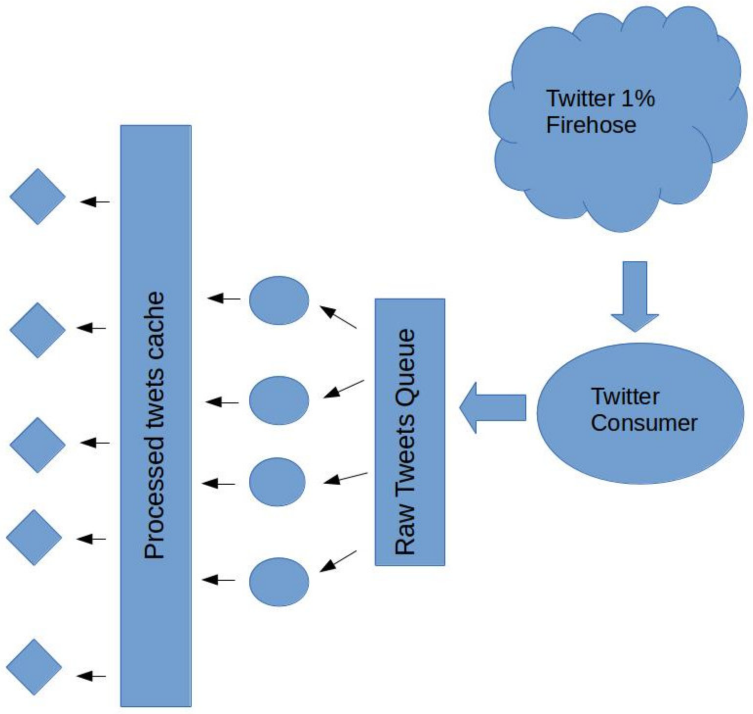
Twitter Sentiment Analysis

Caltech CS 145 Term Project Proposal
CS 144 - Professor Adam Wierman
Aleksander Bello, Alexandru Cioc, Victor Duan, Archan Luhar, Louis O'Bryan
March 19, 2014

Background

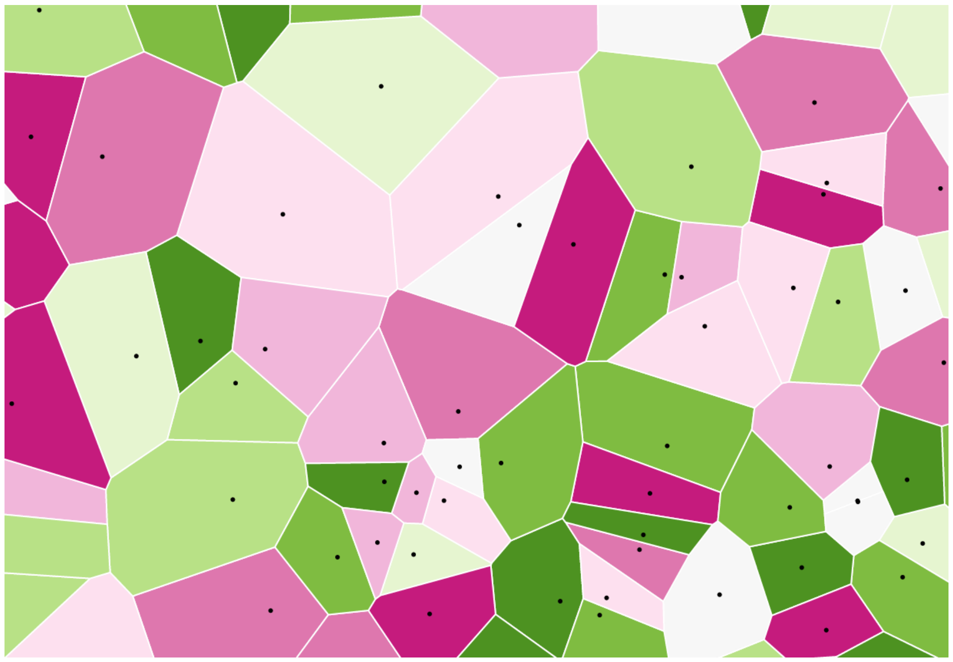
How is the world feeling right now? That is a hard question so to make it easier and narrow the scope down to the quantifiable, we ask, how is Twitter feeling right now? We propose here an application and underlying infrastructure to categorize tweets based on emotional content, create a representative sample, and visualize the sentiments on a map in a web browser in real-time.

