Cameron Bell
W205, section 5
August 4, 2017

Tweetcounter Architecture

Table of Contents

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:

word	count
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

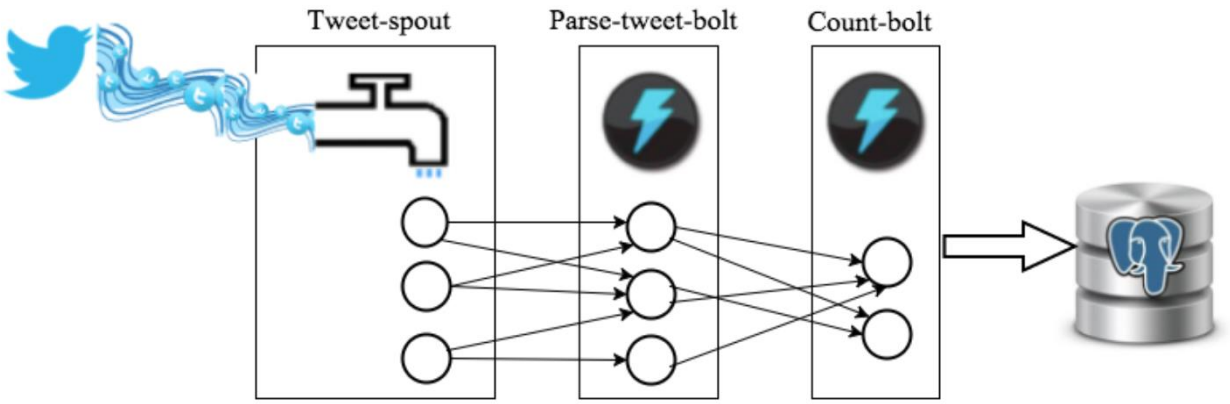
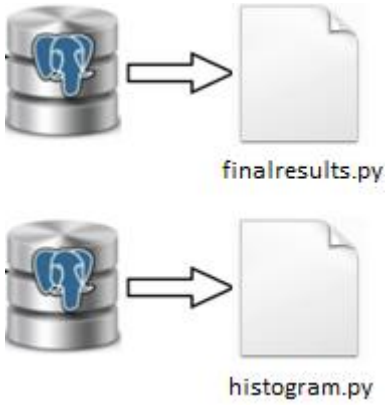


