Tweetwordcount Application

By pbarranger

Application Description

Overall

This tool uses Storm to tap into a stream of real-time tweets. Using the Spouts and Bolts topology below the tweets are split up by word and the count of each word is stored in a Postgres databases. The application is setup to add to the database progressively - each new run will add to the existing data. Two python scripts have been included as well to help query the resulting database. Direct interaction with the dataset is also possible through Postgres.

Detailed process

The extweetwordcount application uses tweepy to access the Twitter API with existing credentials. The entire topology is defined in tweetwordcount.clj, showing the working relation between spouts and bolts. Stating explicitly here, the spout (tweets.py) connects to the Twitter API and reads all english tweets with ascii characters. A single string is then pushed to the bolt parse.py. Parse.py divides each string into individual words and scrubs for common non-words such as @, #, and RT found in tweets. The output is then sent to the second bolt, wordcount.py. Wordcount.py counts the frequency of each word as it appears in the stream of tweets. This bolt stores a tuple ["word", "count"] in a Postgres database as well as pushes the updated count to the terminal window.

Application Topology

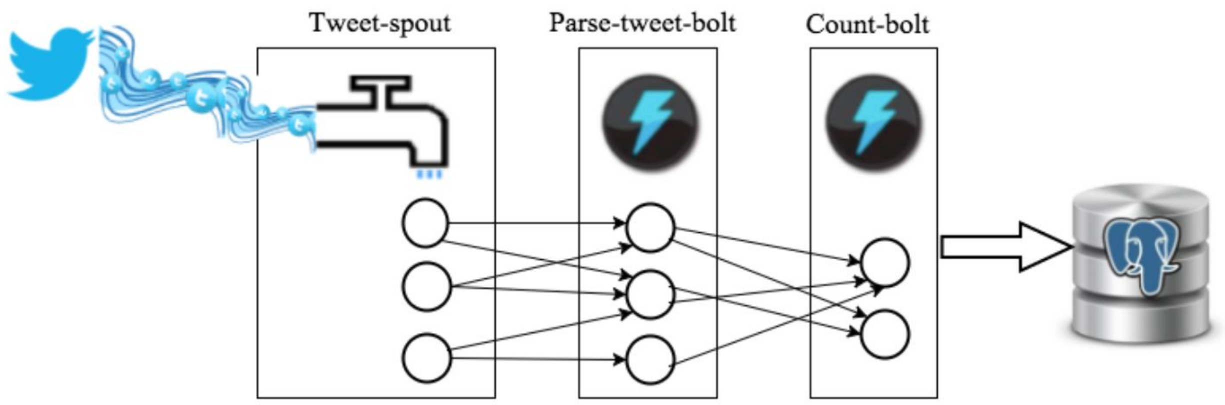

Figure 1: Application Topology

Directory and File Structure

File Name	Location	Description
tweets.py	extweetwordcount/src/spouts/tweets.py	Spout that connects to Twitter API and captures all tweets. The tweet is passed as a string to the bolts.
parse.py	extweetwordcount/src/bolts/parse.py	Bolt consumes the spout and splits string into words and makes sure every word has ascii characters. There is some additional filtering for symbols before being passed to the next bolt.
wordcount.py	extweetwordcount/src/bolts/wordcount.py	Bolt consumes the parse bolt and counts the number of occurrences of each word. It adds this updated count to both the Postgres table 'tweetwordcount' and as an output in the terminal.
tweetwordcount.clj	extweetwordcount/topologies/tweetwordcount.clj	Clojure file that defines the application topology
finalresults.py	extweetwordcount/finalresults.py	Python file that takes an input <word> and find the number of times it has occurred. If no word is input then it returns the count of all words alphabetically.
histogram.py	extweetwordcount/histogram.py	Python file that returns all words with counts between k1 and k2 inclusive.

How to Run

**Assumptions: It is assumed that you have python and postgres installed

1. Start a EC2 instance using the following AMI:

AMI Name: UCB MIDS W205 EX2-FULL
AMI ID: ami-d4dd4ec3

2. Clone this repository: git clone https://github.com/pbarranger/W205-S3.git

3. Navigate to extweetwordcount in Exercise 2: cd W205-S3/exercise_2/extweetwordcount

4. Start Postgres

5. Install psycopg: pip install psycopg2==2.6.2

6. Install tweepy: pip install tweepy

NOTE: It is assumed that you have the correct database created in Postgres. If this is your first time running and do not have the database setup, run:

```
psql -U postgres
create database tcount;
\c tcount
create table tweetwordcount(word TEXT PRIMARY KEY NOT NULL,count INT NOT NULL);
 \q
```

7. Enter Twitter Credentials in tweets.py: nano W205-S3/exercise_2/extweetwordcount/src/spouts/tweets.py

8. Navigate back to the extweetwordcount file

9. Run the Application: sparse run

**Use ctrl+c to stop the Application

You can now query your resulting data.

Optional:

Run finalresults.py to get the count of a word in dataset: python finalresults.py <word>

Run histogram.py to see all the words with counts between range k1 & k2 (inclusive): python histogram.py <k1>,<k2>