Social Unrest: Classification and Modeling, 229

Dan Saadati, Farah Uraizee, Tariq Patanam

Dec 16, 2016

1 Introduction

As social media rapidly becomes a podium for political opinions and a tool for the organization and facilitation of protests, a powerful stream of data documenting opinions and actions of individuals becomes readily available. This type of information can provide key social insights in predicting areas at risk of social unrest, which can be useful in scenarios prone to violence.

Tweets in particular are very suitable for analyzing this type of sentiment on a large scale. First, tweets give a first-hand account of what people are feeling. Second, they are tied geographically and temporally. Particularly at times of unrest, tweets are written in the moment. Third, they are conducive to scraping because tweets are often relevantly hashtagged. Finally, they can also model the situation in which social unrest spreads from one region to another as relevant tweets and hashtags such as "#ArabSpring" are spread from one region to another.

2 Related Work

Detecting tension or even sentiment in social media is a relatively recent problem and poses unique challenges as opposed to traditional sentiment analysis. One groundbreaking work in sentiment analysis of twitter was conducted by Pak and Paroubek who used distant learning for both subjective and objective sentiment. Subjective sentiment looked simply for emoticons such as ":)" and ":(" in order to classify sentiment as negative or positive while objective sentiment analyzed the tweets of major news outlets like the New York Times or Washington Post [2]. They report that both POS tagging and ngram models helped significantly in sentiment classification.

Focusing on tension detection is a much less explored problem. Burnap et. al find that one of the most significant challenges in dealing with Twitter social unrest sentiment analysis is the noise [1]. At any given time, a small amount of tweets during a social unrest event may actually be dealing with the event. Therefore, they use membership categorization devices (MCDs) to look for certain classes of words such as expletives or racist terminology.

Our approach in sentiment classification of social unrest brings a host of improvements. First, unlike Pak, Paroubek, and many others we rely solely on unigram models and both our approaches, Bag of Words and Bag of Clusters, display a significant improvment on the unigram baseline. Second, in contrast to Burnap and others, our classification system is shown to work broadly across global social unrest events.

We also looked to previous methods of understanding how tweets disseminate, which we believed to be a potential contributing factor to the onset of unrest. We uncovered that using appropriate stop words is vital to extracting the information from tweets when learning about the tweet networking problem, [4].

3 Technical Approach

3.1 Data Grabbing

We batch generate our data by querying for tweets in a certain time period and location. For example, in terms of our training data, we grab a combination of peaceful and social unrest data sets. As an example, we query for tweets from Ferguson and Baltimore during the duration of the protest and riots that happened there. For social rest, we query Ferguson and Baltimore during peaceful times, around a year before.

As we are coming off raw queried data, our data set construction procedure is important and differs for training and testing data. We first combine each collection of data C_i that corresponds to a time and location of social unrest, such as social unrest tweets from Baltimore, into one large pool of data P_{train1}. We then split this large pool into smaller subsets by uniformly randomly selecting tweets from the pool and placing them into subsets of size 100 tweets. Similarly for our label 0 data, we form P_{train0}, a pool of data that corresponds to tweets not related to social unrest. Therefore, our final training data set consists of first, n sets of tweets $S = (s_1, s_2, s_3, ..., s_n)$ where

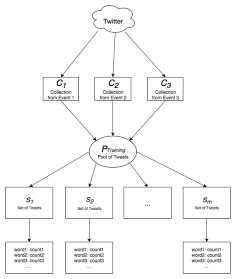

Figure 1: Raw twitter data extraction flow for training data

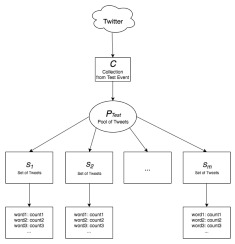

Figure 2: Raw twitter data extraction flow for testing data.

each set s contains 100 tweets $s = (\{t_1, t_2, ..., t_{100}\})$. All these n sets are labeled $Y = 1$ because they correspond to social unrest. Second, our data set also consists 523 sets of tweets $S = (s_1, s_2, s_3, ..., s_{523})$ labeled $Y = 0$ because they correspond to no social unrest.

The pooling of our training data ensures that our classification is ambivalent to location and time and rather focuses on the features corresponding to social unrest. Additionally, it ensures that we are ambivalent to specific times in a social unrest period. For instance, closer to a violent time in the social unrest we may have a higher distribution of certain words like "riot" or "violence" whereas closer to a nonviolent time we may have a different distribution of words such as more occurrences of "protest." In our formulation of the social unrest classification problem, we want to collectively label a whole chunk of tweets corresponding to an event as social unrest or not.

