

## Tweetcounter Architecture

### Table of Contents

Application Purpose .....	2
Operating Instructions (README) .....	3
Directory and File Structure .....	4
Dependencies .....	4
Description of Application Architecture.....	5
Examples.....	6

## Application Purpose

This application streams tweets real-time using Apache Storm into a series of processing programs (called bolts) that count each appearance of every word in a given time frame. The words and their counts are stored in a Postgres table. The application filters out hashtags, retweet notation (RT), URLs, and other punctuation. For example, if the first tweet to come through were:

“Q2 What session was your favorite session from [#EdmodoCon17](#) and why?”

The database *tcount* would contain the following:

<u>word</u>	<u>count</u>
q2	1
what	1
session	2
was	1

(... and so on)

The second part of the application retrieves the information in the database for viewing. The user can see the count of any particular word (and see if it occurred at all) or see the counts of all the words that occurred. For example:

```
$ python finalresults.py world
Total number of occurrences of "world": 14
```

```
$ python finalresults.py
```

```
...
  women: 3
    won: 2
wondered: 2
wonderful: 1
...
(and so on)
```

The user can also see all the words that appeared within a range of counts:

```
$ python histogram.py 15,17
his: 17
them: 17
what: 17
best: 16
day: 16
they: 15
want: 15
```

## Operating Instructions (README)

### SETUP

- ```
-----
```
1. Launch an ec2 instance using the following AMI:  
 AMI Name: UCB MIDS W205 EX2-FULL  
 AMI ID: ami-d4dd4ec3
  2. Attach a volume to your instance, and mount it at /data/  
 (run the following if not already set up:  
 wget https://s3.amazonaws.com/ucbdatasciencew205/setup\_ucb\_complete\_plus\_postgres.sh  
 chmod +x ./setup\_ucb\_complete\_plus\_postgres.sh  
 ./setup\_ucb\_complete\_plus\_postgres.sh <\*the device path\*>)
  3. Install psycopg by running:  
 \$ pip install psycopg2==2.6.2
  4. Install tweepy by running:  
 \$ pip install tweepy
  5. Start postgres by running  
 \$ /data/start\_postgres.sh
  6. Log in to user w205, clone the github directory,  
 and navigate to the project  
 \$ su - w205  
 \$ git clone https://yourusername:yourpassword@github.com/ ...  
 cameronbell75/w205\_2017\_summer.git  
 \$ cd w205\_2017\_summer/exercise\_2
  7. Initialize the postgres database by running  
 \$ python initialize\_db.py

### EXECUTION

- ```
-----
```
1. Navigate to the application directory:  
 \$ cd extweetwordcount
  2. Run the twitter application using:  
 \$ sparse run
  3. Cancel the program with ctrl+C after you have  
 generated enough data.

### ANALYSIS

- ```
-----
```
1. Change directory to w205\_2017\_summer/exercise\_2:  
 \$ cd ..
  2. Run finalresults.py with or without a parameter:  
 \$ python finalresults.py  
 \$ python finalresults.py with  
 If you want to query a word with an apostrophe,  
 wrap it in double quotes:  
 \$ python finalresults.py "you're"
  3. Run histogram.py like this:  
 \$ python histogram.py 8,10

It will output words (and their counts) that appeared 8 <= count <= 10.

## Directory and File Structure

### Exercise 2

README.txt  
initialize\_db.py  
finalresults.py  
histogram.py  
Plot.png  
Architecture.pdf

#### screenshots

Initialize\_Database.PNG  
Stream\_Running.PNG  
finalresults.PNG  
histogram\_program.PNG

