Available Online at www.ijcsmc.com

International Journal of Computer Science and Mobile Computing

A Monthly Journal of Computer Science and Information Technology

ISSN 2320–088X

IJCSMC, Vol. 3, Issue. 8, August 2014, pg.49 – 56

RESEARCH ARTICLE

'Good' versus 'Bad' Opinion on Micro Blogging Networks: Polarity Classification of Twitter

[1] Haziq Jeelani, [2] Khushal Singh

[1,2]Department of Computer Science and Engineering,

Galgotias University, Uttar Pradesh, India

[1] haziqjeelani@gmail.com, [2] khushal.9@gmail.com

ABSTRACT:
Information-gathering has always been an important part to find what other person is thinking. Millions of users tweet on different aspects of life every day. Therefore micro blogging websites are a very good source for polarity classification. We introduce a novel approach which automatically classifies the polarity of Twitter message. These messages so called tweets are classified as positive or negative or neutral. These results are useful for the customers or any general user who wants to research about the polarity of products before purchases, or it can be useful for the companies that want to analyze the reviews from people of their brands in the market. Most of the previous research on classifying the polarity of messages has tried to achieve some good results but have ignored the neutral tweets which lead to wrong polarity classification, so we have tried to solve this issue in our project. We present an approach for classifying the polarity of tweets using machine learning algorithms using a novel feature vector. Our training data contains publically available tweets which are obtained using twitter API's available. The following report shows the steps for preprocessing the dataset to achieve high accuracy. The novel feature vector of weighted unigrams that are used to train the machine learning classifiers is the main contribution of our project.

Categories and Subject Descriptors
[Data Mining]: Machine Learning Algorithm, Statistical Learning, Natural Language Processing

Keywords
Polarity Classification, Twitter, Opinion Mining, Unigrams

1. INTRODUCTION
Micro blogging service like Twitter is very popular where people post status messages so called tweets. These tweets express reviews about various topics, for example about any product, organization, government or any entity. Researchers or Analyst uses polarity classification to research about services or products to study the market. Market analysis is done by companies to fetch public opinion about their products. In case of newly launched products, companies preferably retrieve the user's review to know about the product and user experience.

It is very difficult to manually collect enough dataset to train a polarity classifier for the tweets as a wide range of topics are discussed on Twitter. So, we have used publically available tweets dataset which are obtained via distant supervision proposed in [6]. However, the following dataset only consists of positive and negative tweets. We have used neutral dataset provided by [7]. We ran the machine learning classifiers i.e. Entropy, Naïve Bayes and Support Vector Machine trained o the positive and negative twitter datasets and the neutral tweets Maximum against the test set of tweets.

There has been a large amount of research in the area of polarity classification. Most of it has focused on analyzing larger pieces of text, like reviews [5]. Tweets are different from reviews because of their purpose, whereas the reviews represent summarized thoughts of authors; tweets are casual and limited to 140 characters of text. Mostly, tweets are not as thoughtfully tweets are reviews are composed. But, still tweets offer many companies an additional avenue to gather feedback from people. Past research on analyzing blog posts includes [4]. Pang et al. [3] have worked on different classifiers on movie reviews to analyze the performance. The research and work of Pang et al. has served as a base for many authors and have used the techniques provided in their paper across different domains. Supervised learning mainly requires hand-labeled training data in order to train a classifier.

To visualize the utility of Twitter-based polarity classification tool, we have built a web application tool. This can be used by companies and individuals who are interested to research polarity on any topic.

1.1 Defining the Polarity

For research purpose, we define polarity to be a "personal positive (good) or negative (bad) opinion" and where there is absence of positive and negative opinion, we treat it as a neutral opinion.

Table1. Example Tweets

Polarity	Keyword	Tweets
Positive (Good)	Moto G	I love my new mobile: Moto G
Neutral	World cup T20	World Cup T20 starts this Friday
Negative (Bad)	BJP	Don't vote for BJP.