As aforementioned, we construct our testing data set slightly different. Here we do not want to be ambivalent to the time or location of tweets. We would like to classify tweets from Baltimore separately from tweets from Ferguson. Therefore, once we gather a collection of data C_i that corresponds to a specific time and location, we proceed by drawing our subsets $S = (s_1, s_2, ..., s_n)$ directly from C_i. Once we run it through our learning algorithm, we are able to classify whether collection C_i as a whole evaluates as social unrest ($Y = 1$) or not ($Y = 0$).

3.2 Preprocessing

The first step in the process of tokenizing our words is to remove arbitary punctuation. We remove a number of characters including quotes, exclamation points, and periods. Most importantly to tweets we remove the hashtag. This prevents for instance syntactically similar words like "#ferguson" and "ferguson" being tokenized separately.

The second step is generating stop words and sanitizing our data of them. While it is fairly easy to use a published set of stop words, in many cases, using such stop words is completely insufficient for certain applications. For example, in clinical texts, terms like "mcg" "dr." and "patient" occur almost in every document that you come across. So, these terms may be regarded as potential stop words for clinical text mining and retrieval. Similarly, for tweets, terms like #, RT, @username can be potentially regarded as stop words. In our preprocessing, we use a combination of minimal stop words and Twitter-specific generated stop words.

In order to generate the Twitter-specific stop words, we take a random stream of tweets within the United States to collectively represent a standard Twitter feed and tweet structure. We iterate over this sampling of tweets and keep track of the most frequently used words (which may include symbols like @ or #). We take the most common of these and add them to our stop word list for preprocessing.

3.3 Feature Extraction: Bag of Words

3.3.1 High Dimensional Vocabulary, Word Occurrence

In our first approach, we used a bag of words to define the features of our vector. First, we build a vocabulary $|V|$ consisting of high frequency words. We do this by going through our entire training data set D (collections of collections of tweets), mapping how often each word appears in D. We then set some constant k which is the minimum frequency of the word that we determine to be indicative of it being relevant in our training. Words that appear $< k$ times in D are not added to our dictionary of words. For instance, if in our training set the word "cat" appeared twice and $k = 3$, then cat would not be part of our vocabulary. Once we have gone through creating our bag of words and cut words that do not reach the frequency threshold, we create a training vector for

Figure 3: The most popular terms during social unrest events using our background subtracted Bag of Words model

each collection. This vector represents a multinomial event model, and because it focuses on only the most frequent words in social unrest situations, prunes out many irrelevant words.

3.3.2 Background Subtraction

One of the drawbacks of using Twitter to extract our datasets is that they will be noisy. Even during a significant event like the Ferguson protests, there remain a significant number of tweets about, for instance, going to McDonalds. To remedy this, we use a background subtraction method to filter our Bag of Words so that relevant terms are used in our vocabulary. For each word w in the bag of words we subtract the frequency of its occurrence in non-social unrest situations from its frequency in social unrest situations

$$f_D(w) = f_1(w) - f_2(w)$$

where $f_1(w)$ is the frequency of the word w in social unrest situations and $f_2(w)$ is the frequency of the word w in a normal situation.

3.3.3 Word2Vec to Reduce Dimension (Bag of Clusters)

Our original Bag-of-Words algorithm came at a few disadvantages. Our VC dimension was extremely high since we were treating semantically similar words as separate features with different weights. With the goal to simultaneously group synonymous, lemmatized, and similar words together, and then normalize their weight in our feature vector, we decided to integrate **word2vec** to our solution.

Using a pre-trained **word2vec** model, To generate our bag of clusters, we go through the vocabulary V generated by the training data. For each word in our vocabulary, we use a pre-trained **word2vec** model to retrieve a vector that semantically represents the word using neural layers. Similar to our Bag of Words model, we maintain counts of how many vectors we have for each of the words in V.

In order to reduce the amount of features required during classification, we use k-means clustering in order to group semantically similar words. However, this introduces the problem of finding how many clusters should be introduced when grouping words together. In order to resolve this, we go through the average silhouette distance for each word in V for multiple k values and find the k value such that the average silhouette distance is maximized. This ensures that words are optimally cohesive within their cluster as well as optimally separated to other clusters. The following equation gives us the average silhouette distance for varying k values:

$$max_{kvalue for k-means} \bar{s} = \frac{1}{m} \sum_{i=1}^{m} \frac{b(i) - a(i)}{\max a(i), b(i)}$$

where $a(i)$ is the sum of the euclidean distance from that word to each other word in the cluster and $b(i)$ is the lowest average dissimilarity between any other cluster is not a part of.

