Twitter Sentiment Analysis Using Machine Learning Techniques

Bac Le[1] and Huy Nguyen[2]

[1] Department of Computer Science, VNUHCM-University of Science, Viet Nam
lhbac@fit.hcmus.edu.vn
[2] Department of Information Science, Sai Gon University, Viet Nam
huynguyen@sgu.edu.vn

Abstract. Twitter is a microblogging site in which users can post updates (tweets) to friends (followers). It has become an immense dataset of the so-called sentiments. In this paper, we introduce an approach to selection of a new feature set based on Information Gain, Bigram, Object-oriented extraction methods in sentiment analysis on social networking side. In addition, we also proposes a sentiment analysis model based on Naive Bayes and Support Vector Machine. Its purpose is to analyze sentiment more effectively. This model proved to be highly effective and accurate on the analysis of feelings.

Keywords: Twitter, sentiment analysis, sentiment classification.

1 Introduction

Twitter is a popular microblogging service in which users post status messages, called "tweets", with no more than 140 characters. The million of statuses appear on socical networking everyday. In most cases, its users enter their messages with much fewer characters than the limit established. Twitter represents one of the largest and most dynamic datasets of user generated content approximately 200 million users post 400 million tweets per day. Tweets can express opinions on different topics, which can help to direct marketing campaigns so as to share consumers' opinions concerning brands and products, outbreaks of bullying, events that generate insecurity, polarity prediction in political and sports discussions, and acceptance or rejection of politicians, all in an electronic word-of-mouth way. In such application domains, one deals with large text corpora and most often "formal language". At least two specific issues should be addressed in any type of computer-based tweet analysis: firstly, the frequency of misspellings and slang in tweets is much higher than that in other domains. Secondly, Twitter users post messages on a variety of topics, unlike blogs, news, and other sites, which are tailored to specific topics. Big challenges can be faced in tweet sentiment analysis: *a*) neutral tweets are way more common than positive and negative ones. This is different from other sentiment analysis domains (e.g. product reviews), which tend to be predominantly positive or negative; *b*) there are linguistic representational challenges, like those that arise from feature engineering issues; and *c*) tweets are very short and often show limited sentiment cues.

In this paper, we used a data set about 200000 tweets for training classifiers. Table 1 is an example about the types of Twitter. We built a model which classified tweets collected from Twitter APIs into the positive class or the negative class. The model runs on three steps: a classifier categorizes tweets into objective tweets or subjective tweets, another classifier organizes subjective tweets into positive or negative and finally, the system summarizes tweets into a vitual graph. For training, we applied on three kinds of features: Unigram, Bigram, Object-oriented. The training set contains tweets without emoticons. We evaluated that emoticons were noisy when classifiers analyzed tweets. Our experiments proved to be highly accurate. Related work on tweet sentiment analysis is rather limited, but the initial results are promising

Our main contributions can be summarized as follows: *a*) We extracted a new feature set and appraised effectively based on Information Gain, Bigram, Object-oriented extraction methods. *b*) We built a sentiment analysis model based on supervised learning sush as Naive Bayes and Support Vector Machine for enhancing effective classification.

2 Related Works

There were many studies in sentiment analysis but almost those focused on a part of texts or critiques. A tweet is only limited to 140 characters, so it is as different as a critique. Bing Liu [3] (2010), Tang and colleagues [11] (2009) expressed an overview in sentiment analysis in which analyzed the strong points and the weak points of sentiment analysis and they gave many research ways of sentiment analysis. Pang and Lee [1] [2] (2004, 2008) compared many classifiers on movie reviews and gave a vision of insight and comprehension in sentiment analysis and opinion mining. Authors also used star rating as a feature for classification. Go et al [8] (2009) studied on Bigram and POS. They removed emoticons out from their training data for classification and compared with Naive Bayes, MaxEnt, Support Vector Machine (SVM). They evaluated that SVM outperforms others. Barbosa and Feng [9] (2010) pointed that N-gram is slow, so they researched on Microblogging features. Agarwal et al [7] (2011) approached Microblogging, POS and Lexicon features, also they built tree kernel to classify tweets and applied on POS and N-Gram. Akshi Kumar and Teeja Mary Sebastian [5] (2012) approached a dictionary method for analyzing the sentiment polarity of tweets. On the other hand, Stanford University (2013)[1] performs a twitter sentiment classifier based on Maximum Entropy and they built a Recursive Deep Model with a Sentiment Tree Bank but they applied on Movie Reviews, not Twitter.

3 Our Approach

Our approach used classifiers to categorize sentiment into positive or negative and we applied a good feature extractor for enhancing accuracy. The classifiers which are built into a model to effectively classify are Naive Bayes (NB) and Support Vector Machine (SVM).

