UMKC

Principles Of Bigdata

Phase-II

Submitted By-

Bhuvana Atluri(16186197)

Spandana Surapaneni(16186540)

Surya Prabha Ghanta(12449107)

We have used apache spark to store the collected tweets and spark SQl as a querying language to analyze the data collected using Twitter API. The main aim of our project is to pose interesting queries on the collected data to generate valuable data and present this data to user in the form of graphs and charts

Architecture:

We have used 3-tier architecture for the project. The three tiers of the architecture are

User Interface (Which interacts with the user)

Business logic (runs in back end to operate on data and provide meaningful data for UI)

Data Storage (Apache Spark storage which stores the data for the business logic to operate on)

Implementation

User Interface :

The UI is implemented using HTML, CSS and JavaScript. A JavaScript library Data Driven Document (D3.js) is used to achieve the core part of the UI i.e., visualizations of data. The UI lets user to navigate through different charts driven by data documents generated from Spark.

Querying Data:

# Chart1:

We query the data and extract the top ten languages i.e., the languages in which most of the users tweet. The pie chart represents the percentage of the tweets per language within the ten languages.

Sample Query: val lang=sqlContext.sql("SELECT lang,count(*) as lang_count from MainTable GROUP BY lang ORDER BY lang_count DESC LIMIT 10")

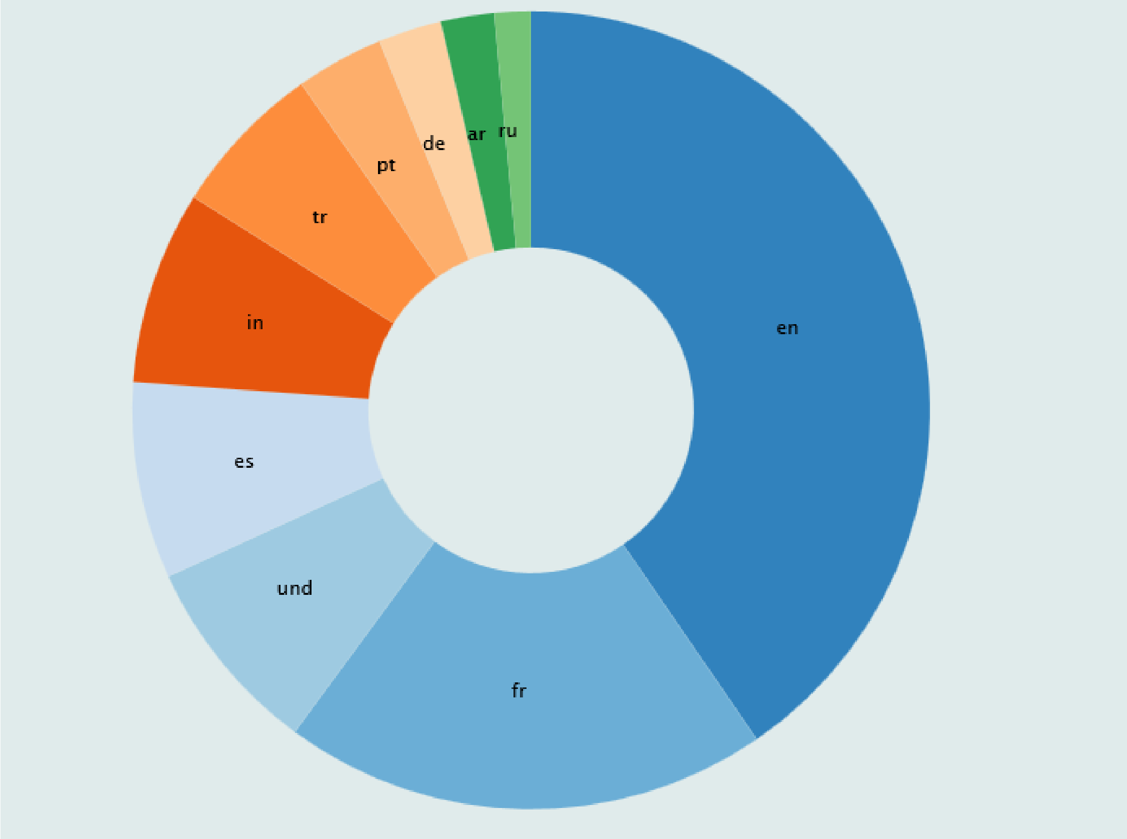
Output:

```
en,9500

fr,4621

und,1898

es,1871

in,1842

tr,1473

pt,835
```

