Sentiment Analysis and Opinion Mining of Microblogs

Kush Shah, Nasir Munshi, Pavan Reddy
CS 583 - Data Mining and Text Mining
University of Illinois at Chicago, USA
kshah57@uic.edu, mmunsh4@uic.edu, preddy4@uic.edu

May 5, 2013

Abstract

Microblogging is a very common mode of communication among Internet users. Microblogs are real time content published by people and this content is generally laden with personal opinions about a variety of aspects in everyday life. This makes microblogs a rich source of data for opinion mining. Here, we use a corpus from the popular microblogging website, Twitter [1]. We consider microblogs from the period before the presidential elections in the United States of America in 2012 to analyze the collective sentiment of the microbloggers, against and in favor of each presidential candidate. We classify the microblogs to positive and negative opinion classes and we use machine learning classification techniques achieve this.

I. Introduction

Microblogging is a very common and powerful mode of communication among internet users. Microblogs often reflect personal opinions and are very useful for opinion mining. We considered a corpus from Twitter. Microblogs in Twitter are called Tweets. We choose tweets from the period of presidential elections in United States of America in 2012. We mainly try to classify the tweets into political opinions against and in favor of the presidential candidates, *Barack Obama* and *Mitt Romney*. We used various feature selection techniques and classification algorithms. The best results were obtained by using n-gram features with support vector machines (SVM) classifier. The Twitter corpus used is manually annotated for sentiments and we use this as a gold standard for evaluation of precision, recall and f-score of our classification.

II. Related Work

Sentiment analysis of Tweets is a growing area of research. Since, twitter has a limit of 140 characters per post, this problem is essentially a sentence level analysis problem. There has been work done in sentiment analysis of Twitter corpus by machine learning classification methods [Pak & Paroubek, 2010].
There has been work done specifically for political opinion mining from tweets [Maynard & Funk, 2011]. Apart from using differnt features and classifiers, there are variety of methods used like use of emoticons [Go et al., 2009], use of opinion reversal words etc for identifying sentiments. we have used some similar ideas in our data processing and feature selection.

III. Algorithm

Our approach consisted a varitey of ideas borrowed from the realms of natural language processing, information retrieval and machine learning. The algorithm mainly consists of the following steps,

1. Data processing
2. Features
3. Training the classifier

Data processing

The corpus has considerable amount of metadata such as date, time, identity number etc. and to extract natural language content we need to subject the data to a series of processing steps before the data can be used to extract

features and train a classifier. Here is a sample of the raw data before data processing.

```
“315,41199, AM 0:26:17, News Analysis:
In Second Debate Obama Strikes Back
(NY Times):  Share With Friends:  | |
Top News - ...  http://t.co/PXoW7wiD
@NCAAlarms,-1,0,1,,”
```

1. **Stop words removal** - We used the Natural Language Toolkit's (NLTK) [2] stopword corpus for the English language to remove the stop words from the data. This helps eliminating the most common stop words from being included in the computation of n-grams and feature extraction.

2. **Stemming** - Twitter data is generally used with informal language and it includes internet jargons, slang and contemporary spellings. We were very frugal in stemming so as to not risk truncating words and losing out on potential features. We employed basic stemming e.g. use of apostrophes.

3. **Spelling correction** - As Twitter users generally use informal language, there are often incorrect spellings in tweets. We used Jazzy Open Source Spell Checker [3] to detect incorrect spellings in the tweets and replace them with the closest word from the English dictionary.

4. **Entities** - The entities we used were *Barack Obama* or *Mitt Romney*. But, these entites are addressed to by various names for e.g. *Barack Obama* is generally addressed to as *Mr.President, Barack, Obama etc*. So, we normalised this by replacing the possible names people use to address the entities by either *Obama* or *Romeny*.

5. **Emoticons mapping to sentiments** - There are a multitue of emoticons that are used frequently in Twitter. We used an approach inspired by the method used by [Go et al., 2009] and mapped some emoticons to positive and negative sentiments and discarded emoticons that are ambigious or irrelevant to sentiments.

