W205 Exercise 2

Name: Manish Shah

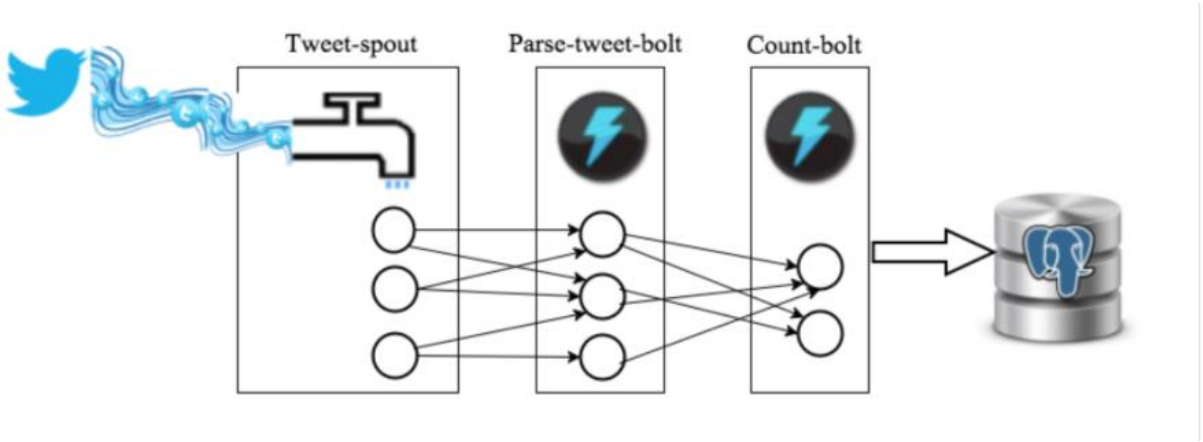
Date: 12/3/2017

The Application

This application subscribes to a tweeter feed and writes the number of occurrences of these words in the database. The tweets containing the words, `["a", "the", "i", "you", "u"]` are tracked and only the tweets in English language are selected.

The application uses Storm technology to subscribe to the tweets and process them. The Tweet spout program subscribes to the tweet, queues them and emits it to the Parse bolt. The parse bolt processes the tweet, excludes retweets and tweets with non-ascii characters and creates a list of valid words and emits it to the wordcount bolt. The wordcount bolt updates the database. The application is implemented in python 2.7.

The application topology is illustrated in the diagram below.



Database

Postgres database is used to store the words and the counts of the words. The name of the database is tcount and it has one table, tweetwordcount. The tweetwordcount table has the following structure.

```
tcount=# \d+ tweetwordcount;
                Table "public.tweetwordcount"
 Column |  Type   | Modifiers | Storage  | Description
--------+---------+-----------+----------+-------------
 word   | text    | not null  | extended |
 count  | integer | not null  | plain    |
Indexes:
    "tweetwordcount_pkey" PRIMARY KEY, btree (word)
Has OIDs: no
```

Topology

The application has one spout, tweet, with 3 instances, one parse bolt with 3 instances and one count bolt with 2 instances. The topology is defined as.

33 lines (31 sloc) 676 Bytes — Raw | Blame | History

```
(ns tweetwordcount
  (:use     [streamparse.specs])
  (:gen-class))

(defn tweetwordcount [options]
   [
    ;; spout configuration
    {"tweet-spout" (python-spout-spec
          options
          "spouts.tweets.Tweets"
          ["tweet"]
          :p 3
          )
    }
    ;; bolt configuration
    {"parse-tweet-bolt" (python-bolt-spec
          options
          {"tweet-spout" :shuffle}
          "bolts.parse.ParseTweet"
          ["word"]
          :p 3
          )
     "count-bolt" (python-bolt-spec
          options
          {"parse-tweet-bolt" ["word"]}
          "bolts.wordcount.WordCounter"
          ["word" "count"]
          :p 2
          )
    }
  ]
)
```

Directory Structure

The directory structure is as follows.

