Emily Geyman

Computer Science III

May 6, 2015

Final Project Design

**Functional Overview**

I will write a stand-alone Java program that will scrape tweets from Twitter and use the geolocation (latitude and longitude) of each tweet to answer a series of questions about the users of Twitter in each state. The three questions I plan to answer are:

1. In what state do people post the longest tweets (mean # of characters)?
2. What state has the “smartest tweeters”? I’ll measure this by calculating the mean word length after I remove the stop-words (a, and, an, etc.).
3. In what state do tweeters have the largest vocabulary? I’ll measure this by counting the number of unique words in the first 1,000 tweets per state.

I will have to write a helper method that will accept the latitude and longitude of the tweet and return the state (or null if the state is outside the U.S.).

**Design Overview**

Because I want a large data set and Twitter limits of the number of tweets I can take per time interval (15 minutes), I will spend a few days scraping tweets and saving them to a file on my computer. My program will use the tweets that are saved to the file(s).

At first, I will hard-code the entire project. If I have time, I may add some user interaction where the user can specify what states he or she wants to look at, etc.

The main output of the program will be a well-formatted table with the information I gather about the tweets from each state.

**Design Details**

Steps:

1. Write a Java program to scrape tweets and save them to a file on my computer
   1. I’ll probably store them in a .csv file where each line will have longitude, latitude, and the text of the tweet
   2. I’ve already written a program that scrapes the tweets and prints them to the console, so I’ll just have to modify the program so it sends the tweets to a file.
2. Create a class that defines “Tweet” objects
   1. Fields
      1. Latitude
      2. Longitude
      3. U.S. state 2-letter code (ex: WA)
      4. The text of the tweet (stored as a String)
   2. Make “Tweet” objects comparable (sorted first by state, then use latitude and longitude as tiebreakers)
3. Main program:
   1. Write a Java “getState” method that accepts a longitude and latitude and returns the state (or null if the location is outside the U.S.)
      1. I’ll use a .csv file I found on the internet that contains 43,000 longitude-latitude pairs in the U.S. and their corresponding zip-code and state.
      2. Pseudocode:

For each tweet

* + - * 1. Scan through zip-code file until find a zip-code where the latitude and longitude are within some value (ex: 0.1) of the tweet’s geolocation
        2. Return the state of that zip code
  1. Construct an ArrayList of Arrays of Tweet objects. Each array in the ArrayList will contain tweets from a particular state.
     1. Pseudocode:

Go through the .csv file:

For each line in the csv file (each line contains info about one tweet)

Call the getState method

Construct a Tweet object and add it to the array for the given state

* 1. Part 1: In what state do people post the longest tweets (mean # of characters)?

For each state

For each tweet

Count number of characters

Divide total characters by number of tweets

* 1. Part 2: In what state do tweets have the longest words (excluding stop words)?

Store a .txt file with stop words on my computer

Put the stop words into a String[]

For each state

For each tweet

If word isn’t a stop word

Count number of characters

Increment number of counted words

Divide total characters by number of counted words

* 1. Part 3: In what state do tweeters have the largest vocabulary?

For each state

Make a state “dictionary” (a Set of words)

If the state has at least 1,000 tweets:

For each of the first 1,000 tweets

For each word

If word isn’t already in dictionary

Add to dictionary

Count number of words in dictionary

* 1. Use printf to print a nice table with the results for each state to the console

**Testing Plan**

I’ll test each part of the project individually:

* + - 1. Test my tweet scraping program:
         1. Run program for a short time (about 30 seconds) and review the tweets that I collect to make sure they all have geolocation, etc.
         2. Run program for as long as it takes to reach Twitter’s scraping limit

Record how many tweets I got and how long it took

* + - 1. Test getState() method by passing in a couple latitude-longitude pairs and seeing if the method returns the correct state. I’ll also pass in latitude-longitude pairs that are outside of the U.S.
      2. Make sure my ArrayList<Tweet[]> is properly sorted
         1. Sort tweets from a small file (about 100 tweets) and check to make sure they are grouped by state
      3. Test parts 1, 2, and 3 on a small group of tweets
         1. Do some of the arithmetic manually to make sure the code works properly

**Grading Rubric**

5 points: program effectively scrapes tweets and saves them to a file

5 points: the Tweet class is encapsulated and follows object-oriented programming standards

10 points: the program answers one or more interesting questions about the tweets

10 points: each part of the code works as expected

5 points: the output is well-formatted

5 points: the code is readable and well-documented

**Proposed Implementation Schedule**

|  |  |
| --- | --- |
| Friday, May 8 | Figure out how to scrape tweets and save them to a file  Test my Twitter-scraping program |
| Monday, May 11 | Create a class that defines “Tweet” objects  Start collecting tweets |
| Wednesday, May 13 | Write the getState method and test it  Continue collecting tweets |
| Monday, May 18 | Figure out how to store the tweets in arrays (organized by state) |
| Wednesday, May 20 | Write code for part 1 |
| Friday, May 22 | Write code for part 2 |
| Monday, May 25 | Write code for part 3 |
| Wednesday, May 27 | Turn in project |

**Potential Show-Stoppers**

I’m worried about hitting Twitter’s scraping limit and not being able to collect enough tweets. I want to only collect tweets that contain location information. I wrote a program to collect all tweets and found that only about 1 in 15 tweets has location information. I’ll probably have to run my collection program over and over again until I collect enough.

I also think it will be fairly difficult to write the method that accepts a latitude-longitude pair and returns a U.S. state.

**Open Questions**

I’ll need to do a lot of thinking about how to extract a fair and representative sample of tweets from each state. For example, one thing I’m worried about is that, if there’s a major event in a given state one day (such as an earthquake), then all the tweets will contain the same types of words. This will affect my “in which state to tweeters have the largest vocabularies?” test.

**Resources**

I will use the twitter4j library to access the Twitter API.

In my initial program to extract tweets, I used and modified some code I found on the following link: http://stackoverflow.com/questions/23341215/extracting-tweets-of-a-specific-hashtag-using-twitter4j