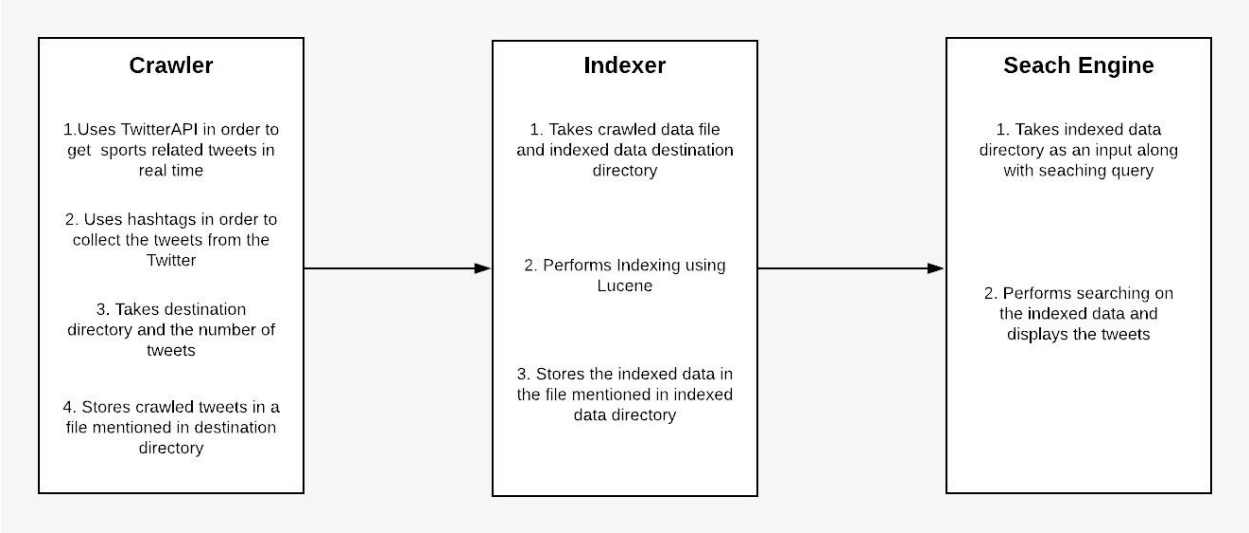
Project Report - Part A
CS 242 : Information Retrieval and Web Search

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. Described as a flowchart, is our project procedure(Fig. below)



(Figure: Project Overview)

The project is divided into two phases, which includes Crawling in the first one and then, Indexing.

Crawling:

We worked on this project with an idea of building a sports-based search. In order to implement this, a lot of tweets related to the latest leagues, players and broadcast corporations are required. A question that turned out soon was the method to extract the tweets. Soon enough, we had two options to do that, one of which was the usage of Twitter's REST API, and the other was the usage of Twitter's Streaming API.

The "Search Engine" that we are building requires the latest information regarding everything, the tweets that we extracted must be recent ones. Therefore, we have decided to go forward with Twitter's Streaming API, since Twitter's Rest API is more useful if we wanted to do some analysis or search on historical data, which is not the current case. Streaming API gives us high volumes of live tweets data as per our request until we want to stop receiving.

We have chosen Python to code for tweet crawling procedure. For fetching the live tweets, we have made use of the "tweepy" library. Tweepy library handles connection, authentication and sessions, which makes it easier in using the Streaming API.

We seeded a lot of hashtags related to sports, channels that broadcast content, teams that participate, latest and implemented the crawler. Some of the hashtags included are '#Soccer', '#Formula1', '#MotoGP', '#Cricket', '#ESPN', '#SuperBowl', '#49ers', '#NBA', '#Lakers', '#CR7', '#Badminton'. The biggest challenge till now was to get those tweets faster. Then, we made the use of multiprocessing in our program, as it utilizes all the processors in the given machine and tries to get the data much faster than earlier. The performance has been improving gradually. A size of 50MB was decided in a single file, wherein the incoming tweets are stored. In a similar way, all the tweets are gathered and processed as a batch of files.