1.2 Characteristics of Tweets

There are many unique attributes of Twitter messages, which differentiate our work from previous research:

Language Model The frequency of slang and misspellings in tweets is much higher as compared to the other domains. Twitter users post tweets from many different media, such as laptops, mobiles, iPad etc.

Length 140 characters are the maximum length of the Twitter message. This is different from the past polarity classification research that focused on analyzing longer bodies of works such as any reviews or movie reviews.

Domain Twitter users post tweets about a variety of topics which is different from other sites which are specified to a particular domain. This differs from the past research, which only focused on specific domain such as mobile reviews, movie reviews etc.

2. RELATED WORK

Polarity Classification is a growing field of Natural Language.

The research in Polarity Classification started with the classical machine learning algorithms like Support Vector Machine, Maximum Entropy, Naïve Bayes etc. using intuitive features like unigrams, bigrams, position of words, parts of speech information etc. However, these approaches are heavily dependent upon the given training data, and therefore can be very limited for Polarity Classification due to out of vocabulary words and phrases, and different meanings of words in different contexts. Due to these problems, several methods have been investigated to use some seed words for extracting more positive and negative terms with the help of lexical resources like WordNet etc., for instance, SentiWordNet, which defines the polarity of the word along with the intensity.

We have used the ideas proposed in [9] where the author classifies the tweets using unigram features and trained the classifiers on data obtained using distant supervision. Read[11] shows that using emoticons as labels for positive and polarity is effective for reducing dependencies in machine learning techniques and this idea is heavily used in [9]. The research of various machine learning techniques in the movie reviews domain was done by Pang and Lee [3]. The recent work by Mass et. al. (Maas et. al., 2011) on using latent concept models presented a mixture model of unsupervised and supervised techniques to learn word vectors capturing semantic term-document information along with the sentiment content.

Most of the previous methods and researches have not taken into account neutral tweets which lead to wrong classification and our project tries to solve this problem by including neutral tweets in the training dataset and thus using a novel feature vector approach to train the machine learning classifier and tries to classify a give tweets as positive or negative or neutral.

3. APPROACH

Our approach is to use various machine learning classifiers and feature extractors. The machine learning classifiers are Maximum Entropy (MaxEnt), Support Vector Machines (SVM) and Naïve Bayes (NB). Unigrams are the feature extractors with weighted positive and negative keywords. We have built a framework that treat these classifiers and feature extractors as two different components. This framework allows easily trying out different combinations of classifiers and feature extractors.

3.1 Query Term

We normalize the effect of query terms. Our assumption is that the user wants to perform polarity classification about the product and not of a product. If a query term has a good/bad polarity by itself, this will bias the results.

3.2 Emoticons

The training process makes the use of emoticons as noisy labels, it is important to discuss the role they plain in classification. We will discuss in detail the test set and training set in the Evaluation section. We have stripped the emoticons out from the training data. If we leave the emoticons in, there will be negative impact on the accuracies of the MaxEnt and SVM but less impact of the Naïve Bayes classifiers. Stripping out the emoticons caused the classifier to learn from the other features like unigrams present in the tweets. The difference lies in the mathematical models and feature weight selection of SVM and MaxEnt. The classifier uses the non-emoticon features to determine the polarity.

3.3 Feature Reduction

There are many unique properties of Twitter language. We took the advantage of the following properties to reduce the feature space:

Usernames Users mostly include Twitter usernames in their tweets so as to direct their messages. A standard is to include @ symbol before the username like @param107. Equivalence class token (AT_USER) replaces all the words that start with @ symbol.

Usage of Links Mostly users include links in their tweets. So, we have used an equivalence class for all the URLSs. That is, we convert the URL like http://blankurl.com/abtetxt to the token “URL”

Stop words Stop words or filler words such as “is”, “a”, “the” used in the tweets does not indicate any polarity and therefore need to be filtered out.

Repeated letters For example, if you search “happy” with an arbitrary number of p’s in the middle (e.g. happppy, happppppppppy, happppppppppppppppy) on Twitter there will most likely be a nonempty result set. We use preprocessing so that that letter occurring more than two times in a row is replaced by two occurrences. In the example above, these words can be converted into token “happpy”.