#### extweetwordcount

.gitignore  
README.md  
config.json  
fabfile.py  
project.clj  
tasks.py

#### src

##### spouts

\_\_init\_\_.py  
tweets.py

##### bolts

\_\_init\_\_.py  
parse.py  
wordcount.py

#### topologies

tweetwordcount.clj

#### virtualenvs

wordcount.txt

## Dependencies

- Postgres must be running
- psycopg2 version 2.6.2
- tweepy must be installed

## Description of Application Architecture

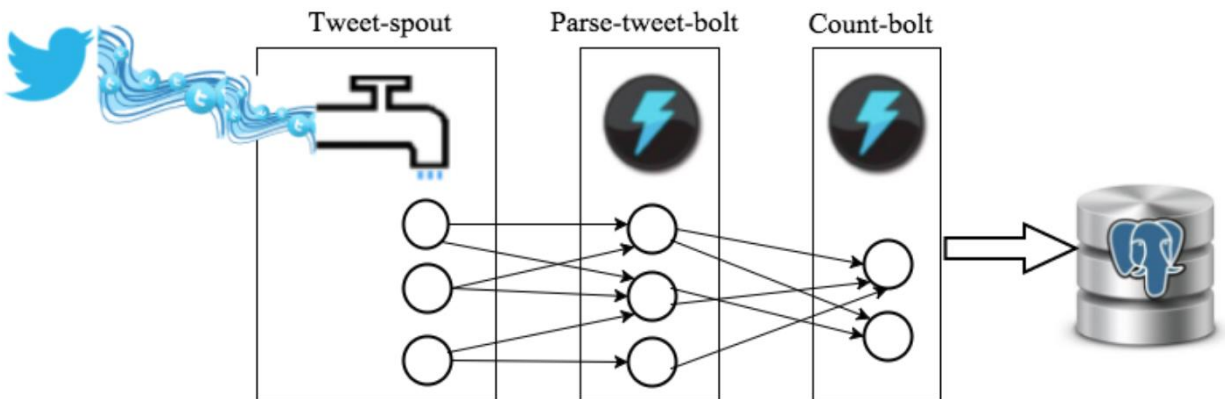
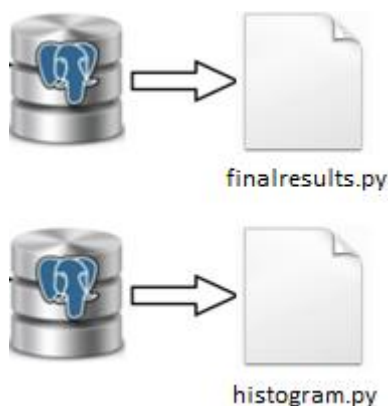


Figure 1: Application Topology

1. As seen above, the application utilizes three spouts that gather incoming data from Twitter.
2. The spouts pass their (shuffled) data to three parse bolts, which filter out unnecessary characters (punctuation, URLs, hashtags, and retweet notation (RT)).
3. The parse bolts pass their (shuffled and filtered) data to two count bolts, which count the occurrences of words that come through.
4. The count bolts finally pass their data (words and counts) into the Postgres database *tcount*, in the table *tweetwordcount*.



5. The python programs `finalresults.py` and `histogram.py` are used to extract information from the Postgres database as described in "Application Purpose" on page 1.

## Examples

```
w205@ip-172-31-14-154:~/w205_2017_summer/exercise_2/extweetwordcount
14034 [Thread-42] INFO backtype.storm.task.ShellBolt - ShellLog pid:16274, name:count-bolt as: 1
14061 [Thread-41] INFO backtype.storm.task.ShellBolt - ShellLog pid:16273, name:count-bolt that: 1
14100 [Thread-24-tweet-spout] INFO backtype.storm.spout.ShellSpout - ShellLog pid:16281, name:tweet-spout Empty queue exception
14114 [Thread-28-tweet-spout] INFO backtype.storm.spout.ShellSpout - ShellLog pid:16271, name:tweet-spout Empty queue exception
14138 [Thread-42] INFO backtype.storm.task.ShellBolt - ShellLog pid:16274, name:count-bolt obama: 1
14146 [Thread-41] INFO backtype.storm.task.ShellBolt - ShellLog pid:16273, name:count-bolt a: 5
14164 [Thread-23-tweet-spout] INFO backtype.storm.spout.ShellSpout - ShellLog pid:16270, name:tweet-spout Empty queue exception
14237 [Thread-41] INFO backtype.storm.task.ShellBolt - ShellLog pid:16273, name:count-bolt of: 1
14240 [Thread-42] INFO backtype.storm.task.ShellBolt - ShellLog pid:16274, name:count-bolt my: 2
14303 [Thread-24-tweet-spout] INFO backtype.storm.spout.ShellSpout - ShellLog pid:16281, name:tweet-spout Empty queue exception
14308 [Thread-42] INFO backtype.storm.task.ShellBolt - ShellLog pid:16274, name:count-bolt money: 1
14316 [Thread-41] INFO backtype.storm.task.ShellBolt - ShellLog pid:16273, name:count-bolt elephant: 1
14317 [Thread-28-tweet-spout] INFO backtype.storm.spout.ShellSpout - ShellLog pid:16271, name:tweet-spout Empty queue exception
14366 [Thread-23-tweet-spout] INFO backtype.storm.spout.ShellSpout - ShellLog pid:16270, name:tweet-spout Empty queue exception
14379 [Thread-42] INFO backtype.storm.task.ShellBolt - ShellLog pid:16274, name:count-bolt goes: 1
14388 [Thread-41] INFO backtype.storm.task.ShellBolt - ShellLog pid:16273, name:count-bolt the: 8
14429 [Thread-41] INFO backtype.storm.task.ShellBolt - ShellLog pid:16273, name:count-bolt summer: 1
14434 [Thread-42] INFO backtype.storm.task.ShellBolt - ShellLog pid:16274, name:count-bolt just: 3
14505 [Thread-24-tweet-spout] INFO backtype.storm.spout.ShellSpout - ShellLog pid:16281, name:tweet-spout Empty queue exception
14519 [Thread-28-tweet-spout] INFO backtype.storm.spout.ShellSpout - ShellLog pid:16271, name:tweet-spout Empty queue exception
14528 [Thread-41] INFO backtype.storm.task.ShellBolt - ShellLog pid:16273, name:count-bolt you: 4
14542 [Thread-42] INFO backtype.storm.task.ShellBolt - ShellLog pid:16274, name:count-bolt presser: 1
14569 [Thread-23-tweet-spout] INFO backtype.storm.spout.ShellSpout - ShellLog pid:16270, name:tweet-spout Empty queue exception
14604 [Thread-42] INFO backtype.storm.task.ShellBolt - ShellLog pid:16274, name:count-bolt please: 1
14628 [Thread-41] INFO backtype.storm.task.ShellBolt - ShellLog pid:16273, name:count-bolt in: 3
14704 [Thread-42] INFO backtype.storm.task.ShellBolt - ShellLog pid:16274, name:count-bolt with: 1
14721 [Thread-41] INFO backtype.storm.task.ShellBolt - ShellLog pid:16273, name:count-bolt a: 6
14771 [Thread-23-tweet-spout] INFO backtype.storm.spout.ShellSpout - ShellLog pid:16270, name:tweet-spout Empty queue exception
14789 [Thread-41] INFO backtype.storm.task.ShellBolt - ShellLog pid:16273, name:count-bolt water: 1
14859 [Thread-42] INFO backtype.storm.task.ShellBolt - ShellLog pid:16274, name:count-bolt cutest: 1
14883 [Thread-41] INFO backtype.storm.task.ShellBolt - ShellLog pid:16273, name:count-bolt trough: 1
14951 [Thread-42] INFO backtype.storm.task.ShellBolt - ShellLog pid:16274, name:count-bolt girl: 1
14963 [Thread-41] INFO backtype.storm.task.ShellBolt - ShellLog pid:16273, name:count-bolt and: 1
14973 [Thread-23-tweet-spout] INFO backtype.storm.spout.ShellSpout - ShellLog pid:16270, name:tweet-spout Empty queue exception
15040 [Thread-42] INFO backtype.storm.task.ShellBolt - ShellLog pid:16274, name:count-bolt ever: 1
15053 [Thread-41] INFO backtype.storm.task.ShellBolt - ShellLog pid:16273, name:count-bolt lions: 1
15097 [Thread-24-tweet-spout] INFO backtype.storm.spout.ShellSpout - ShellLog pid:16281, name:tweet-spout Empty queue exception
15113 [Thread-41] INFO backtype.storm.task.ShellBolt - ShellLog pid:16273, name:count-bolt a: 7
15113 [Thread-28-tweet-spout] INFO backtype.storm.spout.ShellSpout - ShellLog pid:16271, name:tweet-spout Empty queue exception
15120 [Thread-42] INFO backtype.storm.task.ShellBolt - ShellLog pid:16274, name:count-bolt trump: 1
15176 [Thread-23-tweet-spout] INFO backtype.storm.spout.ShellSpout - ShellLog pid:16270, name:tweet-spout Empty queue exception
15186 [Thread-41] INFO backtype.storm.task.ShellBolt - ShellLog pid:16273, name:count-bolt i: 7
15200 [Thread-42] INFO backtype.storm.task.ShellBolt - ShellLog pid:16274, name:count-bolt zombie: 1
15259 [Thread-41] INFO backtype.storm.task.ShellBolt - ShellLog pid:16273, name:count-bolt i: 8
```