Emoticons mapping	
Positive sentiment	Negative sentiment
:)	:(
:-)	:-(
:D	

6. **Part of Speech tags** - We experimented with both NLTK and OpenNLP [4] Part Of Speech tagger with a heuristic that adjectives and/or adverbs are generally used to articulate opinions in natural language. The data was tokenized by spaces and the tokens were subject to the taggers. We also experimented by excluding all data except the entity, adjectives, adverbs and words such as *not, couldn't* etc. which generally indicate a reversal of sentiment. This is similar to the way opinion reversing words were used by [Maynard & Funk, 2011]

7. **Filtering** - Tweets contained a lot of metadata and quite a bit of noise which were removed. The following data was filtered,

   - Identity numbers, date, time etc. of the tweets
   - Irrelevant tags
   - Hyperlinks
   - #tags e.g. *#msnbc2012*
   - Twitter handles e.g. *@pavanred*
   - punctuation, special characters and digits

8. **Encoding** - There are a few tweets that aren't in English. These tweets contain UTF-8 encoded characters for e.g. *naïve* . These characters were excluded from the tweets and only ASCII encoded characters were retained. All content was also transformed to lowercase. These characters interfered with the classifiers.

After subjecting a tweet to this data processing process, we are left with only the natural language content of the tweet and the human annotated sentiment that is used by the classification algorithm in supervised learning. After data processing, the sample tweet we considered earlier about would be transformed to,

```
‘‘news analysis second debate obama
strikes back ny times share friends top
news,-1’’
```

1 for positive, -1 for negative are the annotations used for sentiment classification.

Features

N-gram features - The processed data is used to extract features that will be used to train our classifier. We have experimented with *unigrams, bigrams, trigrams and combination of unigrams and bigrams*. The data was tokenized by spaces using NLTK and and these tokens were subject to NLTK to generate n-grams.

Part Of Speech features - Since the language used in Twitter is generally informal, part of speech tagging isn't very accurate for tweets. We used both NLTK Part Of Speech tagger and OpenNLP Part Of Speech tagger along with a heuristic that adjectives and/or adverbs, *JJ, JJR, JJS, RB, RBR and RBS* in the PennTree bank tagset, are generally used to articulate opinions in natural language. So, we further process the data by excluding all data except the entity, adjectives, adverbs and words such as *not, couldn't* etc which generally indicate a reversal of sentiment. This is similar to the way opinion reversing words were used by Maynard and Funk [Maynard & Funk, 2011]. After using part of speech taggers, we experimented with unigrams, bigrams and combination of unigrams and bigrams.

We experimented with term frequency-inverse document frequency (tf-idf) where we considered only the most frequent terms ordered by tf-idf. We used the absolute approach of conidering all the n-grams as features as well.

Sentiment Lexicon features - We used the terms in positive and negative opinion word lists [Hu & Liu, 2004] as features for classification.

Training the classifier

We experimented various combinations of features, classification algorithms and test options. As mentioned in the Feature extraction section, we extracted various feature sets and used these to construct the feature vectors. These were used to train the classifiers. We experimented with 4 different classifiers,

- Multinomial Naïve Bayes
- Logistic regression
- Random forest
- Support vector machines (SVM)

We used these classifiers from Weka, a popular suite of machine learning software [5] and LibSVM, a library for Support Vector Machines [6]. We also experimented with one-vs.-all classification strategy with SVM.
We experimented with the evaluation methods, 70-30 percent split and 10-fold cross validation.

IV. Experimental Results

We used various combinations of features and classifiers and here are the different experimental results using both Weka and LibSVM. Table-1 shows the results of using 6000 most frequent unigrams without removal of stopwords used with Naïve Bayes Multinomial classifier using a 70-30% evaluation method.

We got no significant changes in results when we experimented with used Logistic regression and Random forest classifiers.

Table-2 shows the results of using 8000 unigram features ordered by tf-idf using Naïve Bayes classifier. We used 10-fold cross validation for the follwing results.

Table 1: *Confusion matrix - 6000 most frequent unigrams without stopword removal as features using a Multinomial Naïve Bayes classifier*

Barack Obama			
Sentiment	precision	recall	f-measure
Positive	63.9	06.3	11.4
Negative	46.6	95.1	62.6
Mitt Romney			
Sentiment	precision	recall	f-measure
Positive	53.3	09.9	16.7
Negative	47.5	94.5	63.2

Table 2: *Confusion matrix - 8000 unigram features ordered by tf-idf using Naïve Bayes classifier*

Barack Obama			
Sentiment	precision	recall	f-measure
Positive	37.8	30.5	33.8
Negative	33.9	60.4	43.4
Mitt Romney			
Sentiment	precision	recall	f-measure
Positive	49.0	75.8	60.2
Negative	44.9	21.1	28.7

We experimented with positive and negative terms in opinion word lists [Hu & Liu, 2004] as features for classification.

