dash4twitter: A Dashboard for Twitter Analytics

Stephen Pryor
The University of Texas at Austin
405 W 25th St, Austin TX 78705
spyor02@utexas.edu

Joseph Perez
The University of Texas at Austin
405 W 25th St, Austin TX 78705
jperez@cs.utexas.edu

Abstract

Around the world, Twitter users generate about 190 million tweets per day. Within these tweets lies an enormous amount of data. Within these is valuable information companies want to understand. Synthesizing this much information into relevant, human readable chunks is a valuable and necessary step to understanding the value of Twitter data. Our project goal is to provide a NLP filter-based framework to examine streaming and historical twitter data in real-time. Our system is called *dash4twitter*, a dashboard built to provide real-time analytics. This report details our implementation to generate keywords using extraction using pointwise mutual information and front-end development.

1 Introduction

Analysis and visualization for twitter data have become a common practice for industry as well as academic research. Some are interested in modeling user behaviors while others focus on a more wide-scale study of twitter data. Perera et al. (2010) observed the tweeting patterns of users based on the frequency of tweets and retweets they post to the president's twitter account. Their idea is to characterize a user based on this interaction data. We found this idea very interesting, however we would to like to focus on a larger analysis of tweets. The idea of characterizing twitter users is quite interesting, especially to those working in industry fields such as advertising. A common methodology to address this problem is through sentiment analysis. Pak and Paroubek (2010) researched this area by creating a corpus from tweets into three separate datasets based on sentiment. Afterwards, they do feature extraction and sentiment classification through n-grams and a naive Bayes classifier, respectively. We find sentiment analysis to be a very interesting problem to research as it provides an effective way to understand massive amounts of data that are becoming commonplace in our daily lives.

The ability to deduce trending topics from social media is another area we are interested in researching. Suh et al. (2010) researched how some information becomes more widespread than others based on retweets. Mendoza et al. (2010) is a juxtaposition of Suh et al. (2010)'s work as they questioned the reliability of information in tweets through case studies. In Marcus et al. (2011), MacEachren et al. (2011), and Earle et al. (2012), they analyzed world events based on tweets by analyzing information such as sentiment and implicit and explicit geographical information. After that has been completed, the information is visualized by displaying things such as a time-line of tweets including the peak activity and common themes from the tweets. This information could prove useful to people who would like to understand what is occurring during an event like a natural disaster. Mathioudakis and Koudas (2010) also explores this idea of detecting trends from the twitter stream through an efficient manner.

This type of analysis is more relevant to what we are interested in. However, we are less focused on large, spike events. Rather, we would like to have the ability to examine twitter data based on rapidly deployable predefined and user-defined filters. Sriram et al. (2010) researched a similar idea by classifying tweets based on predefined classes through a simple feature selector. Hao et al. (2011) performs sentiment analysis based on streaming twitter data and generates a visualization of sentiment using a pixel map of the world. This type of research is what made us interested in analyzing and developing a visualization of twitter data which lead to our first iteration of dash4twitter. Our overall goal is to design a frame-

work so others may be able to improve our classifiers while having an intuitive and modern interface to build upon.

2 Implementation

2.1 Keyword extractor

Using a similar idea from Sriram et al. (2010), our framework has been designed to support live, filter-based analytics centered around the notion of user directed exploratory search of historical and temporal data. In addition, we desired to create a framework for future developers so they may access to a modern interface with the ability to easily integrate more elegant back-end software.

Our first task was to develop functionality which will, given a word or hashtag, return an ordered set of related unigrams and hashtags – generally, keyword extraction. Our initial approach to achieve this end was very simple. We began by counting co-occurrences of words in tweets. For example, if a tweet consisted of a warm day, the co-occurrences for warm with day would be incremented by 1, and vice-versa. This approach seemed to work rather well but was not particular trustworthy. From manual examination, it seemed to vary unpredictably with the data and was naturally susceptible to temporal spikes in keyword frequency which could affect the results long after a spike had disappeared.

