Directory and file structure

Location	File Name	Description
src/spouts	tweets.py	Receives new tweets according to specified filters
src/bolts	parse.py	Splits sentence into the words and check if each word contains only ascii chars
src/bolts	wordcount.py	Save words in DB. If the word already exists in DB, increases word' counter by 1
topologies	tweetwordcount.py	Indicates a data processing flow
scripts	finalresults.py	Command line script. Works in 2 ways: 1) Show number of occurrences of the provided word. To Run: python finalresults.py hello 2) Prints all words in ascending order and their number of occurrences. To Run: python finalresults.py
scripts	histogram.py	Command line script Prints words in descending order of occurrences within provided range To Run: python histogram.py 3,8
src	access.py	Central place to put DB user\pass and Twitter credentials

Application ideas

1) In case of a big number of streams, in order to decrease DB load, probably, would be better to write simple cache in order to periodically save data to DB, instead of INSERT\UPDATE data per word, or even use in-memory data structure store, such as Redis.

2) *finalresults.py* script would be better to make a message for the case, when table is empty in order to prevent thinking that script does not work (Done).

3) *histogram.py* script would be better to make a message for the case, when table is empty in order to prevent thinking that script does not work (Done).

4) *histogram.py* script would be better to add few more formats of passing arguments, such as histogram.py 3,8 or 3, 8 or 3 8 (Done) and print a message in case if arguments format is incorrect (Done).

5) Create one place\file where database and twitter credentials can be stored (Done).

Description of the architecture

The application is capturing and analyzing live twitter data in real-time, which stores in a database. Figure 1 below shows the overall architecture of the application and its storm's topology. The application reads the live stream of tweets from twitter in the Tweet-spout component. The Parse-tweet-bolt parses the tweets, extracts the words from each parsed tweet and emits the words to the next bolt component (i.e Count-bolt) in the topology. Count-bolt counts the number of each word in the received tuples and updates the counts associated with each words in the *Tweetwordcount* table inside the *Tcount* database. *Tcount* is a PostgreSQL Database.

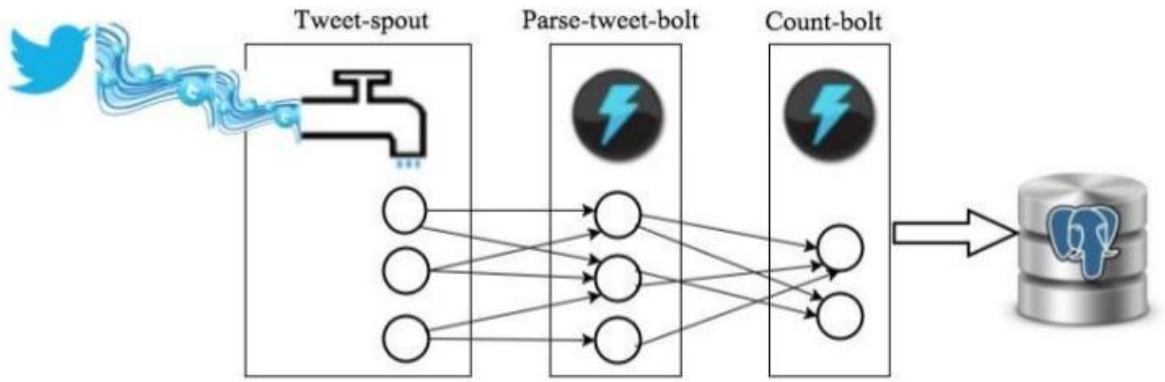

Figure 1. Application Topology

Dependencies

App\Library	Version	Docs
Apache Storm	1.0.2	http://storm.apache.org/releases/1.0.2/index.html
PostgreSQL	9.6	https://www.postgresql.org/docs/9.6/static/index.html
Java	8	

Python	3.5.2	https://docs.python.org/3/
Psycopg2	2.6.2	http://pythonhosted.org/psycopg2/
Tweepy	3.5.0	http://tweepy.readthedocs.io/en/v3.5.0/
Streamparse	3.2.0	http://streamparse.readthedocs.io/en/v3.2.0/