Table 3: *Confusion matrix - Opinion word lists as features using using Logistic regression classifier*

Barack Obama			
Sentiment	precision	recall	f-measure
Positive	52.0	24.6	33.4
Negative	46.6	39.0	42.5
Mitt Romney			
Sentiment	precision	recall	f-measure
Positive	34.8	14.5	20.5
Negative	41.6	95.5	58.0

The results obtained by employing part of speech tagging for feature selection are shown in Table-4.

Table 4: *Confusion matrix - Parts of speech tags (adjectives/adverbs tags) used for feature selection and using Naïve Bayes classifier*

Barack Obama			
Sentiment	precision	recall	f-measure
Positive	27.7	03.0	05.4
Negative	33.7	08.7	13.9
Mitt Romney			
Sentiment	precision	recall	f-measure
Positive	30.8	01.8	03.4
Negative	49.8	95.5	58.0

We then trained the model on the entire training corpus and evaluated the test data by using the combination of unigrams and bigrams as features and SVM classifier employing a one-vs.-all classification strategy . Table-5 shows these results. We used LibSVM for this experiment.

Table 5: *Confusion matrix - Test data - unigram & bigram features using SVM classifier and one-vs.-all classification strategy*

Barack Obama			
Sentiment	precision	recall	f-measure
Positive	41.58	48.79	45.05
Negative	46.86	55.29	50.72
Mitt Romney			
Sentiment	precision	recall	f-measure
Positive	33.70	39.74	36.60
Negative	54.79	57.17	55.95

V. Conclusion

Based on the experimental results we can infer that,

- If the data is not balanced then there can be skewed results e.g. the test data performed better for negative classification because the data was not balanced.

- Stop word removal improved the result significantly.
- Results obtained by using all n-grams as features were better than using only some features ordered by tf-idf.
- Using spelling auto-correction did not show any significant improvement in peformance.
- Using positive and negative opinion word lists as features produced mediocre results.
- N-gram features obtained by adjective and/or adverb parts of speech in tweets performed quite poorly when compared to using regular n-grams features.
- Combination of unigram and bigram features perform better than using n-gram features separately.
- One-vs.-all classification strategy performed better than when we tried classifying into more than two classes.
- Support Vector Machines produced better results when compared to Naïve Bayes, Logistic Regression and Random Forest classifiers.

VI. Future work

There is scope for lot of work further in political opinion mining of tweets. Future work can involve,

- User specific sentiment inference, since political sentiments of most people do not change often, we can identify sentiments using superlative adjectives/adverbs to infer the sentiment of a particular user in case of ambiguity such as a comparative adjective/adverb involving multiple entities.
- To experiment with semantics, that is to create triples involving entities, users, sentiments etc. Similar ideas are used in [Maynard & Funk, 2011]

References

[Go et al., 2009] A. Go, R. Bhayani, L. Huang: Twitter Sentiment Classification using Distant Supervision, Technical report, Stanford Digital Library Technologies Project, 2009.

[Maynard & Funk, 2011] D. Maynard, A. Funk: Automatic detection of political opinions in Tweets, ESWC Workshops, pages 88 -99, 2011.

[Pak & Paroubek, 2010] A. Pak & P. Paroubek: Twitter as a Corpus for Sentiment Analysis and Opinion Mining, LREC, 2010.

[Liu, 2012] Bing Liu: Sentiment Analysis and Opinion Mining, Morgan & Claypool Publishers, 2012.

[Hu & Liu, 2004] M. Hu and B. Liu, KDD, 2004.

[1] Twitter, Microblogging website `http://www.twitter.com/`

[2] Natural Language Toolkit 2.0 (NLTK) `http://nltk.org/`

[3] Jazzy Open Source Spell Checker `http://jazzy.sourceforge.net/`

[4] OpenNLP, Apache Software Foundation `http://opennlp.apache.org/`

[5] Weka 3, machine learning software suite `http://www.cs.waikato.ac.nz/ml/weka/`

[6] LibSVM, a library of SVM `http://www.csie.ntu.edu.tw/~cjlin/libsvm/`

[7] TweetNLP and POS tagging `http://www.ark.cs.cmu.edu/TweetNLP/`