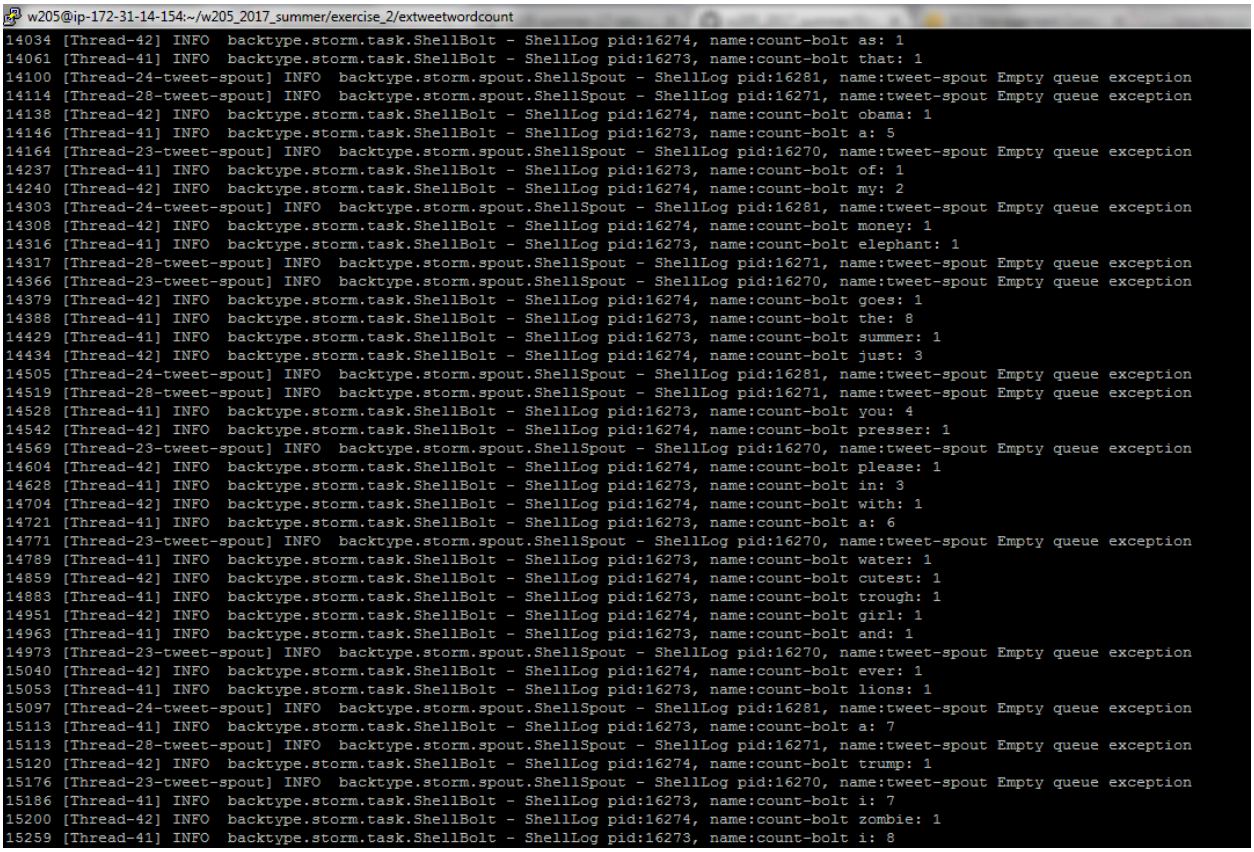
Figure 1: Application Topology

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*.

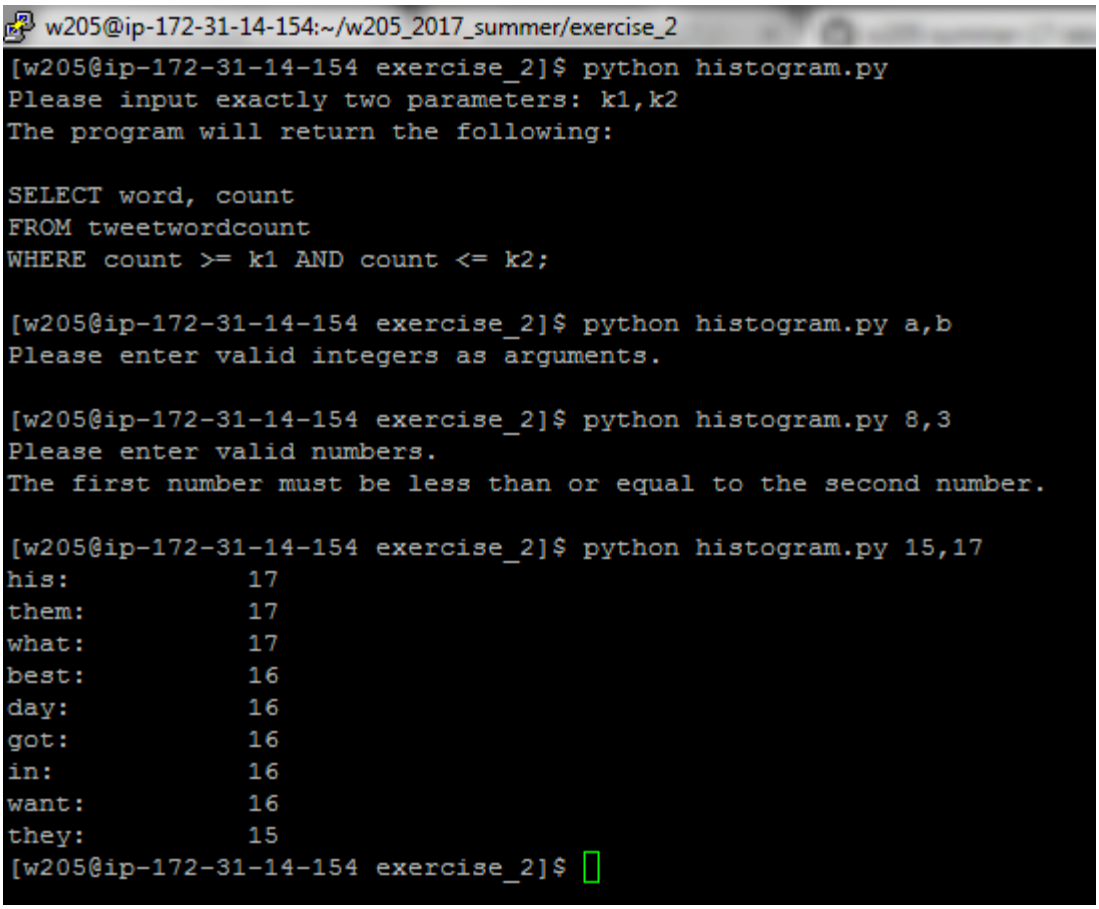


5. The python programs finalresults.py and histogram.py are used to extract information from the Postgres database as described in “Application Purpose” on page 1.

