Twitter word counting application

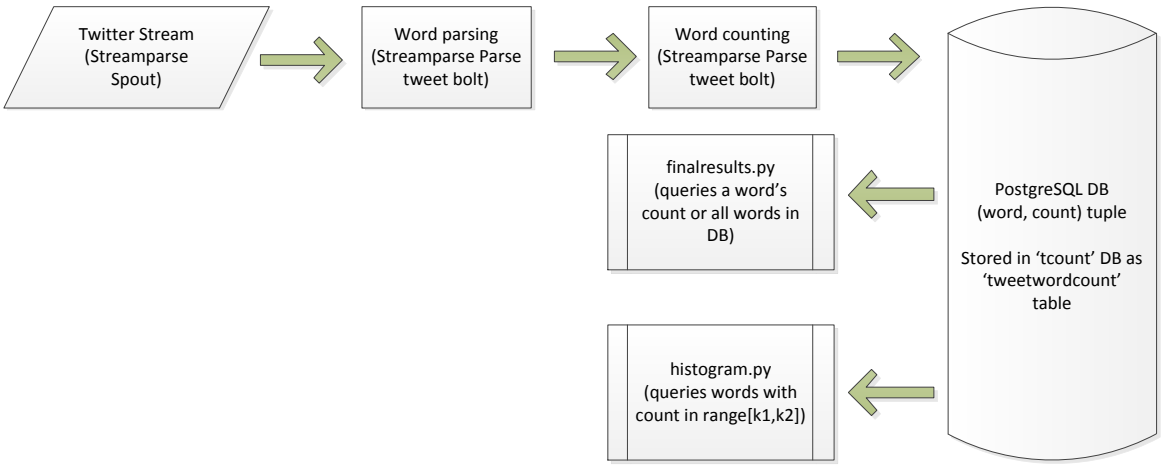
Summary:

This streaming application is based on Twitter livestream processing. It intercepts the live Twitter stream with Streamparse Spout, which feeds the stream to the Streamparse Bolt that parses the words out from the stream. These words, next are then passed on to another bolt that performs aggregation and counts the number of times each word occurs in the stream. This tuple is then pushed to a postgres database in a table 'tweetwordcount'. Two sepratate queries then can be executed to retrieve aggregation statistics:

Outputs:

finalresults.py: query the count of an input word,
or displays all the words currently in the DB and their count
histogram.py: returns the words whose count is in the count range passed as argument

Basic flow execution:



Environment setup:

The application was setup and verified on Amazon AWS EC2 instance. The AMI used was "UCB MIDS W205 EX2-FULL" with ID "ami-d4dd4ec3", which has Hadoop as well as postgreSQL packages installed.

Additionally, the *tweepy* Python package was used to interface with the Twitter API, and the *psycopg2* Python package for interfacing with postgreSQL DB.

Structure:

The following tree shows the main components of the source code. The flow between Streamparse spouts and bolts components is also shown in the aforementioned block diagram.

```
+---Scripts
|     finalresults.py
|     histogram.py
|
\---tweetwordcount
   |  config.json
   |  fabfile.py
   |  project.clj
   |  README.md
   |  tasks.py
   |
   +---src
   |  +---bolts
   |  |     parse.py
   |  |     wordcount.py
   |  |     __init__.py
   |  |
   |  \---spouts
   |        tweets.py
   |        __init__.py
   |
   +---topologies
   |     tweetwordcount.clj
   |
   \---virtualenvs
         tweetwordcount.txt
```

Running the application:

Following are the steps involved for the application deployment on AWS EC2 instance setup; shown with sample code executions:

1. Starting the Twitter stream capture:

```
[root@ip-172-31-86-241 ~]# cd extweetwordcount/
[root@ip-172-31-86-241 extweetwordcount]# pwd
/root/extweetwordcount
[root@ip-172-31-86-241 extweetwordcount]# sparse run
Running tweetwordcount topology...
```

2. Connecting to postgres 'tcount' database' (not required - only for live interaction with DB)

```
[root@ip-172-31-86-241 ~]# psql -U postgres
psql (8.4.20)
Type "help" for help.

postgres=# \c tcount
psql (8.4.20)
You are now connected to database "tcount".
tcount=#
```

3. Running query scripts:
3a1. Retrieving count of a word:

```
[root@ip-172-31-86-241 ~]# python finalresults.py nation

     ( nation , 3 )
```

3a2. Retrieving all words in the DB 'tweetwordcount' (output truncated here)

```
[root@ip-172-31-86-241 ~]# python finalresults.py


( test : 5 )
( Tonga : 1 )
( users : 1 )
( RAMBO : 1 )
( RAJKUMAR : 1 )
( Back : 1 )
( PPNH : 1 )
( chicken : 1 )
( nuggets : 1 )
( ani : 13 )


( regret : 2 )
( girlfriend : 2 )
( last : 1 )
( schedule : 1 )
( KNOW : 1 )
( wrap : 3 )
( HOW : 1 )
( POWERFUL : 1 )
( way : 10 )
( film : 1 )


( mid : 1 )
```

3b. Retrieving words in a certain count range:

```
[root@ip-172-31-86-241 ~]# python histogram.py 20 30


1    >>> now                 : 25
2    >>> her                 : 22
3    >>> not                 : 25
4    >>> but                 : 28
5    >>> from                : 22
6    >>> because             : 28
7    >>> or                  : 28
8    >>> an                  : 21
9    >>> he                  : 21
10   >>> girl                : 21
```
