Twitter Application Architecture

W205 Exercise 2
Chris Murray

This application connects to Twitter and receives real-time tweets, extracts the individual words (ASCII characters only), counts them, and adds them to a database. The application uses Apache Storm to receive real-time streaming data and Postgres to database the words. A serving layer then queries the Postgres database to answer questions such as how many times a particular word appeared in tweets. See Figure 1 for an illustration of the architecture.

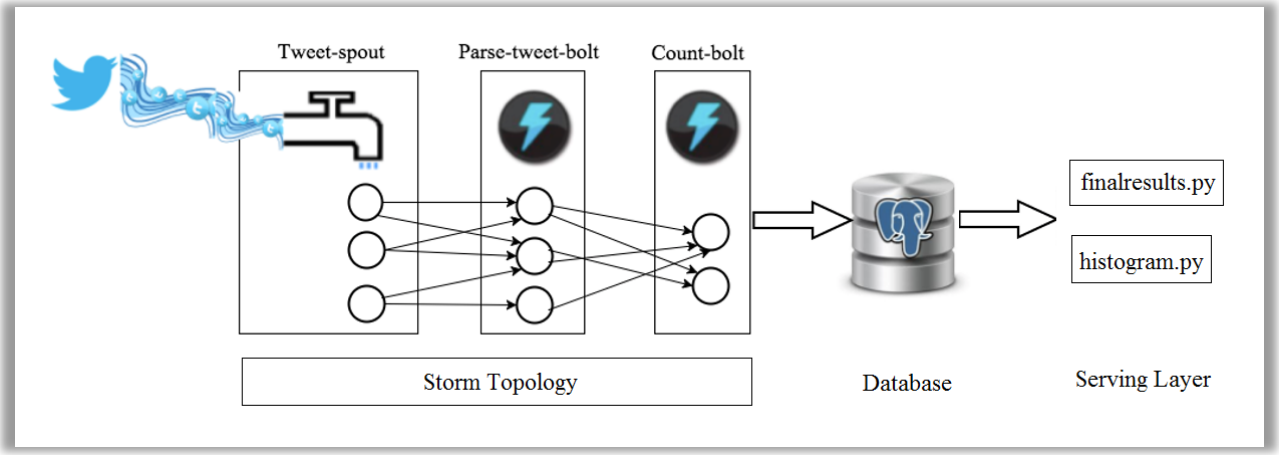

Figure 1: Twitter Application Architecture

Dependencies:

- Apache Storm
- Streamparse
- Postgres
- Python 2.7
    - tweepy
    - psycopg2

Important Files:

- `Twittercredentials.py` – credentials used by hello-stream-twitter.py
- `config.json` – Storm config file
- `finalresults.py` – serving layer script
- `hello-stream-twitter.py` – Twitter test script
- `histogram.py` – serving layer script
- `init.sql` – SQL file to create table
- `project.clj` – Storm config file

- `screenshots/` – folder containing screenshots
- `src/bolts/parse.py` – parse bolt
- `src/bolts/wordcount.py` – wordcount bolt
- `src/spouts/tweets.py` – tweets spout
- `topologies/tweetwordcount.clj` – topology file

Usage:

First, change directory to the Ex2 root directory:

```
# cd W205/Ex2/
```

Initialize the database:

```
# psql -d Tcount -U postgres -f init.sql
```

Run the application:

```
# streamparse run
```

View total wordcount results:

```
# python finalresults.py
```

View individual wordcount results:

```
# python finalresults.py hello
```

Print the words that appear between k1 and k2 times:

```
# python histogram.py 25,35
```