Instalation and Run Application

1) $ sudo apt-get install python3.5-venv
2) $ pyvenv-3.5 ~/project/

3) Install Java
   $ sudo add-apt-repository ppa:webupd8team/java
   $ sudo apt-get update
   $ sudo apt-get install oracle-java8-installer

4) Install PostgreSQL >= 9.5
   $ sudo apt-get install libpq-dev
   $ sudo apt-get install postgresql postgresql-contrib

5) Install Apache Storm >= 1.0.0
   $ wget http://apache.ip-connect.vn.ua/storm/apache-storm-1.0.2/apache-storm-1.0.2.tar.gz -P ~/project/
   $ tar -xvzf ~/project/apache-storm-1.0.2.tar.gz
   $ ~/project/apache-storm-1.0.2/bin

   $ sudo nano /etc/environment
   to the end of PATH add the next line (change USER_NAME to the one, where you store Storm)
   :/home/USER_NAME/project/apache-storm-1.0.2/bin

6) Install Lein
   $ wget https://raw.githubusercontent.com/technomancy/leiningen/stable/bin/lein -P ~/project/bin/

$ chmod a+x ~/project/bin/lein
$ ~/project/bin/lein

7) $ cd ~/project/
9) $ cd ex2tweetwordcount/
10) $ source /home/ubuntu/project/bin/activate
11) $ export LC_ALL="en_US.UTF-8"
12) $ pip install -r requirements.txt

13) Create DB 'tcount' and a new user 'tcountuser':
$ sudo -u postgres psql
-> CREATE DATABASE tcount;
-> CREATE USER tcountuser WITH password 'TcountPass';
-> GRANT ALL PRIVILEGES ON DATABASE tcount TO tcountuser;
-> \q

14) Create Table 'tweetwordcount' on behave of just created user 'tcountuser':
$ psql -h localhost tcount tcountuser
-> CREATE TABLE tweetwordcount (word TEXT PRIMARY KEY NOT NULL, word_count INT NOT NULL);
-> \q

15) Get Twitter application credentials.
In order to do that it is necessary to:
1. Login to Twitter (https://www.twitter.com/).
2. Visit https://apps.twitter.com and click on "Create New App".
3. Fill in the application name, description, and Website. Ignore Callback URL field.
4. Agree to the terms and agreements and click on "Create your Twitter Application"

Once you have successfully created an application, it should take you to the newly created application. Here you must create access keys for subsequent operations by your application. To do so, use the following procedure:
1. Click on the "Keys and Access Tokens" tab.
2. Click on "Create my Access Token" near the bottom of the page.

Now you have four things (as blurred above):
1. A consumer key that identifies your application.
2. A consumer secret that acts as a "password" for your application.
3. An access token that identifies your authorized access.
4. An access token secret that acts as a "password" for that authorized access.
At any point, you can revoke the access key or regenerated any of these values. To completely disable the application, you must delete the application. This does remove the consumer key, secret, and access tokens from Twitter's system and any program using them will immediately stop working.

16) Enter your DB and Twitter credentials in .../ex2tweetwordcount/access.py
According to the steps #13 and #14 above, ‘db_access’ dictionary must look like:

```
db_access = {
    'database' : 'tcount',
    'user'     : 'tcountuser',
    'password': 'TcountPass',
    'host'     : 'localhost',
```

```
    'port'          : '5432'
}
```

Values of keys of ‘twitter_access’ dictionary must be filled in according to the credentials received in step #15 above.

17) Run application (i.e. start Twitter streaming)

```
$ sparse run
```

18) Run finalresults utils:

```
$ cd ~/project/ex2tweetwordcount/scripts/
$ python3 finalresults.py
or
$ python3 finalresults.py some_word
```

Example:

```
$ python3 finalresults.py hello
```

19) Run histogram script:

```
$ cd ~/project/ex2tweetwordcount/scripts/
$ python3 histogram.py min_range, max_range
```

Example:

```
$ python3 histogram.py 3, 8
```