Using IBM's Tonality Analysis of Language and Geolocated Tweets to Map Emotional Intensity.

Isaac Callison (ic2d@mtmail.mtsu.edu)
Middle Tennessee State University

April 9, 2020

Abstract

Metadata, and extraction of relevant information from that data, is exploding. Companies and organizations are competing at a furious rate to glean advantages from ever expanding datasets. One such voluminous and continuously updated dataset are tweets from the social-media giant Twitter. Tweets from various regions were collected in real-time using Tweepy. Geolocation was obtained through location parsing and geocoding. Using IBM's tone analyzer, the emotional nature of these tweets was analyzed and the predominant emotion was chosen. The results were mapped onto a Google Heatmap.

1 Introduction

There is a convergence of powerful technologies that allow for near- instantaneous notification of current events using available meta-data from social media platforms. One can presume a possible correlation between the growing data and the usefulness that can be extracted from said data. One social media platform for which, at least the possibility of obtaining relevant data is present, is Twitter. This is not lost on Twitter. In fact, part of Twitter's business model is selling data through premium API's[1].

The goal of this project was to gather and analyze tweets for emotive tonality, then display this on a "heatmap" of emotive intensity for a specific region. To this end, three separate technologies that were, at least for this project interdependent, were utilized. First, using Twitter's developer API, geolocated tweets were collected from a specific region and radius, then pre- processed to extract latitude, longitude and text. Secondly, using IBM's Watson and natural language processing capabilities, the tweets were assesed for emotive tonality. Finally, a Google map was displayed with a heatmap layer graphing the intensity of these emotions.

In a perfect world with enough data coming in of the type referenced above, a real-time map of the emotional state of a town or city could be analyzed. The reality fell far short of what was envisioned and the the successes and shortcomings will be documented herein.

2 Background

As stated above, there were several interdependent moving parts with this project. With regards to platform, a Jupyter Notebook was used for this project to pull in modules, access API's, and make GET and POST requests to those APIs.

The datasets for the natural language processing came from Twitter. The Twitter API allows for the triangulation of tweets provided certain data[5]. However, the ability to extract out latitude and longitude from a tweet, which was previously available, was culled in June of 2019[2]. In response, other methods were used to triangulage tweets.

Using developer authentication, and a GET request with certain location parameters, one can obtain a list of current tweets within a search radius in JSON format. A great deal of meta-data is

[1] https://developer.twitter.com/en/premium-apis

[2] The author was not aware of this change when launching this project. Please see: https://www.engadget.com/2019-06-19-twitter-removes-precise-geo-tagging.html

returned in this format, but from these tweets one can glean a myriad of data, including up until recently, the latitude and longitude of the tweet.

The second part of this project was natural language processing with IBM's Watson using tonality analysis. IBM has a cloud computing program with various machine learning capabilities[3], one of which is tonality analysis. The natural language tonality processing that Watson offers can, among other things, extract emotion from a corpus. In this specific case, a variety of emotions, including fear, joy, analytical, confident. Further, the intensity of a specific emotion can derived at the sentence and document level.

As a graphical representation of the data collected, a heatmap was used. A heatmap overlay is a feature offered by Google Maps. It can create a visualization to depict the magnitude variation of data at a range of latitudinal and longitudinal points. A heatmap is very usefule when you have a great deal of data points of varying magnitude and their geographic position. One type of dataset that lends itself well to such graphical representation is eathquake data[3]. When the Heatmap Layer is enabled, a colored overlay will appear on top of the map. By default, areas of higher intensity or magnitude will be colored red, and areas of lower intensity will appear green[2].

3 Research Method

The consoles and APIs of Twitter, IBM's Watson, and Google were accessed. The three aformentioned services required developer accounts to be used. Twitter's API required an application and pre-approval. These credentials were secured and inserted directly into the code for ease. Initially, separate Notebook's were used to test the various API's before integration into the main Notebook.

Postman, a REST API testing software was used to experiment with Twitter's API[4] before integrating Tweepy's tweet streaming code into the main Notebook. These accounts were free to use, up to a point. After some extensive testing with IBM's Watson an upgraded account had to be setup to continue.

All work was done on a Linux desktop environment. Code was written in Python 3. A Jupyter Notebook via Anaconda was used to make GET requests of Twitter using the Tweepy module to stream a set number of tweets. The original plan was to extract the latitude and longitude directly from the tweet, and of course the tweet itself. Because Twitter no longer provides the latitude and longitude of tweets a workaround was required. Some of the tweets also had embedded address location. An open source API called Nominatum was used to reverse geocode addresses which were still embedded in tweet data[4].

As a result of the modified method, the Tweets pulled in by the Notebook were still run through the IBM tone analyzer after significatn pre-processing.IBM provides a Watson Software Development Kit for integration into Python and Jupyter. The text from the obtained tweets will be passed into Watson for tonality processing.

Display a heatmap of emotion in a certain region based on the intensity of the selected emotional state. A Google Map will display a heatmap layer with the attendant emotion and intensity. For instance, in areas of a region where the tweets have low sadness, green shading will predominate, changing to red as the intensity of that emotion increases

[3]An eathquake dataset is included in the gmaps package and is used for illustration and tutorials

[4]https://nominatim.org/

4 Results and Analysis

Unfortunately, the fine-grain location data of each tweet was removed by Twitter in mid-2019. As a result, tweets could be pulled from a specific region, but the exact latitude and longitude of that tweet was not discerable. What was initially envisioned was a fine grain heatmap showing data across any number of cities. The method was modified from the original thusly...

5 Conclusion and Future Work

We live in an age of ever expanding meta-data. As this increases, humanity will seek to harness this data through new technologies, for better or worse. In this case, Twitter data can potentially be used to inform and improve the lives of everyday citizens. If the above can be implemented, one could graph the emotional intensity of differing regions based on twitter content.

References

[1] Callison, I. Jupyter Notebook. https://github.com/cthulhu1988/SelTopicsAI/tree/master/NLPpaper, 2020. [Online; accessed 9-March-2020].

[2] Google,. Heatmap API. https://developers.google.com/maps/documentation/javascript/heatmaplayer, 2020. [Online; accessed 10-March-2020].

[3] IBM Incorporated,. IBM's Watson: Cloud Computing. https://cloud.ibm.com/apidocs/tone-analyzer, 2020. [Online; accessed 10-March-2020].

[4] Postman Incorporated,. Postman API. https://www.postman.com/, 2020. [Online; accessed 29-February-2020].

[5] Twitter Incorporated,. Twitter Documentation: Geocode API. https://developer.twitter.com/en/docs/geo/places-near-location/api-reference/get-geo-search, 2020. [Online; accessed 29-February-2020].

A Coding Environment

All code was run on Ubuntu 18.04 LTS. Anaconda was installed and Jupyter Notebooks were utilized. All code was written in Python 3. There was some difficulty in getting Google Maps to display in JupyterLab. Several modules were required to run the various APIs utilized in this project. The following process was used to install Anaconda and the modules used:

```
// From a bash command prompt:
curl -O https://repo.anaconda.com/archive/Anaconda3-5.2.0-Linux-x86_64.sh
bash Anaconda3-5.2.0-Linux-x86_64.sh


// After installation of Anaconda, these modules should be installed:
pip install ibm_watson
pip install tweepy
pip install geopy

// For gmaps:
jupyter nbextension enable --py --sys-prefix widgetsnbextension
pip install gmaps
jupyter nbextension enable --py --sys-prefix gmaps
```

Figure 1: Anaconda & Module Setup