Slang language is ignored because it’s very difficult to analyze slang. For example, “I m w8g 4 ma new fone, its alrdy 2 days L8 delivrd”, so this example shows spelling mistakes but it’s the slang language, to to get some polarity out of slang is very difficult. Every user has his/her own style of tweeting, So we have tried to fetch only English words which has confined meaning in English dictionary, leaving behind slang.

Table 2 shows these feature reductions and their effects. These reductions shrink the feature set down to 8.72 % of its original size.

Table2. Effect of Feature Reduction

Feature Reduction Steps	# of Features	Percentage of Original
None	266390	100.00 %
Username/ URL/ Repeated Letters	103397	34.80 %
Stop Words	23680	8.78 %
Final	24309	8.72 %

3.4 Feature Vector

After preprocessing the training dataset which consists of 9570 positive tweets, 9690 negative tweets and 2472 neutral tweets, we compute the feature vector as below:

Unigrams Table 2 shows at the end of preprocessing with 24309 features which are unigrams and each of these features have equal weights.

Weighted Unigrams Instead of weighting each of the unigrams equally, we introduced bias by weighting the positive and negative keywords more than other features in the feature vector. We used [11-12] as my focal point in order to weight the positive and negative keywords more compared with the remaining features.

4. MACHINE LEARNING METHODS

We tested different classifiers Maximum Entropy, Support Vector Machines and Naïve Bayes.

4.1 Maximum Entropy

The idea behind Maximum Entropy is that one should prefer the most uniform models that satisfy a given constraint [2]. MaxEnt are feature based models. It is the same as using logistic regression to find distribution. Maximum Entropy makes no independence assumptions for its features, unlike Naïve Bayes Classifier. The model is represented by

$$P_M(c/d,\gamma) \; = \; exp[\sum i \; y_i f_i(c,d)] \;\; / \sum c' \,\mathrm{exp}[\sum i \; y_i f_i(c,d)]$$

In this equation of MaxEnt, c is the class, d is the message (tweet) and y is a weight vector. The weight vectors decide the importance of a feature in classification. The higher the weight means that the feature vector is a strong indicator for the class. The weight vector is calculated by numerical optimization of the y_i's so as to maximize the conditional probability. We have used Python NLTK library to train and classify using the Maximum Entropy classifier method. For training the weights we have used conjugate gradient ascent.
Theoretically, MaxEnt performs better that Naïve Bayes classifier because it handles the feature overlap in a better way. Naïve Bayes can still perform well on variety of other problems [2].

4.2 Support Vector Machines

Another popular classification technique is Support Vector Machines. We have used libsvm [17] library. The vectors are of size n are our input data. Each entry in vector corresponds to the existence of a feature. In the unigram feature extractor every feature is a single word found in a message (tweet). If the feature is found, the value is 1 showing positive response, but if the feature is absent, then the value is 0 which shows a negative response. We use feature presence so as to oppose a count that we do not have to scale the input data which speeds the overall processing [1].

4.3 Naïve Bayes

Naïve Bayes is a model which works well on text categorization [8]. We have used multinomial Naïve Bayes model. Class c* is assigned to tweet (d) where

$$c* = argmac\_c \; P\_NB \; (c/d)$$

$$P_{NB}\left(\frac{c}{d}\right) := \left(P(c) \sum\nolimits_{i=1} m \frac{P\left(\left(\frac{f}{(c)^{ni(\mathrm{d})}}\right)\right)}{P(d)} \right)$$

In this formula, f represents a feature and Ni (d) represents the count of fi found in the d tweet. There are total of n m features. We have used Python based Natural Language Toolkit [10] library to train and classify using the Naïve Bayes classifier.

5. EVALUATION

5.1 Experimental Setup

Publically available data sets of twitter messages with polarity [11-12] are publically available. We have used the combination of these two datasets to train the machine learning classifiers. For the test dataset, we randomly choose 5000 tweets which were not used to train the classifier. The details of the test data and training data are explained in Table 3.

