Twitter Application : Arvin Sahni
MIDS W205

Introduction:

With the growing popularity of “social communication“, apps like Twitter have become very useful and increasingly popular source for instant and reliable information on trends and sentiments on wide range of topics - eg public response on a recent controversy, views on an impending rate hike, success/failure of a marketing campaign etc.
This project aims to gather information on words and their frequency of usage in the twitter stream in the 1 min window.Some basic serving scripts have been developed to provide a quick peak into information like which are the top 20 words used, which words have been used more than 50 times , what is the frequency of usage of a particular word etc.
Simple bar charts are provided for the top twenty words used as a visual aid.

Architecture:

The project uses the following technologies :
Amazon AWS(EC2),Apache Storm,Postgres,Psycopg2, Python, Streamparse, Tweepy

The overall topology of the application is as follows

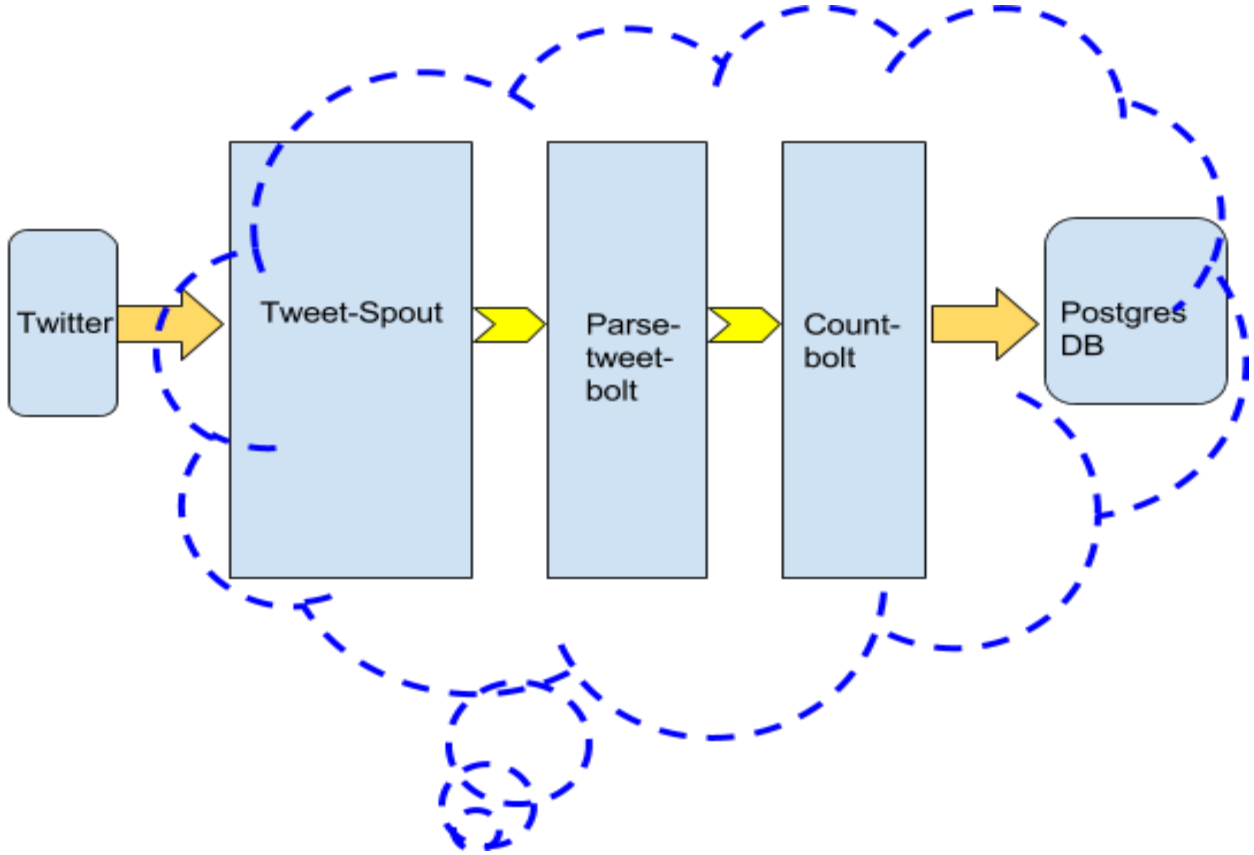

Fig 1: Application Architecture with all components *(the dark blue outline shows that the postgres db and the tweetwordcount project all reside on an EC2 instance in the cloud)*

Postgres DB: **database and table**
The first step is to create a database called tcount. This can be done by executing the file db_create.py
Within this database , a table called tweetwordcount is created using file table_create.py. This table has two columns , word(primary key) and count.
User defined function up_in also needs to be created .Details of the function are in up_in.sql file and its aim is to create an upsert equivalent function

Streamparse project: **tweetwordcount**
Directory/file structure of the project is as follows:

a) tweetwordcount/topologies/tweetwordcount.clj
   This is the clojure file that contains the topologies details
b) tweetwordcount/src/spouts/tweets.py
   This file contains details of how the tweepy library is used to read the twitter feed into the spout .Twitter credentials are updated in this file .In order to avoid “empty queue exception”, the time.sleep is increased from 0.1 to 1 .
c) tweetwordcount/src/bolts/parse.py
   This file uses the tweet passed by the spout to determine valid words by first splitting the tweets and then filtering out retweets, weblinks etc
d) tweetwordcount/src/bolts/wordcount.py
   This file uses psycopg2 to connect to the postgres database and table tcount, tweetwordcount respectively. As python 2.7 does not support upsert function, i wrote my own version of upsert in the file up_in.sql . This function has to be created in the database prior to executing the tweetwordcount topology. It essentially updates the count (increases it by 1) if a row for that particular word already exists otherwise it will insert a row in the table with a count of 1 .

Server Scripts:
Once the topology is run, any of the below server scripts can be used to extract relevant information from the data gathered

a) Finalresults.py
   This script can be used in two ways .
   i) Display all words in the twitter feed with their total count. This is done by issuing command : python finalresults.py
   ii) Display the count of a specific word. This is done by providing the word as a parameter
   python finalresults.py <word>
   The script will display a message stating that the word was “not found in the table” in case the word is not present in the tweetwordcount table else it will display the word and its total count

b) Histogram.py

This script is used to display all words that have counts between two defined numeric parameters. This is done by executing the command as follows
python histogram.py <number1> <number2>
If no or only one parameter is provided the script will throw an error message “check the parameters passed”
If no words are found within the given range, then scripts shows a message “not found in table” else it will display all such words with their counts that satisfy the above mentioned conditions

c) Top_twenty.py

This is used to display a bar chart of the top twenty words with their counts in the twitter feed. The plot is displayed at https://plot.ly/~arvinsahni/
If any credentials are needed, they have been provided as a comment in the top_twenty.py file
Note: the display is not in a sorted order within the top 20 as it’s a known issue with plotly

Possible applications:

1. News analysis : what topics are being discussed most
2. Sentiment analysis : in relation to any impending announcements or recent controversies
3. I have also come across applications that divide words into “happy” and “sad” groups and based on which groups have a higher count, they deem that day/event etc to be happy or sad

   For any analysis, further algorithms would need to be developed on this base model but the applications are immense and in varied fields