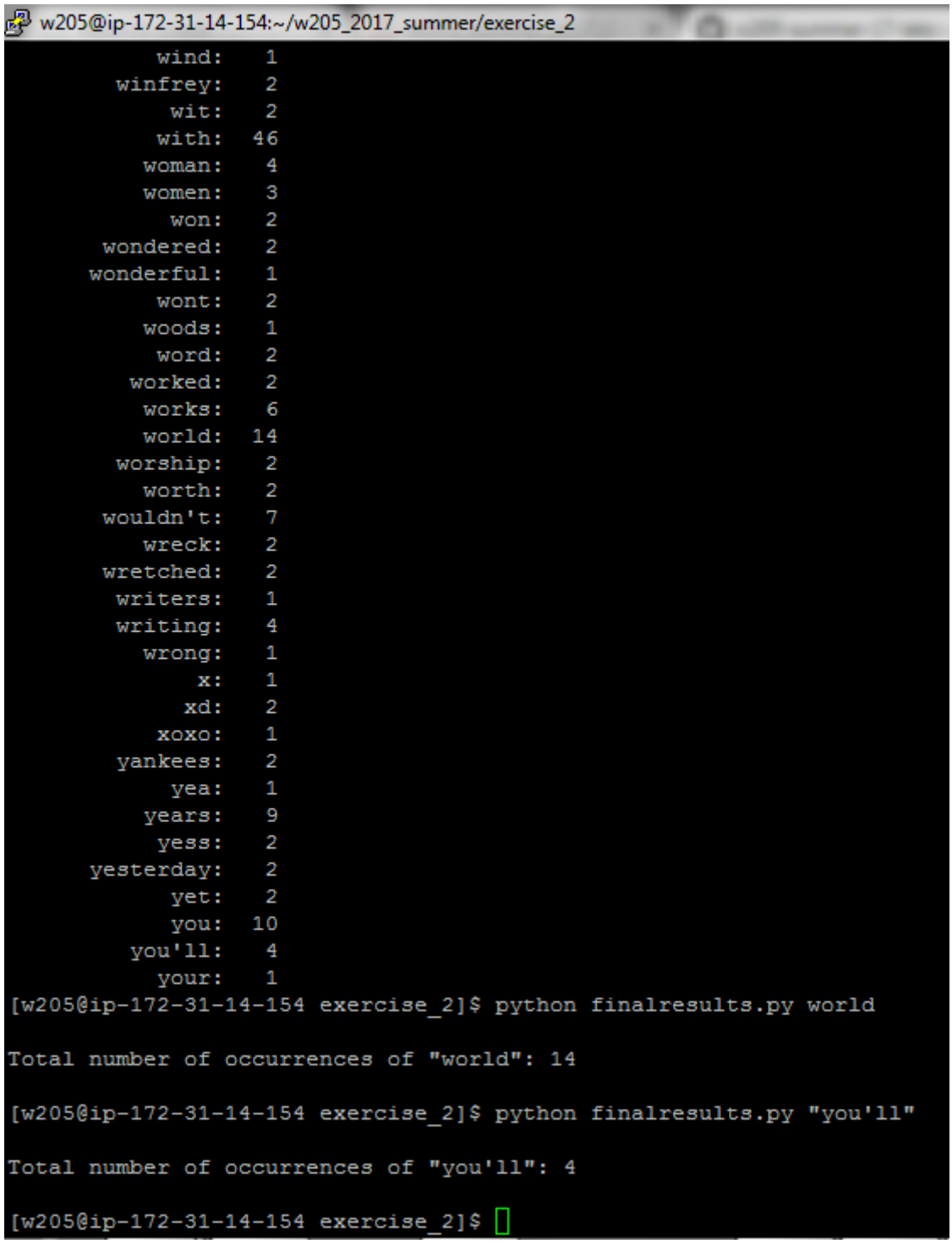
Examples



Application running – counting words in streaming tweets



Operation of the histogram.py program



Program finalresults.py running – query by word

Top 20 words in streaming capture