Architecture of Tweeter Streaming Pipeline *extweetwordcount*

1. Overview

As shown in Figure 1, this twitter streaming pipeline captures live data from Twitter using Twitter API and feeds the tweets into an Apache Storm for analysis. The tweets are then analyzed in real time and the analysis results are stored in a Postgres database. Depending on users' needs, Python serving scripts and a connected Tableau workbook can be then used to fetch data insights.

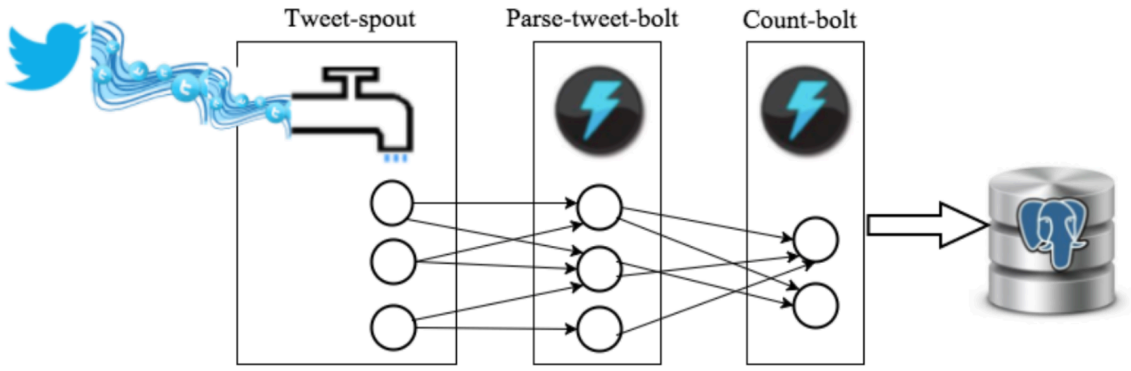
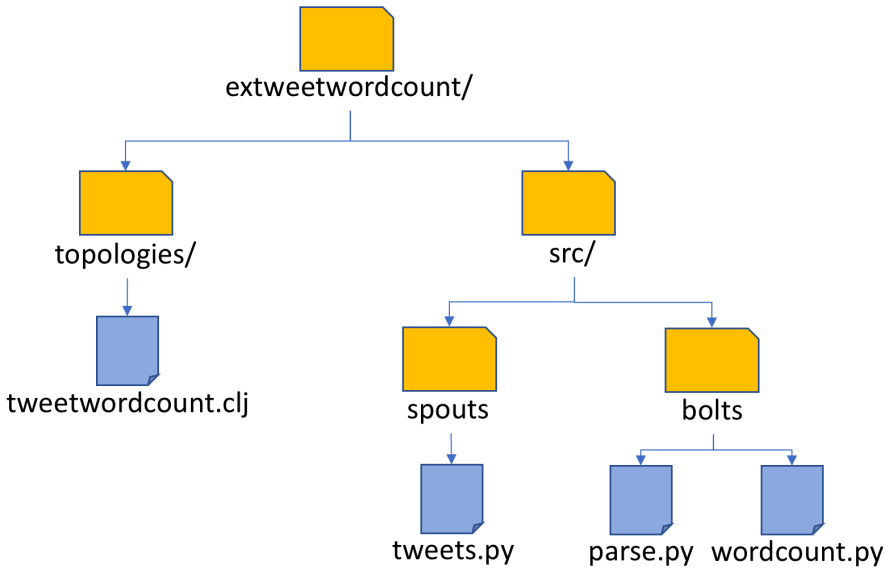


Figure 1: Application Topology

2. *extweetwordcount* in Apache Storm

"extweetwordcount" is a streamparse project built in the Apache Storm environment. It allows users to stream tweets, analyze the numbers of occurrences for words, and store results in a Postgres database. It follows the following file structure (only main files/folders included):



2.1 *extweetwordcount* Topology

The topology of this "extweetwordcount", *tweetwordcount.clj,* specifies the process logic.

2.2 *extweetwordcount* Spouts

tweet-spout process in *tweets.py* serves as a spout process to access Twitter Application and feed in twitter stream (English only) to the pipeline.

2.3 *extweetwordcount* Bolts

Twitter streams feed in from *tweet-spout* to *parse-tweet-bolt*, defined in *parse.py* to parse incoming tweets, which filters out symbols and non-ascii syntax. Then the valid words are emitted by *parse-tweet-bolt* to *count-bolt*, defined in *wordcount.py. count-bolt* connects to a Postgres Database *tcount* and store the words and word counts in a table called *tweetwordcount*.

To run *extweetwordcount*, please follow the following step:

1. start Hadoop
2. start Postgres
3. enter extweetwordcount/ directory
4. run command line "sparse run"

3. *extweetwordcount* Serving Layers

There are mainly three kinds of serving layers designed for *extweetwordcount:*

3.1 *extweetwordcount* Postgres Database

Users can access Postgres database *tcount* for streaming results stored in table *tweetcount.* This is served at 5432 port of the local machine (AWS EC2 in this instance).

3.2 *extweetwordcount* Python Serving Scripts

finalresults.py allows users to either 1) pass a single word as an argument to get its number of occurrences as a result or 2) run it without an argument to fetch all results. For examples:

```
$ python finalresults.py hello
Total number of occurrences of of "hello": 10
```

```
$ python finalresults.py
$ (<word1>, 2), (<word2>, 8), (<word3>, 6), (<word4>, 1), ...
```

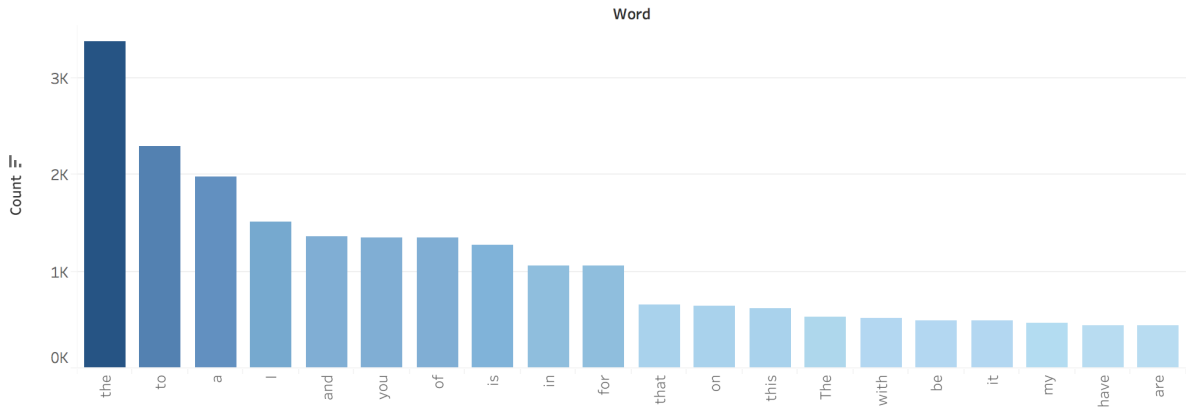
histogram.py allows users to specify two numbers k1, k2, and fetch all the words that occurred at least k1 and at most k2 times:

```
$ python histogram.py 3,8
<word2>: 8
<word3>: 6
<word1>: 3
```

3.3 *extweetwordcount* Tableau Dashboard

A Tableau dashboard has been connected to the Postgres database and can be used to visualize the streaming results:

Histogram of the 20 Most Popular Words



Tree Map of the 20 Most Popular Words