We decided to take a more tested, information theoretic approach by exchanging our simple co-occurrence counting method for Pointwise Mutual Information (PMI, equation 1). We use PMI to measure the association between a given search keyword and keywords in the lexicon. Since the lexicon can become very large, we add an internal inverted index which maps a word in the lexicon to all words which it co-occurred with. Using this inverted index, we only calculate the PMI between the search term and words which, at some point, occurred in the same tweet as the search term.

$$pmi(x, y) = log(\frac{P(y|x)}{P(y)}) \quad (1)$$

Because the PMI object simply extracts the counts for a keywords in a tweet, it is very simple to update the PMI keyword extractor online. This functionality will be useful when we implement streaming.

2.2 Web Framework

To provide an interface for a user to interact with our keyword extractor, it was necessary for us to develop a web framework for both the server and client. We use the Play Framework as a infrastructure for our server side code such as the twitter API. We then required the ability to communicate between the server and client. To accomplish this, we used WebSockets to consume messages from the client and respond accordingly. We also utilize Play to have the server stream tweets via twitter4j.

There are two parts to the Javascript front-end. The first is the web socket functionality. The web sockets are used to send messages to the server to request data and to receive data from the server. These functions are wrapped around particular functionality (such as requesting a new tweet be streamed) to maintina simplicity and readability. The other functions control formating and displaying the streamed data and maintaining the dash4twitter GUI interface.

3 Evaluation

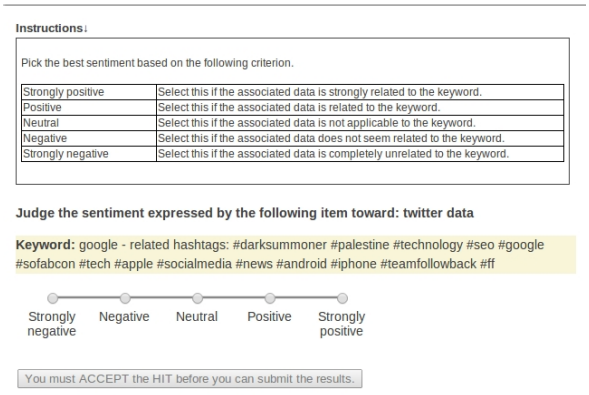

Figure 1: Example of a sentiment question for a user on MTurk

As there is no trivial method for testing a framework such as our project, we decided to use manual evaluations based on a survey. We structured our evaluations based on having workers evaluate the back-end of the framework through Amazon's Mechanical Turk (MTurk). We did a data collection based on predefined sentiment project they have available as a template. Figure 2 shows an example of how the data is presented to the worker along with the available options they can choose from. For each item, the user is shown a keyword with associated data that has been generated by

our keyword extractor. The associated data can be either mentions, hashtags, or unigrams. They are also provided five sentiment choices to select from. These choices available are strongly negative, negative, neutral, positive, and strongly positive. With each choice, there is criteria available that explains which choice the worker should select based on on the input. Using PMI, we prepared several keywords with their associated data beforehand. As we have no knowledge on the background of the users on MTurk, we selected general keywords that would hopefully be relevant to a broader range of people. Some examples of the words we used are ”facebook”, ”summer”, ”football”, ”playoffs”, and ”finance”. A total of 99 items were submitted for evaluation to MTurk. We requested each of these items to be completed by ten workers. By the end of the evaluations, we had 990 responses from our submission.