Table3. Example Tweets

Dataset	Positive	Negative	Neutral	Total
Training	9570	9690	2472	21722
Test	Randomly chosen Tweets			5000

Twitter API has a parameter that specifies that in which language you want to retrieve the tweets. We have set the parameter to English (en). Thus, our classification will only work on tweets in English because the training data is English-only. We have also built a local web interface which searched the Twitter API for a given keyword for the past one day or seven days and fetches those results which is then subjected to preprocessing as specified in Section 2.3. The filtered tweets are fed into the trained classifiers and the resulting output is then shows as a graph in web interface.

6. Results

We explore the usage of unigrams and weighted unigrams features and Table 4 summarized the results.

Table4. Classifier Accuracy

Features	Max Entropy	SVM	Naïve Bayes
Unigrams	60.35	63.90	66.82
Weighted Unigrams	70.42	80.10	78.52

Unigrams Unigram feature vector is the simplest way to retrieve features from a tweet. The machine learning algorithms gives average performance with this feature vector.

The main reason for the average performance might be due to smaller training dataset of 20000+ tweets. If one could get hold of about millions of tweets and then train these classifiers, the accuracy will improve substantially. Twitter API has a disadvantage that it only places 150 unauthorized requests per hour and therefore one can download only 3600 tweets per day via labeled tweets IDs from the publically availably tweets ID dataset.

Weighted Unigrams In this approach, we took the advantage of the fact that makes sense to weight the positive and negative keywords more than the other words whole trying to classify the polarity of a tweet and this trick produced competitive accuracy as shows in Table4. SVM performed the best as expected with the accuracy of 80.10 % and surprisingly Naïve Bayes outperformed the Maximum Entropy by a margin i.e. 78.52 % to 70.42@. This is in accordance with the results shown by Pang and Lee [3].

Figures 1-5 shows the twitter polarity classification

Of topics 'Narendra Modi' 'Arvind Kejriwal', 'Rahul Gandhi', 'India World Cup T20', 'Pakistan World Cup T20' between 22-29 March 2014 using Support Vector Machine Classifier.

Keyword = "Narendra Modi"

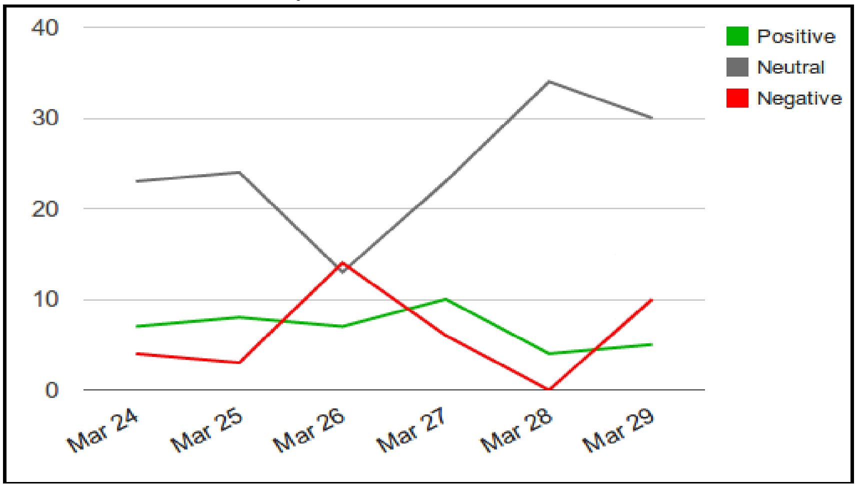

Figure1. Polarity Classification of "Narendra Modi"

It can be noted that famous politician "Narendra Modi" has a good review. He has constant positive review on twitter but between March 25 - March 27, 2014 he has got a lot of negative tweets which can be seen in the graph. It can help BJP government to read the public opinions and improve upon their campaigning. They can actually analyze why public is in favor of them and why public is against them by reviewing all these tweets.