de,606

ar,506

ru,348



# Chart 2:

Number of tweets per location

This emphasis the prominent locations from which the data is being generated.

Sample Query: val timezone = sqlContext.sql("SELECT user.time_zone, count(*) AS location_count FROM Tr GROUP BY user.time_zone ORDER BY location_count DESC LIMIT 10")

Partial Output:

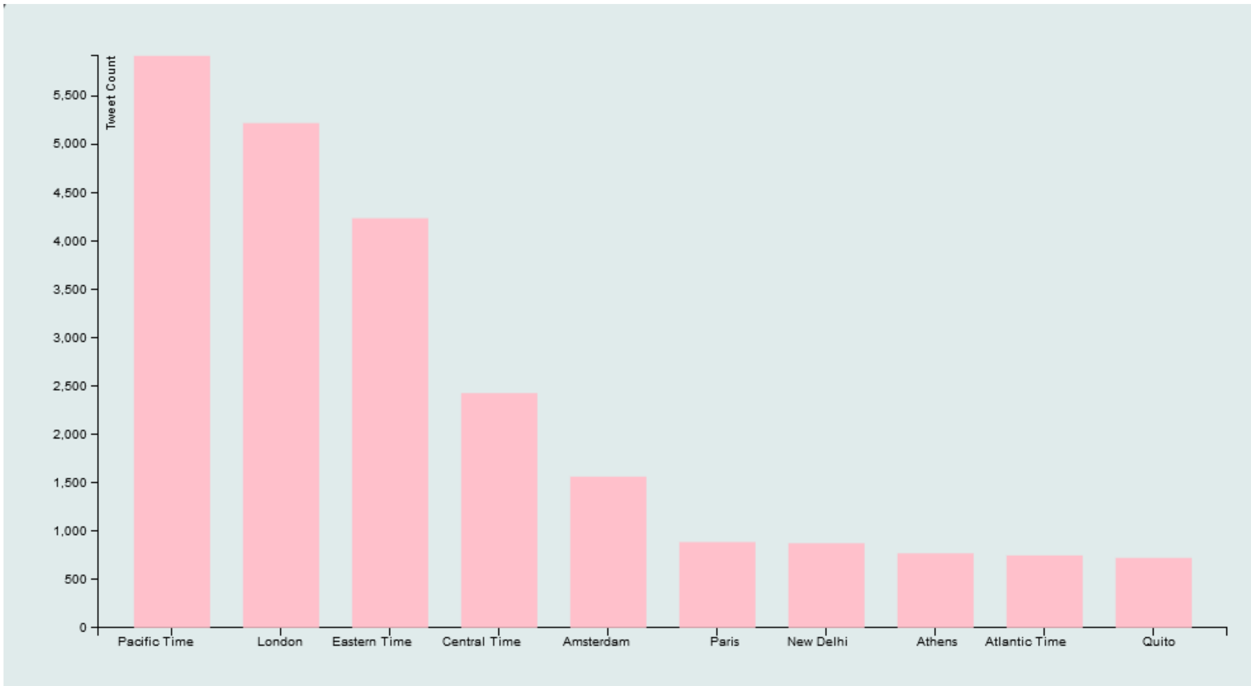
Pacific Time,5913

London,5218

Eastern Time,4234

Central Time,2426

```
Amsterdam,1563

Paris,886

New Delhi,874

Athens,770

Atlantic Time,748

Quito,723
```