This project faces a range of technical challenges, from processing "#2cool4school" to handling the large amount of tweets from the open 1% of Twitter's "firehose" as well as visualizing sentiments in real-time. It is important to get a glimpse of the infrastructure that would have to be built in order to attack this open problem.



The backend for such a system with great emphasis on a real-time experience would have to be extremely efficient. There would be on consumer reading from the firehose since the Twitter Streaming API notes that there is no speed benefit in having multiple consumers (they'd get the same tweets). The system would have to utilize time-expiring, short-lived, fast, in-memory caching, probably in a distributed cluster. The natural language processing can be parallelized using multiple instances getting tweets from the raw tweets queue since each tweet is independent. The processed tweet would then be saved in the appropriate cache, ready to be served to clustering clients that aggregate and cluster the sentiments to geographic locations.

Lastly, a frontend would be necessary to visualize the sentiments. It would receive in pseudo-real-time an updated map of sentiments. We propose a Voronoi Tessellation as seen in the figure below. Each cluster of similar sentiments in a geographic location can be represented as a dot surrounded by a color representing the emotion and its intensity. The region surrounding a dot encompasses all points to which it is closest.



Source: http://bl.ocks.org/mbostock/4060366

Literary Review

Each of the challenges this project faces could be a project of its own. Luckily, the following are sources whose ideas are directly related to our project and will be immensely helpful:

Go, Alex, Richa Bhayani, and Lei Huang. "Twitter Sentiment Classification Using Distant Supervision." *Processing*, 2009.

Go *et al.* used machine-learning algorithms to characterize the sentiment expressed in Twitter posts (Tweets). Classifiers included Naïve Bayes, maximum entropy (MaxEnt), and support vector machines (SVMs). In order to train the classifiers, Tweets with emoticons were used. ":)", ":-)", ":)", ":D", "=)", were all mapped to ":)", while ":(", ":-(", and ": (" were mapped to ":(". During post processing, emoticons were removed for training purposes in order to ensure that MaxEnt and SVMs did not put weight into the emoticon symbol itself, but rather in the words the Tweet expresses. Furthermore, tweets with opposing emoticons were removed in order to better train the classifiers, as were retweets, repeated tweets, and tweets featuring ":P" (when the report was written, the Twitter API returned ":P" during queries for the unhappy emoticons). Using the aforementioned training samples, all three classifiers were able to achieve accuracy greater than 80%. This helps us conclude that using such classifiers might be beneficial to our classification purposes. Another interesting conclusion is that using part-of-speech (POS) tags, which aim to further classify a word's connotation by its part of speech, were not useful and resulted in worse overall predictions. For more information, in addition to Go *et al.*, we can refer to the original Stanford class final project that the research paper arose from (http://www-nlp.stanford.edu/courses/cs224n/2009/fp/3.pdf).

Kouloumpis, Efthymios, Theresa Wilson, and Johanna Moore. "Twitter Sentiment Analysis: The Good the Bad and the OMG!" *Proceedings of the Fifth International AAAI Conference on Weblogs and Social Media*, July 5, 2011.

Kouloumpis *et al*. used three different data sets for their approach of analyzing the usefulness of linguistic features in sentiment detection. The first was a hashtagged data set, from which top hashtags were manually selected and corresponding Tweets were associated with positive, negative, or neutral emotions. This allowed the polarity of the hashtag to indicate the polarity, or lack thereof, of the Tweet. The second data set was created by Go *et al*. It was noted that the dataset contained very few hashtags – a possible indicator that emoticons and hashtags do not commonly coexist within the same Tweet. The final dataset was the iSteve dataset, which is a collection of hand-annotated Tweets used for evaluations. Using the aforementioned data sets, Tweets were tokenized, normalized, and finally tagged using part-of-speech (POS) tagging. Unigrams and bigrams were used as a baseline, and a number of features were tested. These included: *n*-gram features (where the top 1,00 *n*-grams were selected in a "bag-of-words" model after the removal of stop words and negation detection), lexicon features (from the MPQA subjectivity lexicon), POS features (which counted the number of parts of speech in each Tweet), and micro-blogging features (usage of the Internet Lingo Dictionary, net slang dictionaries, and the presence of emoticons and abbreviations that indicate sentiment). Upon training and categorization, it was found that the use of *n*-grams with lexicon features and micro-blogging features provided the best result. As Go *et al*, it was found that POS features actually decreased accuracy. Overall, top accuracy remained around 74 – 75%. Finally, another conclusion was that hashtags could be as good as emoticons in the detection of sentiment.