Indexing:

After the crawling process is completed, we will need to perform indexing on all the tweets. On learning about the various indexing architectures, we have learnt that the current architecture for a search engine uses the inverted indexes concept. The procedure inside it is that all the indexes are stored as pairs in which the word is key and document ID is the value. In addition to these, there are ranking parameters such as term frequency. Lucene is a text search engine that helps search over huge amounts of data along with the provision of the necessary API's to provide high performance during the process.

The standard analyzer of Lucene is used to prevent stopwords while index generation. Stopwords are not considered as they have a chance to affect the overall ranking of the documents. We have not performed a case sensitive search, where the same word with a different case can be taken as the same word. Indexing has been done only on two attributes as they are the key. The rest of the attributes have been stored for use in the next part.

List of tweet attributes:

Tweet Attribute	**Description**
HashTag	Keyword for search
Tweet	Content of the tweet
Title	Title of the URL
createdAt	The timestamp of the tweet
CoOrdinates	Tweet Location
URL	Complete URL of the tweet

Implementation Procedure:

Deployment of the entire project:

```
$ bash execute_partA.sh <index-file-directory> <crawled-data-directory>
<number-of-tweets-to-crawl>
#eg bash execute_partA.sh /Users/lavidhaliwal/Desktop/cs242/indexed/
/Users/lavidhaliwal/Desktop/cs242/crawled/ 1000
```

Deployment of Crawler:

```
$ python <file-name> <crawled-data-directory> <number-of-tweets-to-crawl>
#eg: python tweets_crawl.py /Users/lavidhaliwal/Desktop/cs242/crawled/ 1000
```

Building Lucene Index:

```
$ java -jar <jar-file> <index-file-directory> <crawled-data-directory> 1
#eg: java -jar tse-0.0.1-SNAPSHOT-jar-with-dependencies.jar
/Users/lavidhaliwal/Desktop/cs242/indexed/ /Users/lavidhaliwal/Desktop/cs242/crawled/ 1
```

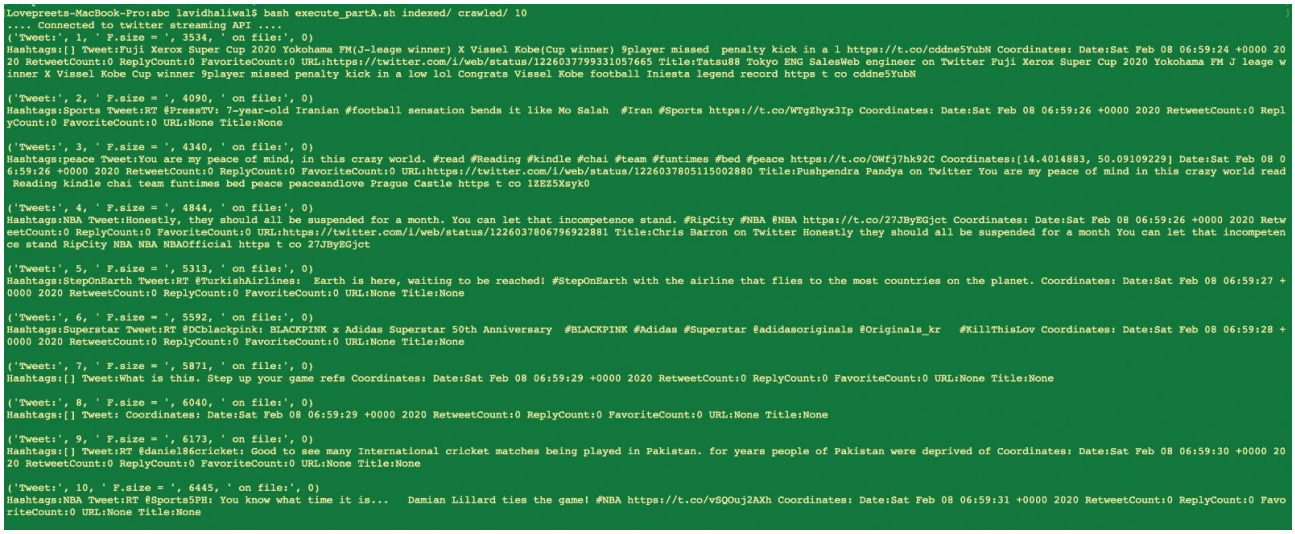
Using the Search Engine:

```
$ java -jar <jar-file> <index-file-directory> 2
#eg: java -jar tse-0.0.1-SNAPSHOT-jar-with-dependencies.jar
/Users/lavidhaliwal/Desktop/cs242/indexed/ 2
```