# Chart 3

Analyzing the hashtags of tweets to get the most talked about topic among all the tweets collected.

```
 val table2=sqlContext.sql("SELECT entities.hashtags.text as ht from Lang_Refiner where entities.hashtags.text IS NOT NULL)
```

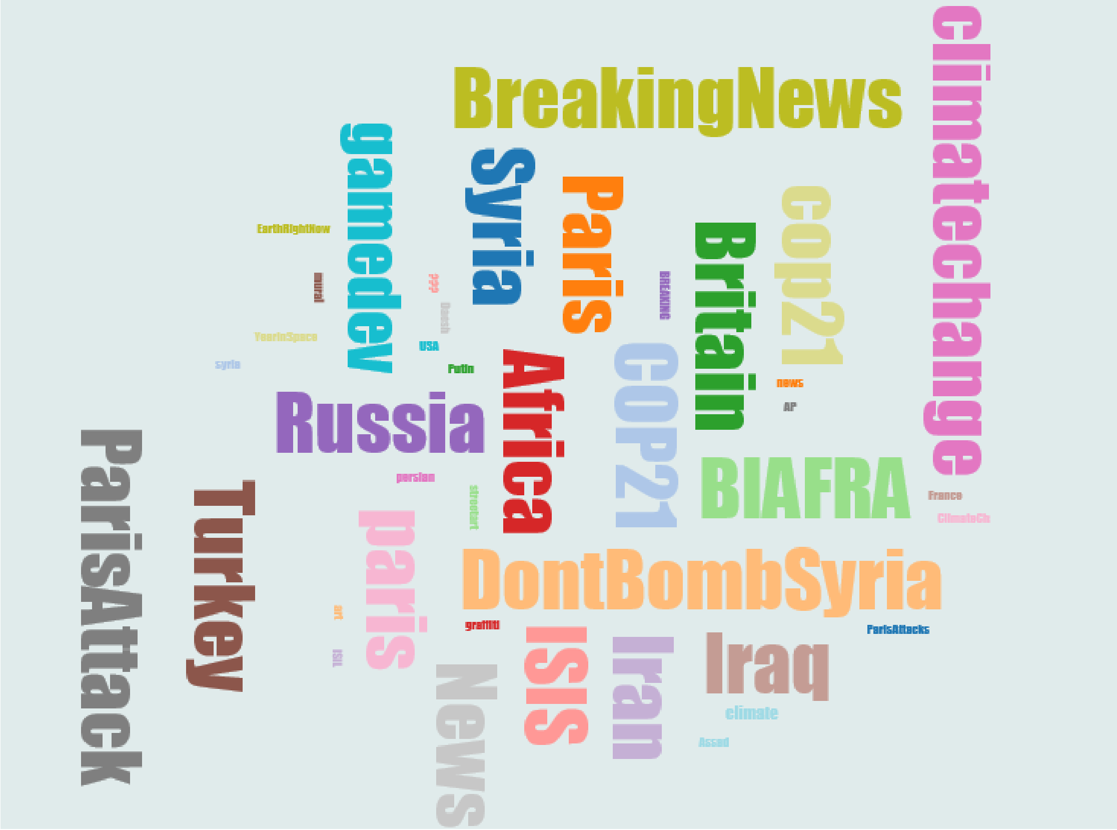
Partial Output:

```
Syria,5181

COP21,3874

Paris,2699

DontBombSyria,1088
```

Britain,1086

BIAFRA,1076

Africa,1075



# Chart 4:

The graph represents the tweets per minute. It highlights the time of the day during which the activity is more

```
val created =sqlContext.sql("SELECT created_at, count(*) AS tc FROM Lang_Refiner GROUP BY created_at ORDER BY created_at")
```

