Course Project

CS 242: Information Retrieval & Web Search

Winter 2020

Build a Search Engine

Part A

Twitter generates thousands and thousands of Gigabyte of data everyday through millions of tweets generated by users. These tweets give an insight on popular moods of the users which is essential insight to many big corporations which use twitter to work on their marketing strategies

We aim to build an indexer through this project which searches tweets scraped from twitter efficiently. We plan on using the Twitter REST API to fetch geospatial tweets so that we can plot them on a map in the next step. Further we use Lucene to index and search on the basis of keywords.

a. **Collaboration Details**

Paranshu singhal

1. Designed crawler strategy
2. Worked on project report

Pranshu Shrivastava

1. Learned how Lucene works.
2. Indexed and searched tweets using lucene

Satyam Prasad

1. Learned how twitter REST API works
2. Co-wrote web crawler for twitter

Shashank Dahiya

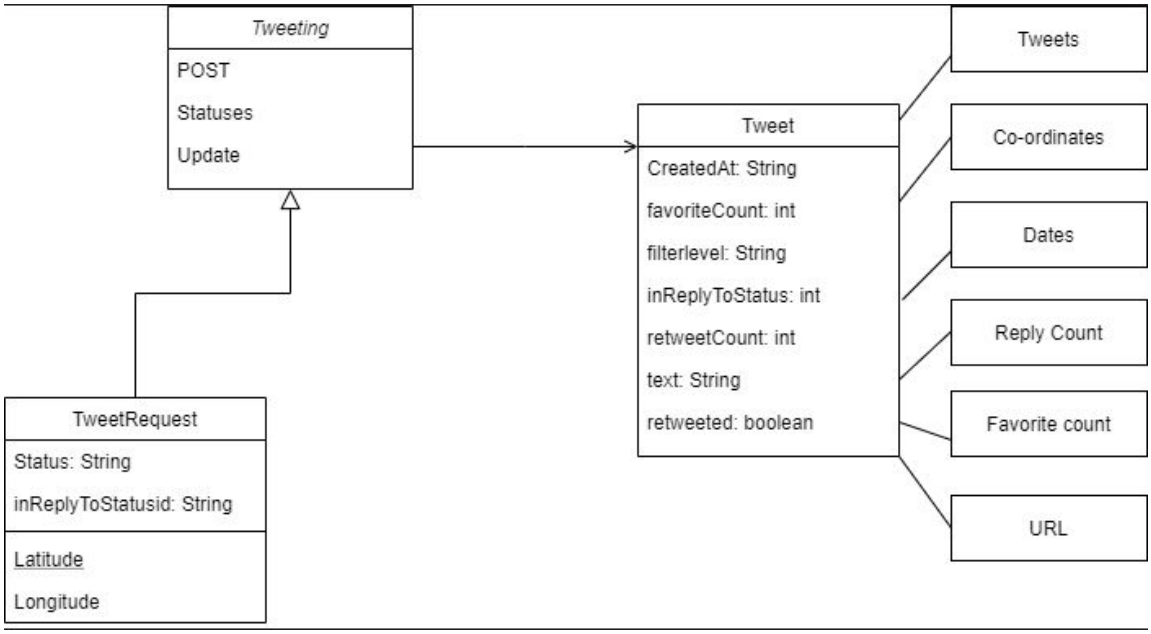
1. Configured Lucene to work on local system
2. Worked on project report

Yash Deshpande

1. Designed text analyzer choices
2. Helped in visualizing the indexed file using matplotlib

b. Overview of the crawling system

1) Architecture



We use the Twitter REST API since it is particularly useful when we have to do analysis on historical data. Since we are not able to get all the tweets in one response we use pagination to retrieve a manifold number of tweets in one response.

c. Overview of lucene indexing strategy
Fields in the lucene index

Tweet Attribute	Description
Tweets	Tweet text
Coordinates	Coordinates of tweet
Date	Date when tweet was posted
ReplyCount	The number of replies to the tweet
FavoriteCount	Number of favorites to the tweet
URL	URL of the tweet
Title	Title of the tweet

