, in parallel to crawling
2. Usage of sharding, to improve the read and write throughput

Collaboration Details:

Abhishek Ayachit

- Implemented Tweet Indexing using Lucene
- Implemented Indexing of tweets using Lucene in Java
- Worked on Map-Reduce index generation for Hadoop
- Implemented API's for Lucene and Hadoop
- Helped in designing crawler strategies

Gowtham Tumati

- Implemented Reducer method in Map Reduce for Inverted Index generation
- Designed text analyzer choices
- Learnt the working of Twitter Streaming API
- Helped in visualizing the indexed file
- Worked on the project report

Lovepreet Singh Dhaliwal

- Developed the front end of the website
- Coding for Tweet crawling from the Streaming API
- Researched the basic layout of the search engine
- Implemented Query Search Logic to retrieve top results
- Checked out the optimization procedures for the crawling code

Sudip Bala

- Performed initial system level configurations
- Worked on visualizing the results
- Implemented Multiprocessing
- Helped in Tweet Indexing using Lucene
- Worked on the project report

Teja Vemparala

- Implemented Mapper method in Map Reduce program for Inverted Index generation
- Researched the frameworks and libraries for building the search engine
- Performed necessary analysis and design for Tweet Indexing
- Managed the data stored in MongoDB
- Designed crawler strategies

References:

1. Apache Lucene
2. Stack Overflow - Where Developers Learn, Share, & Build Careers
3. MapReduce Tutorial
4. Twitter Developer
5. Leaflet JS Tutorials
6. Apache Tomcat - Tutorial