1. Introduction

This document provides an overview of the architecture of the twitter application created for w205 exercise 2. The application consists of a storm topology with a spout that streams tweets from twitter's API, a first bolt that parses words from the tweets, and then a second bolt that records the word counts in a postgres data base. It also includes python scripts for querying the results from the database once it is populated.

2. Directory and File Structure

The application repo contains the files needed for the sparse project and storm topology in /w205/exercise_2/tweetwordcount. The executable file to set up and launch the application is located at /w205/exercise_2/setup_and_launch.sh. The repo also includes the python scripts used to query the populated database at /w205/exercise_2/finalresults.py and /w205/exercise_2/histogram.py. there is an executable file you can use to clean up and delete both the sparse project files and the postgres database located at /w205/exercise_2/cleanup.sh.

3. Storm Architecture

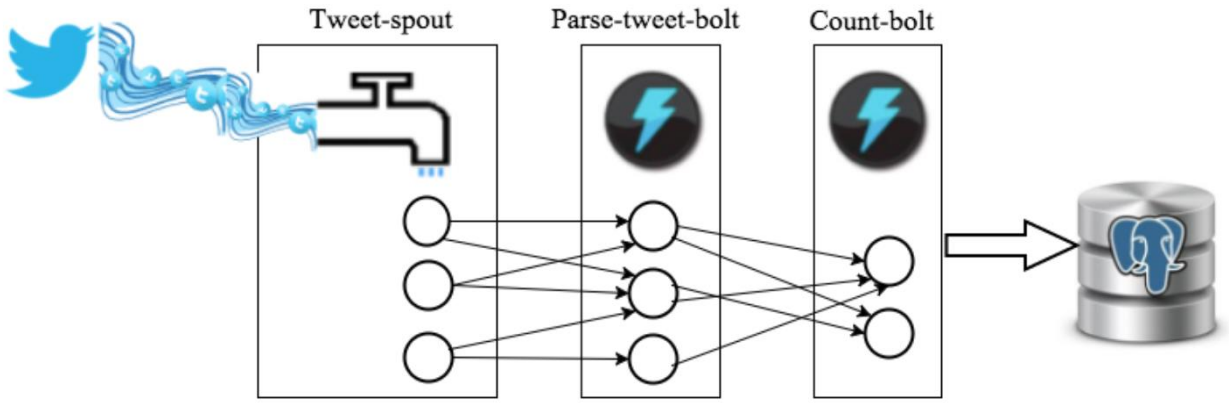

Figure 1: Application Topology

a. Topology:
   The topology consists of one spout and two bolts and is defined by the clojure file at /w205/exercise_2/tweetwordcount/topologies/tweetwordcount.clj
b. Tweet-spout:
   The spout of our Storm topology uses the tweepy package to stream tweets from twitter's api and send them into our system. The code for the spout can be found at /w205/exercise_2/tweetwordcount/src/spouts/tweets.py
c. Parse-tweet-bolt:
   The first bolt parses the tweet for valid words before sending the words to the next bolt. The code for the bolt can be found at /w205/exercise_2/tweetwordcount/src/bolts/parse.py
d. Count-bolt:
   The second bolt counts the incoming words and increments the count by updating a table in a postgres database. It uses psycopg2 to connect to postgres through python. The code for the bolt can be found at /w205/exercise_2/tweetwordcount/src/bolts/wordcount.py

4. Results Output

Besides the files used for the Storm streaming topology the application includes three python scripts that can be used to query the results stored in the postgres database.

a. /w205/exercise_2/finalresults.py
   This script returns the final count for the word passed as an argument. If no argument is passed it returns the counts for every word in the database
b. /w205/exercise_2/histogram.py
   This script returns all words and counts with counts falling in between the min and max values passed as arguments. If no arguments are passed all words are returned and if only one argument is passed, then it is used as the min value with the max set to the highest count in the database
c. /w205/exercise_2/plot_top.py
   This script creates a bar chart showing the top 20 most frequent words and saves the image output file. This script requires the Matplotlib package to be installed in order to run.

5. Execution Instructions

a. Be sure you have both python packages psychopg2.6.2 and Tweepy installed and postgres running on your system before executing
b. Run setup_and_launch.sh to create the postgres database and launch the topology
c. Once the topology has launched and has run for a few minutes ingesting tweets hit Ctrl-C to stop it
d. We now have a populated database we can extract results from using the python scripts finalresults.py and histogram.py
e. Use command "python finalresults.py word" to find the count for a given word
f. Use command "python histogram.py min max" to find all words within the given count range