Bifet, Albert, and Eibe Frank. "Sentiment Knowledge Discovery in Twitter Streaming Data." *13th International Conference on Discovery Science (DS 2010)*, October 2010, 1-15.

Bifet *et al* focuses on challenges and techniques that arise from the Twitter data stream. It notes the authentication and file formats (XML or JSON) that the Twitter API supports. Furthermore, Bifet *et al* states, and concludes, that the use of a "prequential estimate of [the sliding window] Kappa [statistic]" is the best measure of accuracy for a real-time data stream because it can be calculated efficiently using a matrix and does not require that data be roughly 50-50 good/bad (some statistics require that the overall data pool is, on average, neutral – something that cannot be guaranteed when considering real-time data streams). Three incremental methods were used to process the data streams. The first was a multinomial naïve Bayes classier that finds the probability of observing a certain word in a "bag-of-words" Tweet. This frequency can be used to predict the sentiment of the overall Tweet since. A second method was stochastic gradient descent (SGD), which, given a specific learning rate, can be used to quickly associate words with sentiments – it is noted that specifying a "good" learning rate is crucial, since a learning rate that is too low will now allow the classifier to adapt to new changes rapidly enough. Finally, a Hoeffding Tree was also used. A Hoeffding Tree is a decision tree that can be used for classification, though it is rarely used for documents because it requires "high-dimensional feature vectors." Massive Online Analysis (MOA) and WEKA were used for classification, as was the Edinburgh Corpora and the Twitter Sentiment data sets from Go *et al*. Bifet *et al*, concluded that SGD was the best measure of real time data while the Hoeffding Tree was the worst. Although the Go *et al* data set was used for classification, the Edinburgh Corpora dataset was also necessary since it was not balanced and, therefore, represented a better simulation of the Twitter data stream.

Sakaki, Takeshi, Makoto Okazaki, and Yutaka Matsuo. "Earthquake Shakes Twitter Users: Real-time Event Detection by Social Sensors." *WWW '10 Proceedings of the 19th International Conference on World Wide Web*, April 2010, 851-60.

Sakaki *et al* discusses the usage, and implementation, of a system that can detect Earthquakes using real-time Tweets. An i.i.d. exponential distribution is assumed for the occurrence of Tweets after/ during an event. It is noted, however, that such a distribution should not be assumed for the spread of media events – rather, it should only be assumed for naturally occurring phenomenon where each person can independently experience the event. Graphs depict the spread of Tweets and interactions and show these differences. Sakaki *et al* also creates different models for identifying the location of an event. These models, and their corresponding algorithms, are described in detail. The models still consider positive and negative classes in Tweet classification because they must detect whether a Tweet is literally referring to a current seismological event or whether it is referring to a figure of speech/ past, resolved, event. Using the described methods, Sakaki *et al*, was able to detect earthquakes with a 96% probability (for scale 3 or higher) and deliver notifications faster than the Japan Meteorological Agency (JMA).

Timeline

It will be of utmost importance to get each part of this project working in a timely fashion. Of most importance is the backend infrastructure which will require great planning -- especially when designing an inter-daemon API.

Collection of Twitter Data (1-2 weeks)

The first part of our project will be to gather twitter data. We will need to figure out how to use Twitter's one-percent firehose API or the Twitter streaming API to obtain public tweet information. We will also need to find a good storage solution, such as a MySQL or SQL Server database, for the data so that we can analyze it later on. If we choose not to use a database, we will need to find an easy way to parse the data once it has been downloaded because most of our work will depend on NLP processing of tweets. Twitter uses OAuth to authenticate application access, so we will need to get a developer account and connect through their API. It will be important to collect the data early because Twitter limits the rate of its streaming public data, and we would like a significant sample.