Keyword = "Arvind Kejriwal"

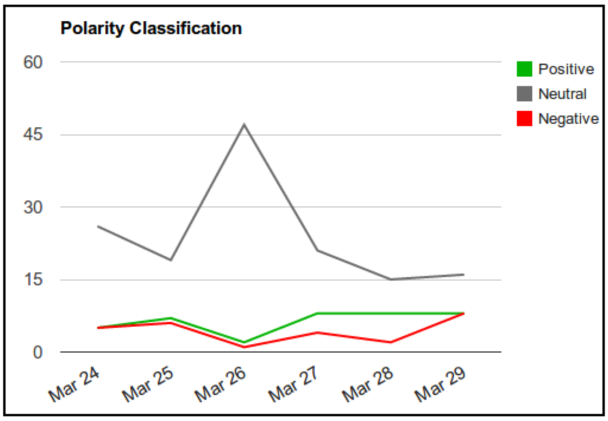

Figure2. Polarity Classification of "Arvind Kejriwal"

It is noted that Arvind Kejriwal has almost equal polarity which shows that 50-55% are in favor of him and 40-45% are against him. Tweets from different states can be fetched and see in which state, people are following "AAP" and where in which states people are against them. Accordingly they can improve their strategy, which will help them in changing people's mindset and have a good positive impact all over the world.

Keyword = "Rahul Gandhi"

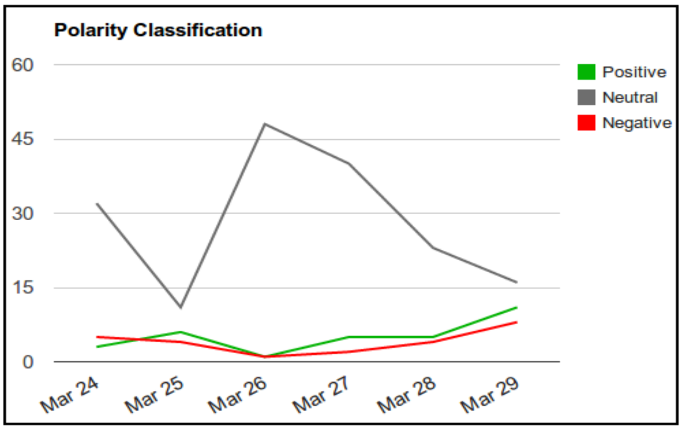

Figure3. Polarity Classification of "Rahul Gandhi"

From the above figure, it can be seen that "Rahul Gandhi" has very low positive and negative polarity. Users are not much tweeting about Rahul Gandhi and the results achieved so far also shows that he doesn't have good polarity among the micro blogging sites.

Keyword = "India World Cup T20"

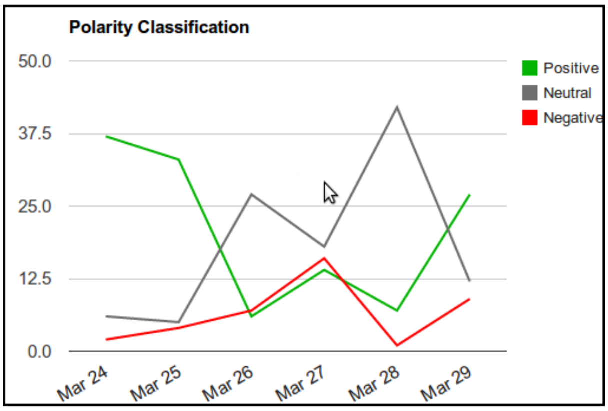

Figure4. Polarity Classification of "India World Cup T20"

Keyword = "Pakistan World Cup T20"

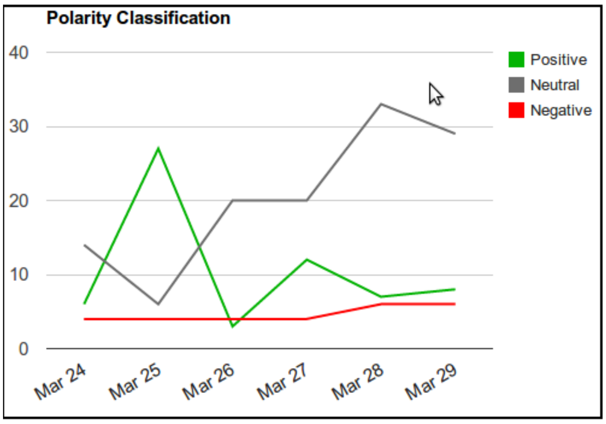

Figure5. Polarity Classification of "Pakistan World Cup T20"

