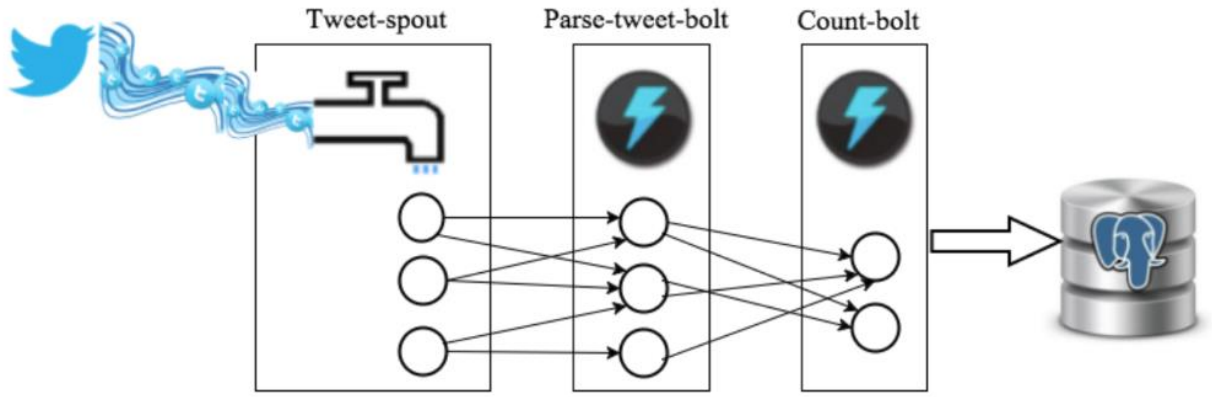
MIDS W205 Spring 2017
Exercise 2
Yifan SUN

Twitter Streaming Application Architecture

The purpose of this document is to outline the architecture of the Twitter streaming application. The document will go over the overall data overflow, dependencies, structure and directories of the repository, and general deployment steps.

I. Overall Dataflow

The application first captures live tweets from Twitter, then processes them in real time to get insights, and finally aggregates the results in a database.



The tweet-spout first stream live tweets using tweepy and Twitter API. Then the tweets are parsed and individual words are extracted via the parse-tweet-bolt. Finally, the words are fed into a counter bolt, where the number of times it appears is counted. The results from the counter bolt then feeds into a postgres SQL database and where further aggregation and analyses are done.

II. Dependencies

The application is designed to run on AWS EC2 machines, specifically AMI "UCB MIDS W205 EX2-FULL". Please make sure to use a security group that's compatible with Hadoop cluster. The user will also have to attach a volume to the machine. System requirements for deployment is listed below, please make sure all these applications are installed/up to date with the correct version before running the application:

- Python 2.7 with streamparse, tweepy, and psycopg2 packages installed
- Apache Storm, Postgres installed and configured

The user will also have to have a twitter account, and obtain their own customer key, consumer secret key, access token, and access token secret in order to run the application successfully. Detailed instructions are located in the README.md file in the repository.

III. File and Directories Structure

- README.md – Instructions on how to run the application
- DB_Setup/setup_DB.py – Creates a new database in PostgresSQL and tweetwordcount table which will be populated while running the application
- extweetwordcount – Streamparse pipeline for the application
    - src/spouts/tweepy.py – Tweet stream script, using tweepy via Twitter API
    - src/bolts/parse.py – Parses and splits tweets into individual words, also gets rid of punctuations and symbols in the tweet
    - src/bolts/wordcount.py – Take the individual words from parse.py as input and creates a tuple pair (word, count) for every new word and updates the existing tuple pair for exiting words. The results is then fed into the tweetwordcount table in the tcount database
    - topologies/tweetwordcount.clj – Storm topology to execute the spouts and bolts. This particular topology uses 3 threads for spouts, 3 for parse bolt, and 2 for counter bolt
- serving_scripts – scripts that aggregate and display results
    - finalresults.py – displays word count for one word or all words in the tweet stream
    - histogram.py – displays words and their counts in a specified interval
- Snapshots – screenshots used in the README file
- Other codes in the folder are sample codes from the W205 repository

IV. Deployment

1. Launch a fresh AMI in AWS, attach volume, and install and configure software and packages as necessary – see dependencies.
2. Run DB_setup/setup_DB.py to set up the database and table needed to store results
3. Launch the storm topology using sparse run
4. Use serving scripts to extract results as needed
5. Detailed instructions and snapshots are located in the README.md file