Exercise 2: w205

Ashton Chevallier, 11/20/2016

Streaming Popular Words on Twitter

Summary:

Twitter is an unlimited source of thoughts n and feelings across the globe, but it's impossible to understand the full picture of all the data that is pulsing through the system. By analyzing live Twitter streams, we can get a better picture of the world around us. This application is a simple way to understand the trending words that come through the twitterverse. We simple pipe the incoming tweets from Twitter via a STORM Spout, parse them and save them to a Postgres database where we can use simple queries to understand more about the thoughts and feelings of the English speaking word (at the time).

- Execute: . db_setup.py
- Switch to tweetwordcount directory
- Execute: sparse run
- Use Control-C to stop input stream when desired.
- Swich to main directory
- Use finalresults.py, histogram.py or custom postgresql commands (Use Database = 'tcount', table = 'tweetwordcount") to analyze incoming tweets.

Directory:

Our directory structure is rather simple. The point is to provide a simple interface to understand trending words on the internet, thus we have provided a relatively simple interface. This document was written as if you have recent downloaded the entire directory off of github (https://github.com/AshtonChevallier/UCB_MIDS_W205_EXCERCISE2).

Screenshots (directory)

Simple subdirectory showing sample output of our system.

Tweetwordcount (directrory)

Main subdirectory where all of the STORM topology, bolts and spouts are contained. To properly run this system, one must run "sparse run" after switching to the tweetwordcount directory (after

initializing the database using the 'db_setup.py' script). The important files are the topologies subdirectory and the src/spouts and src/bolts subdirectories which contain the STORM topology along with the associate spout and bolt files. The following are important files, the remainder are left overs from using "spare quickstart" command.

- Topologies/tweetwordcount.clj
  - Clojure topology file specifying how to import Twitter data using TWEEPY python package.
- Spouts/tweets.py
  - Python code to pipe incoming tweets into our storm topology
- Bolts/wordocount.py
  - Python script to route incoming tweet words into postgressql database.
- Bolts/parse.py
  - Python script to clean incoming tweets of problematic input (Ex: '?$#%') for storage into postgres.

README.md

Simple Markdown file to explain the setup to github users.

Twittercredentials.py/Twittercredentials.pyc

Python files containing the secret codes needed to access the live Twitter stream.

db_setup.py

Essentially our init file. This file is a simple shell script to clear and create a simple postgres database to absorb incoming parsed tweets. Since twitter is a live information source, we felt that the best way to interpret word counts is to analyze them in their context. To meet that goal, we felt starting with a fresh dataset was most appropriate. All word counts are relative to when the program began, not of all time.

finalresults.py

Simple python script that outputs the results of the last twitters stream analysis. There are two different command structures: A) Simply calling "python finalresults.py" will give all of the words in the last twitter input stream in alphabetical order. B) Adding additional arguments to the command will give you the specific counts of that word. For instance calling "python finalresults.py YOU" will give you all the times that YOU was mentioned while the storm spout was running.

Hello-stream-twitter.py

A simple example to code to show off the Tweepy package.

Histogram.py

Another python script that takes 2 numeric arguments separated only by a comma showing all the words have counts in-between the two inputs. For instance the command “python histogram.py 10,50” shows all the words that have a count between 10 and 50 from the last spout execution.

Psycopg-sample.py

Another sample code that shows how the psycopg2 package opertates.

Readme.txt

Similar to markdown file, but provide SPECIFC instruction on how to operate the included code.

Plot.png

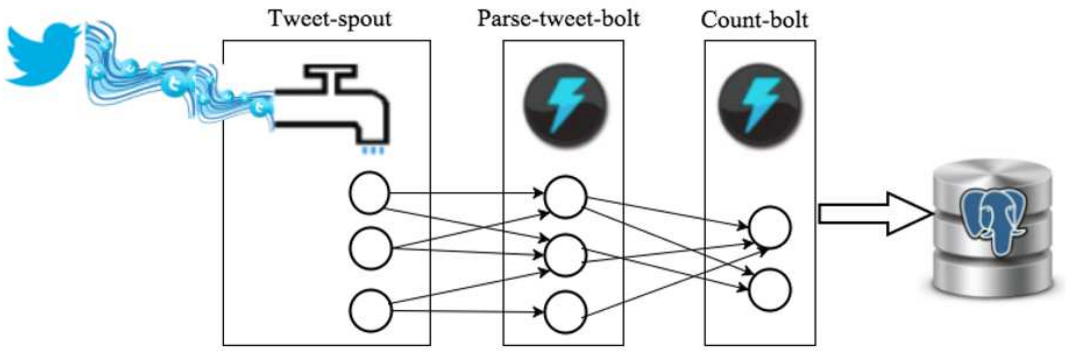
Since there is no elegant way to display data on our EC2 instance, we have provided a screenshot of an R scipt to visualize the top 20 words in our postgres db at the time. The command was given as below:

- select * from tweetwordcount order by count desc LIMIT 20;

The data was then visualized using R studio.

ARCHITECTURE:

Our architecture is relatively simple. Our STORM topology provides a tweets.py spout that acts as a live Twitter feed input to our parse.py and wordcount.py bolts.



In addition to counting the words in the twitter stream, our wordcount bolt provides the instructions to save the incoming tweet data into our postgresql database for further analysis. Once saved into our Tcount database one can use simple SQL commands, or provided python scripts, to analyze the streamed Twitter feed data.