Now that the we have found the optimal k value, we continue with constructing our bag of clusters. Because we are bound to have clusters around meanings that do not represent social unrest, we set a threshold in removing clusters by counts of words assigned to that cluster. The remaining clusters are then used as our bag of clusters so that testing data points can be converted to a vector representation; each word is assigned to the semantically closest cluster and is then a feature of the data point.

4 Results and Analysis

4.1 Tuning SVM

We ran a series of experiments in order to tune our SVM. First, we tried an SVM with a linear basis kernel in order to reduce possible overfitting with the radial basis kernel. However, a linear basis kernel did not improve the results (achieving 75% accuracy on American social unrest verus the 90% with the RBF kernel). The RBF kernel likely did better because of the high dimensionality of our Bag of Words feature

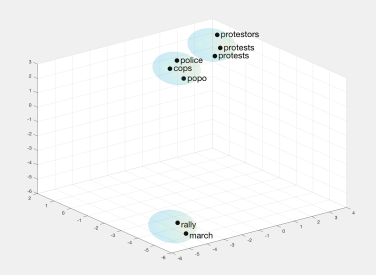

Figure 4: A visualization of the word2vec cluster feature space

vectors (typically on the order of 1000 features). Second, we tried to minimize the γ factor of our kernel $K(u, v) = exp(-\gamma||\mathbf{u}-\mathbf{v}||)$. In our default testing (results reported above), γ is set to be 0.001. Thereafter we tried setting γ to 0.00001 and to 0.01. Interestingly, minimizing γ severely impacted predicting on social unrest with an accuracy of 33% globally (versus the 66% reported for the default γ above). Increasing γ we correctly predicted social unrest with an accuracy of 91%. This demonstrates that contrary to our hypothesis that we were overfitting, the RBF-basis SVM was underfitting the data.

4.2 Bag of Words with no Background Subtraction

Our initial experiment used a basic Bag of Words feature vector and an SVM classifier. This involved running through the Bag of Word algorithm described in section 3.3.1, where we essentially took social unrest events and extracted features that weighted words by their frequencies (this was after having preprocessed our raw data to exclude stop words). In terms of results, we achieved a high rate of correctly determining events with social unrest, but we were also capturing a lot of false positives and incorrectly classifying events that had no social unrest as having social unrest. As to why this was happening, our hypothesis was that there were a lot of general terms used in situations of both social unrest and no social unrest and they were being weighted too high in our feature vector.

4.3 Bag of Words with Background Subtraction

In order to address the issue of having a very high false positive rate when determining social unrest, we ran experiments altering the Bag of Words to include the background subtraction approach described in section 3.3.2. Our results showed great progress, with the false positive rate turning almost to 0. Our precision with social unrest was no longer 100%, but still fairly high at around 90%. Overall, the hypothesis described above in section 4.2 proved to be true through this experiment. Eliminating popular words from no social unrest situations from our bag of words reduced the rate of incorrectly classifying peaceful situations as social unrest.

4.4 Bag of Words with Background Subtraction on Global Test Set

Our preliminary testing and validation sets contained largely American events and data. To experiment, we decided to test on events from international English-speaking areas. For example, events we tested on included the London riots and protests from areas such as South Africa, Ukraine, and Egypt. Our results indicated that our classifier was not strong on these events as it was on American events; we had around a 66% precision rate when it came to classi-

SVM Regression Results			
	Bag of Words (No BG Subtraction)	Bag of Words (BG Subtraction)	Bag of Words (BG Subtraction, Global)
Precision	1.000	0.905	0.662
Recall	0.614	0.988	0.984
F1	0.761	0.945	0.792

Table 1: Result of each experiment

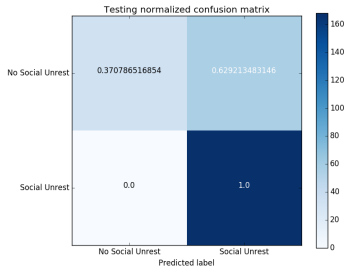

Figure 5: Confusion matrix for American data-set; no background subtraction on Bag of Words.

