Exercise 2 - Architecture
Yannie Lee

Application Idea

The application retrieves tweets from the Twitter API, counts the occurrences of each words for every tweet, and updates a database to reflect the total number of times every word was tweeted. The application also has three out of the box analysis tools which the user can use to analyze the data in the database.

Each "word" defined in the database is case sensitive, so "hello" and "Hello" are two different entries. The program also includes punctuations as part of the word, so "hello", "hello!", and "Hello," are considered different entries. Also, once the database is created, it accumulates the cumulative sum for tweets while the streamparse application is runnning, so if the user starts and stops the streamparse application several times, the database will include counts of words in tweets from all the sessions.

Application overview

There are 3 main pieces to the application, all in the "Applications" folder within the Exercise_2 file. Each part of the application has it's own file, named accordingly:

- Setup - Database Set Up and Twitter API Testing Scripts
- Extweetcountword- this part of the application pulls Tweets from Twitter and counts unique words, storing results in a database
- Analysis Tools - several functions to help the user analyze the results in the database

It is recommended to run the application in the order listed above when running the application for the first time on an instance. This will ensure the application runs smoothly.

Setup

The scripts in this folder set the instance up to ensure Extweetcountword will run properly. There are two pieces to this: the database setup script, and the Twitter connection test.

The databae set-up script, named "createDatabases.py", creates a connection to postgres to set up a database and table. Simply run this program from command line by navigating to the folder it is in, and running: *python createDatabases.py*. If everything runs smoothly, you will see a "Success! The database and table have been set up" message. The user only needs to run this script once upon setting up an instance.

The Twitter Connection Test ensures that the Twitter credentials the user has are valid. The TwitterConnectionTest folder within the Setup folder contains two files, one with the credentials and the other with the application. To set up and test that you are able to connect to Twitter's API:

1) Update credentials in Twittercredentials.py using the text editor of your choice. There are four things that need to be updated:
    a) consumer_key
    b) consumer_secret
    c) access_token
    d) access_token_secret

    To update, simply replace your keys between the quotes (“”).
2) Once you have updated the Twittercreditials.py file, you can run the hello-stream-twitter application from this folder. To do this, navigate to the TwitterConnectionTest folder and run the script from command line: *python hello-stream-twitter.py* . If successful, you may see some tweets that contain “hello”, although there may be none if there were no tweets containing hello. You will also see the count of tweets that were received over the past 60 seconds, as well as the amount of time the application was running. If the application runs properly, you can now start to run the main application, extweetcountword.

Extweetcountword

The part of the application retrieves tweets from the Twitter API, counts the occurrences of each words for every tweet, and updates a database to reflect the total number of times every word was tweeted.

Specifically, using the tweet spout component utilizes the Tweepy library, to read the live stream of tweets from Twitter. The parse-tweet-bolt parses the tweets, extracts the words from each parsed tweet, and emits the words to the next bolt component (count-bolt) in the topology. The count-bolt counts the number of each word in the received tuples, and updates the counts associated with each word in the tweetwordcount table inside the tcount Postgres database. Figure 1 shows the topology of the application, including the number of instances for each spout and each of the two types of bolts.

