# Surfstream: Word Counts of Surf Tweets


## Purpose

Surfstream will show you trending words in surfing by analyzing Twitter data in real time. Surfstream uses streamparse, Storm, Postgres, and Python to deliver the end-user valuable summary information about the counts of words in tweets. You can use this awesome app to discover which surfer people are talking about most, how people feel about surfing at different times of the day, and much more!

## How it works

Surfstream initiates a Storm application that pulls tweets from Twitter with a Spout that employs the streamparse technology, in connection with the tweepy package, to send tweet contents to a Bolt. The Spout filters specifically to tweets that contain the word “surf”, because we want to know the latest words associated with surfing. This Bolt splits up the tweet into words and filters out retweets and other non-standard types of tweets. It then removes non-standard characters and send the set of words to a second Bolt, which stores the counts of each word. This second Bolt also adds the word to the Tweetwordcount table in Postgres Tcount, if it hasn’t been added yet. If it has been added, then it updates the “count” field for the word in the database to the current count.

As an example, imagine there are two tweets:

1) “I like to surf”
2) “I do not surf”

After the first tweet, the table Tweetwordcount appears as follows:

| word | Count |
|---|---|
| I | 1 |
| like | 1 |
| to | 1 |
| surf | 1 |

After the second tweet, the table appears as follows:

| word | count |
|---|---|
| I | 2 |
| like | 1 |
| to | 1 |
| surf | 2 |
| do | 1 |
| not | 1 |

# Running the application

## Dependencies

- AMI setup as in Lab 6
- Twitter credentials (https://apps.twitter.com/)
- Postgres
- Storm
- lein (build tool)
- Python packages:
  - tweepy
  - psycopg2
  - redis
  - streamparse
  - matplotlib

## Instructions

1. Navigate to exercise_2 (you should already be in this folder)
2. If you haven't set up postgres yet, run "setup_postgres.sh"
   a. Run "psql -U postgres"
   b. Run "CREATE DATABASE tcount;"
   c. \c tcount to make sure it worked
   d. \dt to show that it is empty
   e. \q to quit
3. Run "create_postgres_table_tweetwordcount.py" to create the table that the tweets will be put into.
4. Run "sparse run" to activate the Storm / streamparse topology, which will look for tweets containing the word "surf", split the tweets by word, and then add/update the counts in the postgres database "Tcount", table "tweetwordcount".
   a. Press CTRL+C to stop that process once you have gotten enough tweets.
5. Analyze results:
   a. To obtain the total number of occurences for a specific word, run “python finalresults.py test_word", where test_word is the word you want to see the counts for.

b. To see a list of the counts for all words, run "python finalresults.py" with no arguments specified.
c. To see a list of words with occurence counts in a certain range, run "python histogram.py 5 10" where 5 and 10 represent the minimum and maximum counts allowed (inclusive).
d. To see a histogram of the top 20 words, run "plot_top_x.py 20". The output will be saved as "Plot.png".

## Architecture