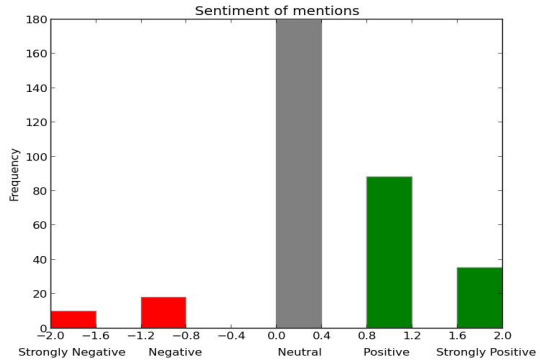

Figure 2: Histogram of overall sentiment scores for mentions

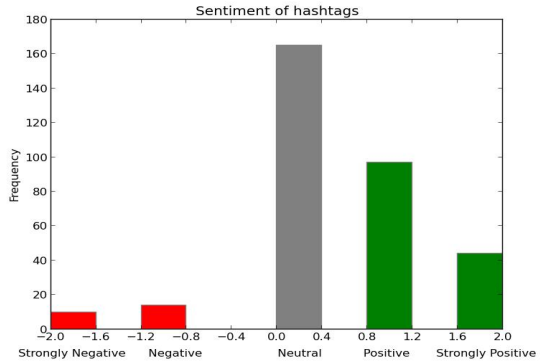

Figure 3: Histogram of overall sentiment scores for hashtags

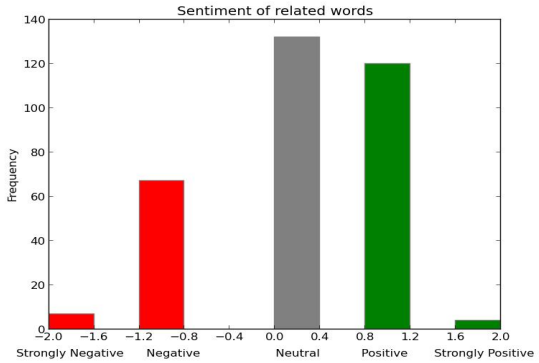

Figure 4: Histogram of overall sentiment scores for unigrams

4 Results

Figures 2 through 4 show the evaluations of the keyword extractor from MTurk workers. For each category (mentions, hashtags, unigrams), we summed the responses of each sentiment and colored the relative sentiments. ”Positive” and ”Strongly Positive” are colored green, ”Negative” and ”Strongly Negative” are colored red, and ”Neutral” is colored gray. The keyword extractor performed well with respect to hashtags and mentions, while the unigrams had a lower performance. There are several possible reasons why ”Neutral” was the highest for each category. One possibility is because there may have been no associated data for a given keyword. More specifically, for some of the predefined keywords we used, there was occasionally no hashtags, mentions or unigrams for a given word. This resulted because we wanted to treat each query word identically. Another possibility could have been because of the criterion for selecting ”Neutral” was too vague. Figure 5 shows the current interface for the framework. The location of this interface can be found at `http://www.cs.utexas.edu/~spryor/dash4twitter/version[2]/`. The user may enter an arbitrary amount of keywords they are interested in. Once they press ”Go” or Enter, four fields labeled as ”Keywords”, ”Tweets”, and ”Trends” appear beneath the site’s name. Each of these interactive fields will generate analytics based on the query of the user based on PMI running on a twitter stream. The codebase can be found at `https://github.com/spryor/dash4twitter.git`.

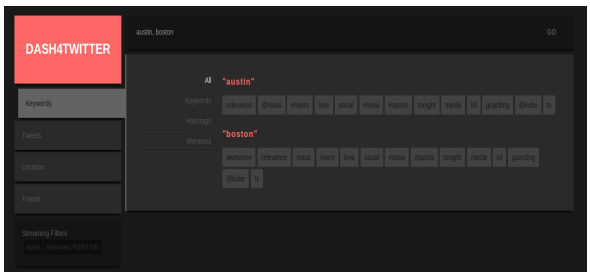

Figure 5: The interface for *dash4twitter*

5 Future work

While working on this project, there were many ideas had that we wanted to pursue. Unfortunately, time constraints prevented us from completing several of them. We initially were interested in focusing on generating visualization of twitter analytics. For instance, given a query word from the user, we wanted to generate a visualization of a time series analysis of the word based on its occurrence in tweets from the stream. This could provide insight on if an arbitrary word is trending. Another idea was to provide meta-data along with the generated data from PMI. This metadata, such a google search of the data, could help the user determine if the word is indeed related to the query word. We believe this could be useful because we had several instances were the generated data appeared unrelated to the query word. For example, we queried the word "google" and the keyword extractor returned "palestine". These two seem unrelated, however Google recently recognized Palestine as a state, which was a very popular topic on Twitter.