Files/Folders	Description
Create_database.py	Creates the database tcount and table tweetwordcount
Extweetwordcount	The streamparse project folder used by streamparse run
Final_results.py	Queries the database for words
Histogram.py	Queries the database for words with counts >= k and <= l
Main.sh	Main program to run the script
Top20.py	Python program to query top 20 words based on count
Top20_plot.py	Python program to plot the bar chart (tested on windows)
Plot.png	Top 20 bar graph.
Output	Folder for the output files
Screen shots	Screen shots of the project

The topology, spout and bolts are defined in the Extweetwordcount folder.

Run Instructions:

The analysis has been tested on AWS hosted EC2 instance with 100 GB of diskspace, 8 GB of Ram and 2 processors and using Linux operating system.

The AMI used is "UCB W205 Spring 2016 (ami-be0d5fd4)" .

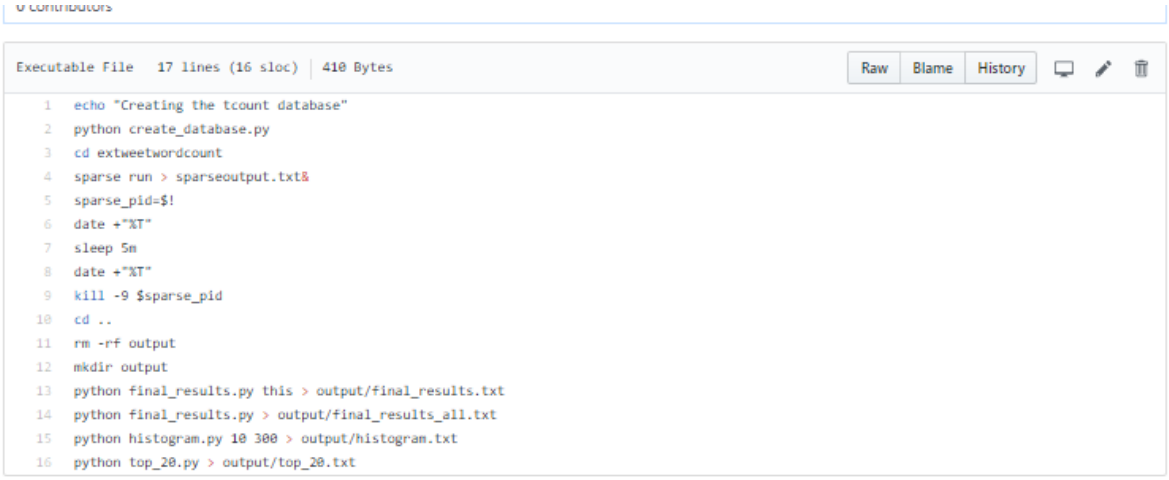
The application needs tweepy and matlibplot libraries to be installed for python.

To run the program, on the command prompt, type,

./main.sh

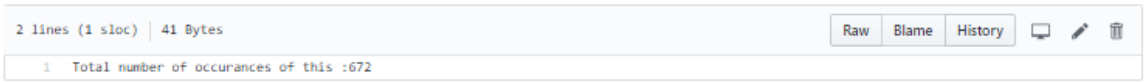
This script will create the database, run the storm application for 5 mins and then run the result scripts.

The main.sh script is shown below.



Output:

1. Output of number of occurrences of word “this”.



2. Output of all words sorted alphabetically with counts.

7996 lines (7995 sloc) | 87.6 KB — Raw | Blame | History

```
! : 26
!! : 10
!!! : 4
!!!! : 2
!Proudly : 2
!SHILL! : 2
$$ : 2
$$$ : 6
```

3. Histogram of all words >= 20 and <= 50

2091 lines (2090 sloc) | 24.4 KB — Raw | Blame | History

```
Oregon's : 50
difficult : 50
Trust : 50
Harry : 50
middle-class : 50
Economics : 50
code : 50
won't : 50
Ocean : 50
Dreams : 50
```

4. Histogram bar graph.