Application running – counting words in streaming tweets

```
w205@ip-172-31-14-154:~/w205_2017_summer/exercise_2
[w205@ip-172-31-14-154 exercise_2]$ python histogram.py
Please input exactly two parameters: k1,k2
The program will return the following:

SELECT word, count
FROM tweetwordcount
WHERE count >= k1 AND count <= k2;

[w205@ip-172-31-14-154 exercise_2]$ python histogram.py a,b
Please enter valid integers as arguments.

[w205@ip-172-31-14-154 exercise_2]$ python histogram.py 8,3
Please enter valid numbers.
The first number must be less than or equal to the second number.

[w205@ip-172-31-14-154 exercise_2]$ python histogram.py 15,17
his:          17
them:         17
what:         17
best:         16
day:          16
got:          16
in:           16
want:         16
they:         15
[w205@ip-172-31-14-154 exercise_2]$
```

Operation of the histogram.py program

```
w205@ip-172-31-14-154:~/w205_2017_summer/exercise_2
wind: 1
winfrey: 2
wit: 2
with: 46
woman: 4
women: 3
won: 2
wondered: 2
wonderful: 1
wont: 2
woods: 1
word: 2
worked: 2
works: 6
world: 14
worship: 2
worth: 2
wouldn't: 7
wreck: 2
wretched: 2
writers: 1
writing: 4
wrong: 1
x: 1
xd: 2
xoxo: 1
yankees: 2
yea: 1
years: 9
yess: 2
yesterday: 2
yet: 2
you: 10
you'll: 4
your: 1
[w205@ip-172-31-14-154 exercise_2]$ python finalresults.py world
Total number of occurrences of "world": 14
[w205@ip-172-31-14-154 exercise_2]$ python finalresults.py "you'll"
Total number of occurrences of "you'll": 4
[w205@ip-172-31-14-154 exercise_2]$
```

Program finalresults.py running – query by word





Top 20 words in streaming capture