Sentiment Analysis of Tweets (5 weeks)

Analysis of our data will use the majority of our time on this project. We will need to study existing NLP techniques in academic papers, as well as technical details such as NLP libraries, in order to find the meaning of the tweets and their intensity. Several algorithms, such as multinomial naive Bayes classifiers, support vector machines, and stochastic gradient descent have been listed in our literature review. Our whole team will need to get up to speed on what the available methods are and how to tailor them to our specific problem. Some of the NLP libraries that may help us implement these algorithms include Stanford NLP software and Apache OpenNLP. We will need to test a few of our options in order to find which ones perform the best on our data. Ultimately, we may choose to modify an existing algorithm or create our own. This will be an interesting part of the process because sentiment analysis may need significant modifications for short messages, such as tweets, compared to longer articles. The performance of our algorithms will be based upon either Twitter hashtags, some other metadata given by the API about the general topic of the tweet, or manual evaluation of a set of messages for their sentiment. We may also need to evaluate the intensity of the sentiment manually since it is not included in the tweets themselves if we choose to use supervised learning for this part. At the end of our analysis, the intensity and sentiment of tweets should roughly reflect the actual intensity and sentiment.

In addition, we will need to find a way to store our results so that our frontend will be able to query them based on specific criteria. We will need to figure out what information actually needs to be kept for later use so that it can be quickly accessed when users request it.

Data Visualization and Frontend (3 weeks)

After analyzing our data, we will need a way for it to be visualized by users from a browser. Part of this will be finding ways to query our results without flooding the clients with too much information. So, we will need to find representative samples or aggregations of the data based on certain search terms. Additionally, we will need to provide a meaningful visualization of the data on the frontend. Potential tools may include JavaScript D3, JS Graph It or other JavaScript graphing libraries. We would like to create a heat map of our results based on geographic regions, the sentiments searched for, and the sentiment intensity. It will also be interesting to display how sentiments have changed over time through a series of graphs, changing display, or some other means.

End Product

The end product would be a web app that shows a map the USA with the overall Twitter sentiment as well as sentiment of trending topics. The main points of interest would be the heatmap and the trending topics. The heatmap could show how people are reacting to a specific topic or just overall Twitter sentiment. The trending topics now only shows the user what is currently going on, but also allows for interaction with the heatmap to see how the topic is being received.

The topics will be found as a result of our learning process and can be displayed at the top or side of the map. These topics would allow our users to get a view of what is going on in the area at a simple glance. The topics shown could be updated whenever the map is moved or zoomed and could also auto-update every so often. Clicking on these topics would change the map to only show the sentiment map of that topic. Users will also be allowed to choose multiple topics so all of those topics will be mapped. This will allow users to analyze any topics they may be interested in any region. Furthermore, user supplied topics can be used (provided they are in our database of topics).

The heatmap is the primary visualization of the information we get. For the chosen topics, the average sentiment will be shown on the map. If a region is very positive about the topics overall, it will be displayed, and similarly for negative responses. The granularity will be determined by the tools we use, but the user will also have some access to playing with this. The map can initially be on the entire USA, but will work similar to google maps where the focus can be moved and users can zoom in or out depending on what they are interested in.

The final element of our product would be historical data. This information might only be saved for the top topics. The idea is that a user may select his topic(s) of choice over a certain region, and then access the sentiment history. This could be shown in a graph form and could provide a very useful statistic to study how a topic's perception changes over time.

As far as worst-case scenarios go, there are a few steps that are prone to problems. First, managing the data stream we can get from Twitter can be difficult to get working correctly. Next, sentiment analysis in general is a very hard problem and it will be difficult to do well. It will be key to have good NLP and ML techniques and a general strategy that tends to work well. Finally, the visualization will be a very involved process and may be very time consuming to get it working as we want it to. In the end however, we should be able to get everything working. Our primary problem will likely be finishing everything we want to do within a single term. Having a large group, however, will definitely be beneficial to completing on time. However, if we end up getting stuck on data management, sentiment analysis, or visualization, we should still be able to complete a minimal viable product that includes the basics of our project. This would include collecting all the data from Twitter still. The tweets will then be analyzed by our machine learning and natural language processing techniques, which hopefully do a good job of determining sentiment. Finally, the visualization should include at least a map of the USA as a whole. The choice of topics may not be as advanced as we hope. Also, the historical data may not be implemented, as it is more of a secondary visualization of the data. Overall, however, this embodies the focus of our project.