Socio-Geographic Map of Doha

Twitter Languages in Qatar v1.0

Project Overview

The objective of this project was to create an interactive map that shows the language distribution of geolocated tweets that have been collected around the State of Qatar. The idea was to visualize the data available from Twitter and to bring out interesting trends that exist in the diverse population demographics of Qatar.

In this project, first the raw tweets during a particular period of time were collected, and the languages of the geolocated tweets were then determined using the Language Detection API. The tweets were then marked on the map using grid-based clustering, where the map was divided into squares of a certain size, and the tweets were grouped into each grid square. For each grid square, there was a centroid marker which displayed a pie chart representing the different languages that people tweeted in. With this step, we hoped to get a rough insight about the population demographics in the corresponding areas.

Tools

- Language Detection API
  https://code.google.com/p/language-detection
- Google Fusion Tables
  http://www.google.com/drive/apps.html#fusiontables
- Google Maps JavaScript API v3
  https://developers.google.com/maps/documentation/javascript
- Google Image Charts API
  https://developers.google.com/chart/image/docs/gallery/pie_charts
- Perl and Python scripts

Data Visualization Process

I. Data Pre-Processing with Perl and Python Scripts

For the purpose of data extraction, a tab-separated .txt file of the following format was used:

userid [TAB] tweetid [TAB] tweet text [TAB] latitude [TAB] longitude [TAB] language [NEWLINE]

A Perl script, **round.pl**, was run on the .txt file, which had two main implications:

(i) Eliminate all the tweets whose location was not within the boundaries of Qatar:

For this, the coordinates of the Qatar border on all four fronts were used to write a conditional statement, that eliminated all out-of-bound tweets.

Here are the coordinates of the Qatar borders:
North (26.18471, 51.237144)
West (25.65762, 50.73349)
East (25.296854, 51.645355)
South (24.47112, 51.098616)

(ii) Map individual tweets to centroids in the grid

We wanted to accumulate the tweets that were geographically very close to each other into a single point, since that way we would have sufficient data at a point that shows the language trend in a particular region or square kilometer.

The method 'round' in the script essentially rounds the latitudes and longitudes of individual tweets and maps them to the nearest centroid.

At the end of these two steps, the script generated an output file with the following format:

userid [TAB] tweetid [TAB] tweet text [TAB] centroid latitude [TAB] centroid longitude [TAB] language [NEWLINE]

The next task was to aggregate the counts for each language in each grid square. So the output file generated by the round.pl script was run through two other Python scripts, **process.py** and **sum.py**, which aggregated the counts for each language and put them all in a single tab-separated line.

At the end of these steps, the generated output file was of the following format:

centroid latitude [TAB] centroid longitude [TAB] en [TAB] ar [TAB] tl [TAB] es [TAB] fr [TAB] ur [TAB] other [TAB] ne [TAB] hi [TAB] ml [TAB] bn [NEWLINE]

where,
en = English
ar = Arabic
tl = Tagalog
es = Spanish
fr = French
ur = Urdu
other = language not detected
ne = Nepali
hi = Hindi
ml = Malayalam
bn = Bengali

This output file was again run through a Perl script, **pie_micro.pl**, for generating pie charts as .png files, which would later be used as markers. The pie charts represented the 3 major languages: English, Arabic, and Tagalog, and aggregated the other languages as 'Other' for each grid square.

It also generated a new file with an added column, which stored the URL of the .png file for each row. The format of the final output file was as follows:

centroid latitude [TAB] centroid longitude [TAB] en [TAB] ar [TAB] tl [TAB] es [TAB] fr [TAB] ur [TAB] other [TAB] ne [TAB] hi [TAB] ml [TAB] bn [TAB] image url [NEWLINE]

II. Importing to Fusion Tables

Once the data file was processed and ready for use, it was then imported into Google Fusion tables. Fusion Tables is an experimental data visualization web application to gather, visualize, and share large data tables. We decided to use Fusion Tables particularly because it had an inbuilt map generator where we could visualize our data for testing purposes.

The .txt file was imported into the Fusion Tables successfully using Tab as the Separator character and UTF-8 as the Character encoding. The table can be found at the following link: https://www.google.com/fusiontables/DataSource?docid=1JvYEXp4jSKD2uiec0IGGYXAVoMvYbE9n8cRfpXE

Each table has its own unique table ID as the identifier. Below is the table ID for the one we used in this project:
Table ID: 1JvYEXp4jSKD2uiec0IGGYXAVoMvYbE9n8cRfpXE

III. Visualization using Google Image Charts API and Google Maps JavaScript API v3

We decided to use a website for the visualization, and used Google Maps Javascript API to embed the map onto the website. The website basically makes a call to the particular Fusion Table using the table ID and pulls data from it.

First, it uses the pie chart .png files generated earlier to create markers on the map, and then generates another set of pie charts using all the languages for each row, with the help of Google Image Charts. These pie charts are embedded into the infowindows for each marker, which can be viewed upon clicking the markers.

While creating the markers, we also eliminate the points that have less than a minimum threshold of tweets (in this case, 10 tweets).