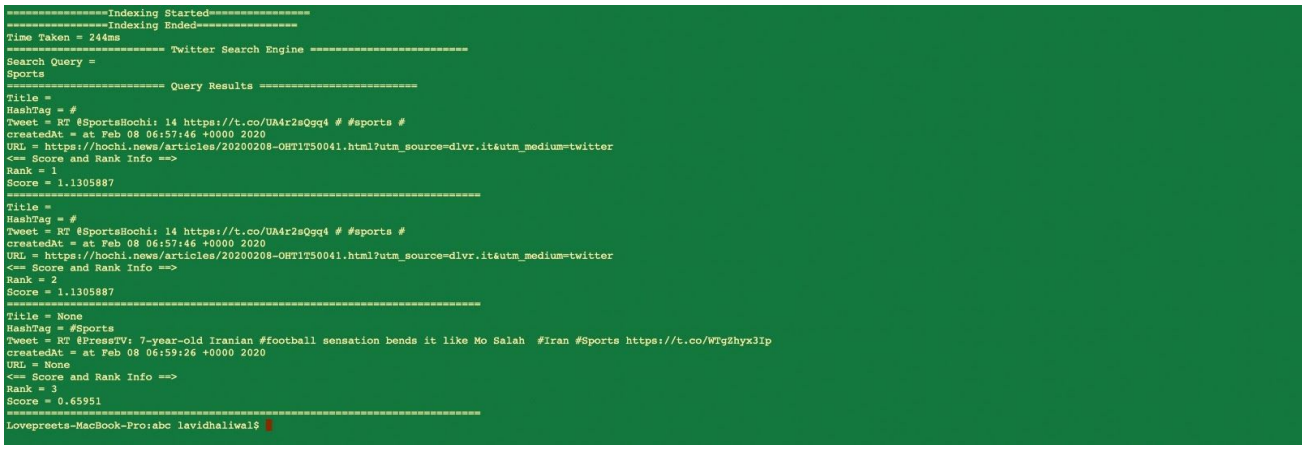
More details regarding this are mentioned in the README file.

Results:

1. Crawling



2. Indexing and Searching



Collaboration Details:

Abhishek Ayachit

- Contributed for Tweet Indexing using Lucene
- Implemented Indexing of tweets using Lucene in Java
- Learnt how Lucene works
- Helped in designing crawler strategies

Gowtham Tumati

- 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

- 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

- Worked on visualizing the indexed file
- Implemented Multiprocessing
- Helped in Tweet Indexing using Lucene
- Worked on the project report

Teja Vemparala

- Performed necessary analysis and design for Tweet Indexing
- Learnt how Lucene works
- Designed crawler strategies
- Helped in deciding the hashtags to seed

References:

1. http://lucene.apache.org
2. http://www.getopt.org/luke
3. https://docs.python.org/2/library/multiprocessing.html
4. http://docs.tweepy.org/en/3.7.0/streaming_how_to.html
5. https://developer.twitter.com/