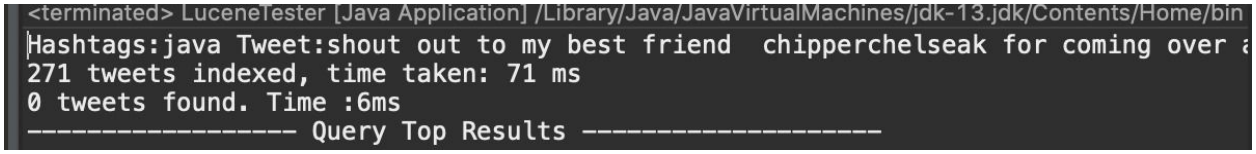
Text Analyzer choices

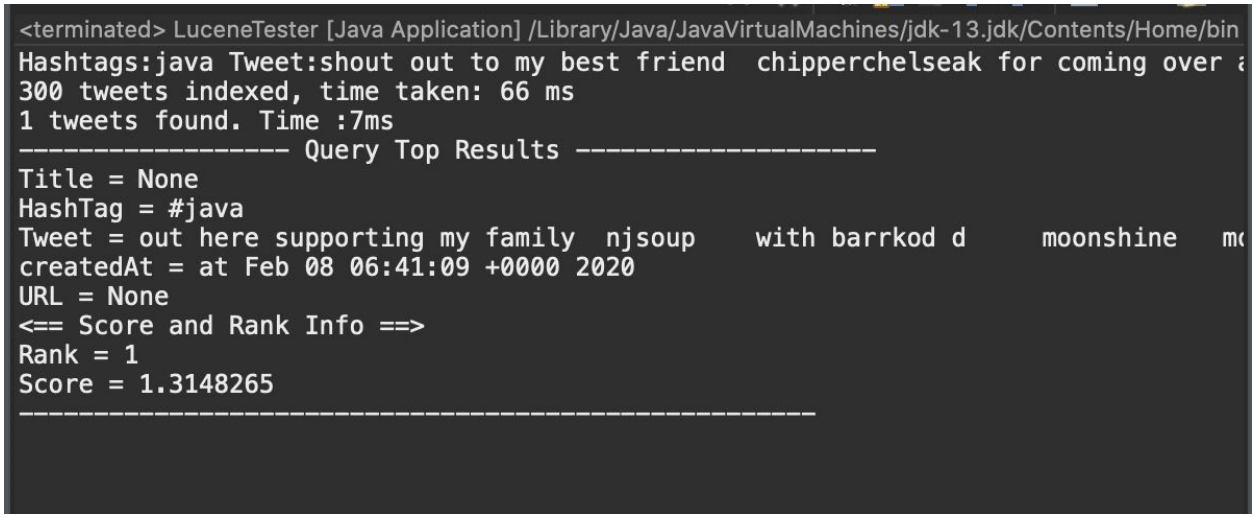
1. Stop word removal : Stopwords are words which do not add anything to the meaning of the sentence such as “the”, “as”, “a” etc. Removal of such stopwords reduces noise in the text which in turn improves indexing speed.
2. Tweet searching is case agnostic, i.e. the case of the characters does not matter.

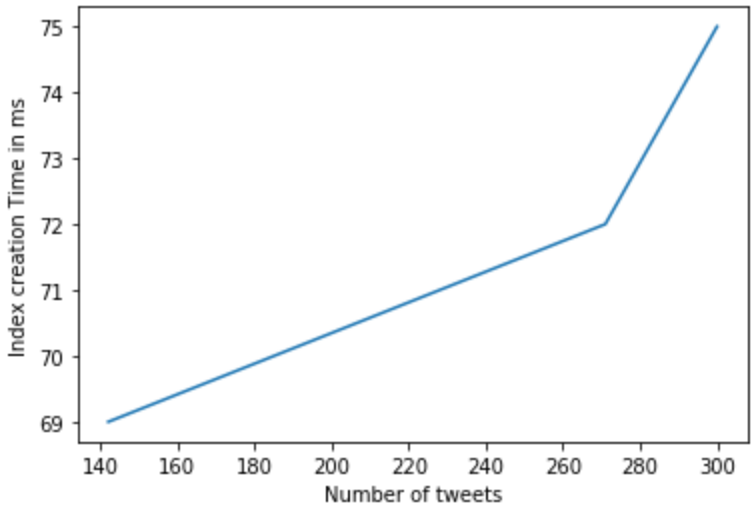
Implementation of Lucene

1. First, we Initialize the QueryParser object that was created using analyzer which contains the index name on which this query is to be run.
2. Then, we create an object of IndexSearcher. Then we create a Lucene directory which points to the location where indexing is to be done. Finally, we initialized the IndexSearcher object that was created with the index directory.
3. We create Query objects by parsing search expressions through QueryParser and perform search by calling the IndexSearcher.search() method.

Run time of the Lucene index creation process







d. Crawler deployment instructions

- Crawler is java based application we need to create the jar file using **mvn compile assembly:single**

  Once the jar is built you can run this command to run the crawler
  java -jar target/crawler-0.0.1-SNAPSHOT-jar-with-dependencies.jar

  {
  "apiURL" : "https://api.twitter.com/1.1/tweets/search/30day/staging.json",
  "upperBoundonHttpRequests" : 3,
  "keyword" : "adidas",
  "maxResults": 100,
  "location" : "ND",
  "bearerToken" :
  "AAAAAAAAAAAAAAAAAAAACI%2BBAEAAAAAbpZyZNfIpKwd3UrFpPTP15K
  K39g%3Dut3PGAzXQT63PIuBzgdkTBvZSGVxkDzKwpTCae8PrNFJ8JMXXc",
  "absolutePathForFilteredData" :
  "/Users/sattu/eclipse-workspace/crawler/prod_filtered_data.json",
  "absolutePathForRawData" :
  "/Users/sattu/eclipse-workspace/crawler/prod_raw_data.json",
  "longitude" : "-74.099200",
  "latitude" : "40.739974"
  }

  Change the absolutePathForFilteredData and absolutePathForRawData

e. Instructions on how to build Lucene Index

- Give the directory names of the input file and output file in LuceneTester class.
- Keep the input file from the crawler in the directory.
- The tweets will be indexed based on the parameters mentioned in the code.