MIDS W205 Exercise 2 Architecture

Objective: To create an application that reads the stream of tweets from the Twitter streaming API, parses them, counts the occurrences of each word in the stream of tweets, and writes the final results back to a postgres database.

Tools used :

- Apache Storm, Amazon EC2, Postgres
- Python 2.7 with streamparse, tweepy, psycopg

Directory in Github repo:

- Extweetwordcount:
    - src/bolts/parse.py: python script to parse tweet, extract words, filter out special symbols and characters
    - src/bolts/wordcount.py: count words and store the results to postgres
    - src/bolts/creatdatabase.py: create database tcount and table in postgres

- src/spouts/tweets.py: connect twitter account, using streamparse, tweepy to stream twitter through API
- topologies/tweetwordcount.clj: application topology.

- Screenshot: some demonstration of the execution of this application.
- README.txt: instructions on how to execute this application
- Finalresults.py: script that can passed word with or without argument
- Histogram.py: script that can return all the words with a total number of occurrence between two integers.

Application topology overview:

Spout: is the scripts that using tweepy to connect to the Twitter streaming API, that pulls tweets and emits them to

the parse bolt. There are 3 threads, and we defined a 100 size queue, so there would be 300 tweets at a given time. Note that when executing the sparse run, there are some "Empty queue exception", is because we use the waiting time of 0.1 seconds under tweets.py script. We can also switch to 0.5 or 1 second here.

Parse bolt: bolt is used to parse the tweets emitted by the spout, and extracts individual words out of the received tweet text. For each tweet, we split the word within the tweet, and then each word would have several transforming steps, including removing the numbers, coverting to lowercase words, deleting hashtag # and other special symbols and characters that are common within a tweet or word. There are 3 parse bolt threads.

Count bolt: We also have count counts the number of words emitted by the tweet-parse bolt, and updates the total counts for each word in the Postgres table. There are 2 bolts thread in this application.

Postgres: We created a database tcount and table

tweetwordcount with two columns word and count.

Results:

Since I don't use twitter a lot and haven't followed a lot of celebrities or news, it doesn't have a lot of very unusual top words selection. For the top 20 tweets, it is not surprising to see common words appear most of the time: the, a, to, I, you, with, this have, etc.