Partial Output:

```
09:28:23,17
```

09:28:24,47

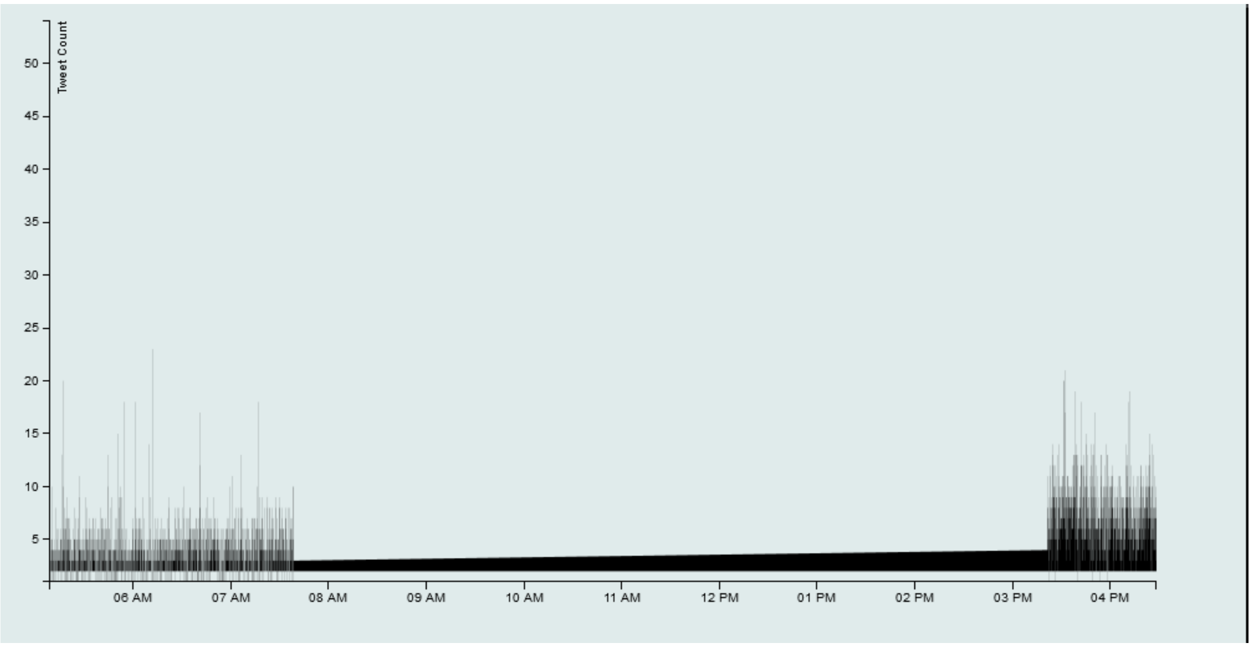
09:28:25,54

09:28:26,66

09:28:27,58

09:28:28,45

09:28:29,53



Query:

To get the popular people on twitter who tweet often and whose follower count is comparatively more

Sample Query:

val join= sqlContext.sql("select NT.u_sn, FC.ft from NT join FC on (NT.u_sn = FC.ust) GROUP BY NT.u_sn,FC.ft ORDER BY FC.ft DESC")

Here we get the popular user's by joining the user table shuffled by highest Number of Tweets(NT) and users table shuffled by maximum Follower Count (FC).

Query:

Getting the users whose friends count is highest from data and limiting the results to top 20

```
val ts=sqlContext.sql("SELECT user.friends_count as rtc,user.screen_name as ust from Lang_Refiner ORDER BY user.friends_count DESC LIMIT 20")
```

Java Helper Classes :

TweetStream.java and TweetUtils.java are two helper classes where TweetStream connects to Twitter API and steams twitter data and TweetUtils take files generated from spark and convert them into suitable formats so that they can further be used by UI to generate data visualizations

Development Environment

UI development is done in text editor and the debugging in case of unexpected output to find out the errors is done using firebug extension for the browser. Eclipse IDE (Integrated Development Environment) is used for Java

Contribution of Team members:

Design and Implementation of UI : Surya Prabha Ghanta

Spark SQL queries and transformations : Bhuvana Atluri

Java Helper Classes: Spandana Surapaneni