Nick Chen's Twitter Stream Application Documentation
November 16, 2016

This twitter streaming application streams tweets from twitter.com, parses them into individual words, and then updates a postgres database with the count of the number of times each word appears in the stream. There is a finalresults python script that allow the user to access the database results via bash shell commands. This documentation will describe each of the pieces of the application, the file structure and dependencies, and a description of how to run the application.

I. Application Components
  a. Topologies
    i. EX2Tweetwordcount.clj
      - Contains instructions for storm about how each of the application components are meant to interact.
      - Defines parallelism and number of instances of each spout and bolt are meant to run.
  b. Spouts
    i. tweets.py
      - Contains the API keys and secrets that were downloaded from twitter.com.
      - Interacts with the twitter API and streams tweets from twitter.com
      - Emits tweets for processing by the parse.py bolt.
  c. Bolts
    i. parse.py
      - Receives tweets from the tweets.py twitter spout.
      - Parses tweets into individual words, removing non ascii characters, and capitalizing all letters for standardization.
      - Emits individual words for processing by the wordcount.py bolt.
    ii. wordcount.py
      - Receives parsed words from the parse.py bolt.
      - Counts the number of times it encounters a given word.
      - Stores words and their counts in the postgres database.
  d. Serving Scripts
    i. setup_postgres_db_for_ex2.py
      - If tcount database already exists, then this script will delete it.
      - Creates tcount database.
      - Creates tweetwordcount table within the database.
    ii. finalresults.py
      - Allows the user to access the counts recorded by the streaming application via bash shell commands.
II. File structure within the ex2 directory
  a. ExerciseTweetwordcount
    i. topologies
      - Contains the EX2Tweetwordcount.clj topology.
    ii. src/bolts

        - Directory containing the parse.py and wordcount.py bolts
    iii. src/spouts
        - Directory containing the tweets.py spout
b. Serving_scripts
    i. Contains setup_postgres_db_for_ex2.py which sets up the database and table.
    ii. Contains finalresults.py which allows the user to access the application's resulting output.

III. Dependencies
a. Python packages
    i. tweepy
    ii. psycopg2
    iii. sys
    iv. future
    v. collections
    vi. streamparse
    vii. re
    viii. time
    ix. itertools
    x. Queue
    xi. threading
    xii. copy
b. Other software applications
    i. Apache storm
    ii. postgres

IV. Run Instructions
a. Create the Database
    i. Navigate to the folder ex2/serving_scripts and run the python file called setup_postgres_db_for_ex2.py.
    ii. This script deletes any existing database called 'tcount', creates a new blank database called 'tcount', and creates a table called 'tweetwordcount' into which words and their counts will later be placed by the streaming application.
b. Step 2 - Run the Streaming Application
    i. Navigate to the folder ex2/ExerciseTweetwordcount.
    ii. Run the command: sparse run
    iii. After about one minute, you will begin to see parsed words logged to the screen. This indicates that the application is running. Let the application run for as long as you would like.
    iv. As words are logged to the screen, they are also written to the DB.
    v. When you want to stop the stream, use ctrl + C to stop the stream.
c. Step 3 - Get Results
    i. The python script finalresults.py in the folder ex2/serving_scripts allows you to easily access the results of twitter stream.
    ii. If you want to get the the number of times a specific word occurred in the stream, run this script followed by the word you are interested in. For example: python finalresults.py testword.

iii. If you would like to see all words that occurred within a certain range of times, run this script followed by two integers. For example: python final results.py 5 10.
iv. If you would simply like to see all of the words that occurred in the stream along with their counts, run this script with no additional arguments: python finalresults.py.