6 Conclusion

We have developed an initial web framework which is based on modern web tools coupled with an underlying NLP components. Our goal was to complete the web implementation first so we could focus on improving the back end. Unfortunately, this proved to be a non-trivial task. By using the Play Framework, we were backed by good software but hindered by limited documentation. A significant amount of time was spent on simply connecting the server to the client. Once this was completed, the issue of maintaining a twitter stream while serving clients became a difficult problem to solve. Overall, a significant amount of time was spent on building the web framework. After completion, we began on improving the back end code. Admittedly, the back end implementation has much room for improvement. However, the keyword extractor performed adequately with respect to the evaluations from MTurk. Although there are many things we were unable to finish, we have built a solid foundation for generating twitter analytics. The framework has been designed so anyone who is interested can utilize our code for their own needs while having the support of modern web software.

References

[Earle et al.2012] Paul S Earle, Daniel C Bowden, and Michelle Guy. 2012. Twitter earthquake detection: earthquake monitoring in a social world. *Annals of Geophysics*, 54(6).

[Hao et al.2011] Ming Hao, Christian Rohrdantz, Halldór Janetzko, Umeshwar Dayal, Daniel A Keim, L Haug, and Mei-Chun Hsu. 2011. Visual sentiment analysis on twitter data streams. In *Visual Analytics Science and Technology (VAST), 2011 IEEE Conference on*, pages 277–278. IEEE.

[MacEachren et al.2011] Alan M MacEachren, Anthony C Robinson, Anuj Jaiswal, Scott Pezanowski, Alexander Savelyev, Justine Blanford, and Prasenjit Mitra. 2011. Geo-twitter analytics: Applications in crisis management. In *Proceedings of the 25th international cartographic conference. Paris, France*.

[Marcus et al.2011] Adam Marcus, Michael S Bernstein, Osama Badar, David R Karger, Samuel Madden, and Robert C Miller. 2011. Twitinfo: aggregating and visualizing microblogs for event exploration. In *Proceedings of the 2011 annual conference on Human factors in computing systems*, pages 227–236. ACM.

[Mathioudakis and Koudas2010] Michael Mathioudakis and Nick Koudas. 2010. Twittermonitor: trend detection over the twitter stream. In *Proceedings of the 2010 international conference on Management of data*, pages 1155–1158. ACM.

[Mendoza et al.2010] Marcelo Mendoza, Barbara Poblete, and Carlos Castillo. 2010. Twitter under crisis: Can we trust what we rt? In *Proceedings of the first workshop on social media analytics*, pages 71–79. ACM.

[Pak and Paroubek2010] Alexander Pak and Patrick Paroubek. 2010. Twitter as a corpus for sentiment analysis and opinion mining. In *Proceedings of LREC*, volume 2010.

[Perera et al.2010] Rohan DW Perera, S Anand, KP Subbalakshmi, and R Chandramouli. 2010. Twitter analytics: architecture, tools and analysis.

In *MILITARY COMMUNICATIONS CONFERENCE, 2010-MILCOM 2010*, pages 2186–2191. IEEE.

[Sriram et al.2010] Bharath Sriram, Dave Fuhry, Engin Demir, Hakan Ferhatosmanoglu, and Murat Demirbas. 2010. Short text classification in twitter to improve information filtering. In *Proceedings of the 33rd international ACM SIGIR conference on Research and development in information retrieval*, pages 841–842. ACM.

[Suh et al.2010] Bongwon Suh, Lichan Hong, Peter Pirolli, and Ed H Chi. 2010. Want to be retweeted? large scale analytics on factors impacting retweet in twitter network. In *Social Computing (SocialCom), 2010 IEEE Second International Conference on*, pages 177–184. IEEE.