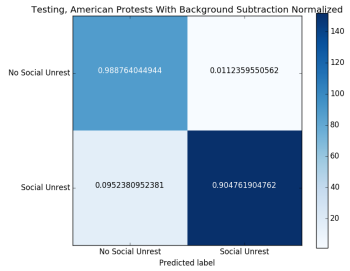

Figure 6: Confusion matrix for American data-set; background subtraction on Bag of Words.

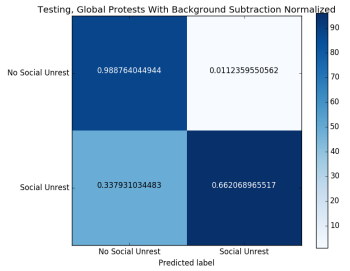

Figure 7: Confusion matrix for global data-set; background subtraction on Bag of Words.

fying social unrest. We believe this is largely due to regional overfitting. Most of our training data was of American origin, so naturally our feature vector contained a majority of American vernacular. Even if the international data we tested on was in English, different phrases and terms are preferred according to culture when it comes to situations of high social tension.

4.5 Bag of Clusters

Our Bag of Clusters, contrary to our hypothesis, did not achieve accurate results. Despite optimizing our K value using silhouette scores, it classified everything as social unrest. Taking a closer look at the top clusters the Bag of Clusters approach produced, many of the top clusters had insignificant words. For instance in terms of social unrest sentiment, words relating to food like "breakfast" do not really matter, but many of the top clusters contained exactly these kinds of words. In essence, our Bag of Clusters model was not prepared to deal with the amount of Twitter noise, but there is reason to believe that if the data were pruned better, the Bag of Clusters model would still achieve better results. For instance, one of the top clusters produced contained the words (alongside their counts) "seatbelted 2, untouched 2, safe 560, unharmed 0, hostages 2, alive 6, intact 2, ordeal 2, unscathed 1, safely 6." The optimal K value produced by maximizing silhouette score was K = 500 with a silhouette score with 0.025.

5 Conclusion

Originally, we approached our problem as one that was predictive in nature; given a set of tweets we wanted to determine whether it was indicative of an unrest event being imminent. We explored this approach with the baseline and realized that the nature of raw twitter data would make any potential tweets that could point to unrest occurring infrequent. We observed how most protests would have significant Twitter presence on their onset, most of the time before media outlets could begin to cover the story, meaning that live Twitter data would be the first source to determine if unrest was occurring. As it turns out, classifying tweets as indicative of unrest was a more possible, but still challenging problem, and so we shifted our focus to predicting social unrest at its onset.

We tuned and adjusted our approach several times throughout our implementation. The Twitter data was noisy and inconsistent from event to event, so feature extraction was especially important. We went through processes related to sanitizing the tweet and, generating Twitter-specific stop words. When we started with Bag of Words, our results demonstrated high potential in predicting accurately situations of unrest, but was clear that we still had a lot of progress to make. We built on the basic Bag of Words technique by adding a subtraction method, which ended up producing superior and reliable results.

However, we realized that the Bag of Words mode had setbacks by not being able to lemmatize/stem related words. We decided to implement the Bag of Clusters approach that uses word2vec in order to more accurately retrieve features from our Twitter data. By clustering around semantically words, we were able to significantly reduce the dimensionality of our feature vector, however our model wasn't accurate in classifying situations with no social unrest.

Overall, the results of the experiments conducted in this study show that there is high potential in both being able to determine the onset of social unrest and predicting it. After seeing the results of our model, we believe that the proper resources could provide a sophisticated model of social unrest classification that could be leveraged by both sides: those who want to voice their discontent and those whose job it is to contain it.

6 References

[1] Pete Burnap et al. "Detecting tension in online communities with computational Twitter analysis". In: *Technological Forecasting and Social Change* 95 (2015), pp. 96–108.

[2] Alexander Pak and Patrick Paroubek. "Twitter as a Corpus for Sentiment Analysis and Opinion Mining." In: *LREc*. Vol. 10. 2010, pp. 1320–1326.

[3] Tauhid Zaman, Emily B Fox, Eric T Bradlow, et al. "A Bayesian approach for predicting the popularity of tweets". In: *The Annals of Applied Statistics* 8.3 (2014), pp. 1583–1611.

[4] Tauhid R Zaman et al. "Predicting information spreading in twitter". In: *Workshop on computational social science and the wisdom of crowds, nips*. Vol. 104. 45. Citeseer. 2010, pp. 17599–601.