It is noted that Indian team were having a lot of negative review before the world cup started, But after their three wins in the World Cup T20 have lifted their positive polarity and it can be seen in Figure 4, same is with Pakistan, after winning two matches in a row their positive graph is also lifted which was earlier down due to their loss to India.

7. FUTURE WORK

Machine learning techniques perform well for classifying polarity in tweets. We believe the accuracy of the system could be still improved. Below is the list of ideas we think could help the classification:-

Bigger Dataset The training dataset in order of millions will cover a better range of twitter words and therefore provide a better unigram feature vector which results in overall improved model.

Internationalization Currently we focused only on English tweets but twitter has a huge world wide audience. It should be possible to use our approach to classify in other languages as well with a language specific positive/negative keyword list.

Semantics The algorithms classifies the overall polarity of tweets. The polarity of the tweet may depend on the perspective that you are interpreting from the tweet. For example "India beats Bangladesh in warm up match :)"; the polarity is positive for Indian people and negative for Bangladesh. In this case, the semantics may help Using the semantic role label indicates which noun is associated with the verb and then classification will take place accordingly. This may allow "Bangladesh beats India :)" to be classified differently from "India beats Bangladesh :)".

8. Conclusion

Using a novel feature vector of weighted unigrams, we have shown that machine learning algorithms such as Support Vector Machines, Naïve Bayes and Maximum Entropy achieve competitive accuracy in classifying tweet polarity.

ACKNOWLEDGEMENTS

We would like to acknowledge "Manoj Rajora" for his insightful introductory lectures on classifying tweet sentiment which helped us immensely in implementing this project.

We would also like to thanks "Dr. Abhay Bansal", HOD of Computer Science, Amity University for providing us excellent labs and all required technical assistance and useful guidance.

References

1. D. O. Computer, C. Chung chang, C. Wei Hsu and C. jen Lin. "*A practical guide to support vector classification*". Technical report 2009

2. K. Nigan, A. Mccallum, J. Lafferty. "*Using Maximum Entropy for text classification*". In IJCAI-2010. Workshop on Machine Learning for Information Filtering, Pages 62-68, 2010.

3. B. Pang, S. Vaithyanathan and L Lee." *Thumps up? Sentiment classification using machine learning techniques".* Published in the Conference on Empirical Methods in Natural Language, pages 79-85, 2002

4. G. Mishne "*Experiments with mood classification in blog post*" In the workshop on Stylistic Analysis of Text for Information Access, 2005

5. Richard M. Tong "*An operational system for detecting and tracking opinions in on-one discussions*. Working Notes of the ACM SIGIR 2011 Workshop on Operational Text Classification, pages 1-7, 2011

6. Alec Go, Lei Huang, Richa Bhayani "*Twitter Sentiment Classification using Distant Supervision*". In the Technical Report of Stanford Digital Library Technologies Project, 2010

7. Publically available tweets dataset – http://www.sananalytics.com/lab/twitter-sentiment/sanders-twitter-0.2.zip

8. C. D. Manning and H. Schutze "*Foundations of statistical natural language processing*" MIT Press, 2011.

9. M. Thelwall, K. Buckley, D. Cai, A. Kappas, G. Paltoglou *"Sentiment strength detection in short informal text"*. In the Journal of the American Society for Information Science and Technology

10. Python Natural Language Toolkit – http://www.nltk.org/

11. Positive Keyword List : https://github.com/param107/Polarity-Classification/blob/master/pos_mod.txt

12. Negative Keyword List: https://github.com/param107/Polarity-Classification/blob/master/neg_mod.txt