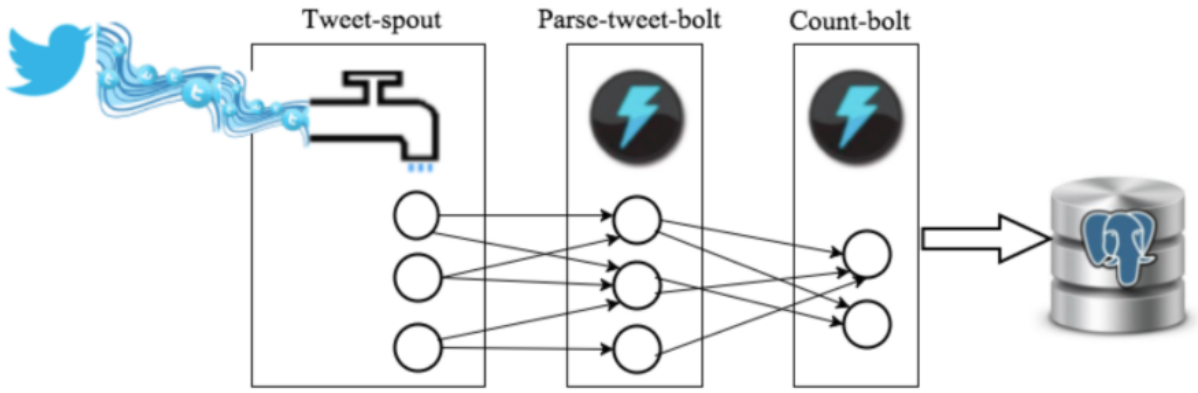

Figure 1: Application Topology

There are two steps to running this application:

- If this is the first time the user is running the extweetcountword application on the instance, the user must first update their Twitter credentials in the tweets.py folder in the "spouts" folder, which is in the "src" folder. Ideally, the user will have tested the credentials using the Twitter Connection Test as described in Setup above. There are four items that need to be updated within the tweets.py file:
    a) consumer_key
    b) consumer_secret
    c) access_token
    d) access_token_secret

  To update, simply replace your keys between the quotes ("").
- Once the user has updated their credentials (this only needs to be done once), the user can run the extweetwordcount from the extweetwordcount directory. Start the program from the command line using the command: *sparse run* . The application runs indefinitely until it is manually stopped by the usual. To stop the program, use this command at any time: *control + c*

Analysis Tools

There are two scripts in the Analysis Tools folder within the Applications directory. Both scripts connect to the database and return the results to the user.Each is described below:

- finalresults.py
    - if called with no arguments, it returns all of the words in the database in alphabetical oder with their count. To call this function, navigate to the AnalysisTools folder and run the following command: *python finalresults.py*
    - If called with an argument, the program will return the count of that word in the database. To call this function and find the number of times "word" was used in a Tweet, navigate to the AnalysisTools folder and run the following command: *python finalresults.py word*
- Histogram.py
    - This script only takes one arguement in the form k1,k2. It returns the words with their respective counts for words with counts greater than or equal to k1 and less than or equal to k2. To call this function to produce a histogram with frequency 3 to 10 inclusive,, navigate to the AnalysisTools folder and run the following command: *python histogram.py 3,10*

Directory and File Structure Overview:

Name	Desciption
Application/	This folder contains the code for the entire application
SetUp/	This folder contains scripts to set up your instance to run the application
createDatabases.py	This script sets up Postgres by creating a database and connectioin
TwitterConnectionTest/	This folder contains files needed to test the Twitter API connection
Twittercredentials.py	This script contains login information for testing the Twitter API connection
hello-stream-twitter.py	This script continas a script that tests the Twitter API Connection
ExTweetWordCount/	This folder contains files needed to run the map/reduce application.
topologies/	This folder contains the topology of the application
tweetwordcount.clj	This script contains the settings for the application's topology
src/	This folder contains all the bolts and spouts used by the application
bolts/	This folder contains all the bolts used by the application
parse.py	This script contains the parse bolt application
wordcount.py	This script contains the wordcount bolt application
spouts/	This folder contains all the spout used by the application
tweets.py	This script contains the tweet spout
Analysis Tools/	This folder contains some tools for basic analysis
finalresults.py	This script returns every entry of the database, or the count for a specific word
histogram.py	This script returns words and their counts for a range of counts
Ignore Me/	Ignore this directory
OtherSubmissions/	This file contains other deliverables not mentioned above, specific files not detailed here
Screenshots/	This file contains screenshots, specific files note detailed here.

Running the application

We have included some tips around how to run the application above. For some short hand notes, refer to the README.txt file within the OtherSubmissions folder.