[1] `http://nlp.stanford.edu/sentiment/index.html`

3.1 Pre-processing of Data

The language model of Twitter has its properties. We pre-processed data for enhancing accuracy and removing noisy features. we pre-processed data on the following steps: *a*) we converted tweets to lower case. In every tweet, we splitted sentences. We removed stopwords out from tweets. In addition, we removed spaces and words which do not start with the alphabet letters. Characters which repeat in a word sush as "huuuuuuuuuuungry", we converted into "huungry". *b*) Usernames. They are attached in tweets to transfer a status to other usernames. An username which attached in a tweet starts with "@". We replaced usernames by "AT_USER". *c*) URLs. Users attach urls in tweets. Urls look like "http://tinyurl.com/cvvg9a". We replaced those into URLs. *d*) #hashtag. Hashtags might provide useful information for us, so we only removed "#" at hashtags. For example, from "#nice" to "nice". *e*) Emoticons. We removed emoticons from tweets because we believed that those are noisy when we analyzed sentiment. For example, we have a following tweet: "@soundwav2010 Reading my kindle2... Love it... Lee childs is good read. #kindle". We converted the tweet into: "AT_USER reading my kindle2 love it lee childs is good read kindle".

3.2 Feature Extraction

We identified the object words of tweets which have sentiment polarity and we extracted those into a feature set. Its name is the object-oriented feature. On the other hand, we detected that a number of words are meaningless or insignificant. So, we have to remove them. We used the Information Gain method to extract these words. Another consideration is that positive words mean negative if they stay behind a negative preposition and in a contrary way. This is an important problem that we used Bigram to resolve. We chose following features for increasing the accuracy of classifiers:

a) We believe that there are an expression of moral of an entity or a keyword which could be extracted from tweets. Objects which belong to a group sush as "company", "person", "city" that they describe an object as "Steve Jobs", "Vodafone", "London" are significant. They have correlation to positive or negative. This is the sentiment polarity of objects. We used this object-oriented feature for increasing the accuracy of classifiers. We used an open data set of Alchemy API to extract these objects. The data set was evaluated by Rizzo and Troncy [10] (2011) on five data sets: AlchemyAPI[2], DBPedia Spotlight[3],Extractiv[4], OpenCalais[5] and Zemanta[6]. Their research showed that AlchemyAPI outperforms others. In additon, Hassan Saif, Yulan He and Harith Alani

[2] `http://www.alchemyapi.com/`
[3] `http://dbpedia.org/spotlight/`
[4] `http://wiki.extractiv.com/w/page/29179775/Entity-Extraction`
[5] `http://www.opencalais.com/`
[6] `http://www.zemanta.com/`

[12] also used AlchemyAPI to extract objects from Twitter and they evaluated that AlchemyAPI got the accuracy of 73.97% of extracting objects. we believe that the object-oriented feature is significant for enhancing the accuracy of sentiment analysis, especially tweets do not contain verbs, adjectives and we used the sentiment polarity of object-oriented features for classifying tweets. We chose the AlchemyAPI data set for extracting this feature.

Table 1. The accuracy of object extraction tools on 500 tweet from Stanford Corpus

Data sets	extracted object-oriented features	Accuracy (%)
AlchemyAPI	108	73.9
Zemanta	70	71
OpenCalais	65	68

b) However, verbs and adjectives are also important that we must extract. Information Gain (IG) is a good method for extracting features (term-goodness) in Machine Learning. IG measures the number of information of bits obtained to predict classifications by the presence or the absence of features in documents. Another problem that the number of words are meaningless or insignificant but classifiers have to assign them into any class. So, we have to remove them from feature sets. Information Gain resolves this problem. We chose Unigram words which are highly significant for classifying. The formula of Information Gain is as follows:

$$\begin{aligned} IG(t) = -\sum_{i=1}^{m} Pr(c_i)logPr(c_i) + Pr(t)\sum_{i=1}^{m} Pr(c_i|t)logPr(c_i|t) \\ +Pr(\bar{t})\sum_{i=1}^{m} Pr(c_i|\bar{t})logPr(c_i|\bar{t}) \end{aligned} \quad (1)$$

In which: *i*) Pr(c_i): the probability of class c_i happens. *ii*) P(t): the probability of feature t happens. *iii*) P($\bar{t}$): the probability of feature t do not happen. *iv*) Pr($c_i|t$) the probability of feature t appears in class c_i. *v*) Pr($c_i|\bar{t}$): probability of feature t do not appear in class c_i.

c) However, when we used Unigram, classifiers are only learned single words. For example, if classifiers process a sentence: "I do not like Iphone", this sentence is categorized into the negative class because word "not" might more weight than word "like", but a sentence: "It is not bad", classifiers categorize this sentence into the negative class because word "not" and word "bad" are negative. We have to train classifiers multiple words. We measured the correlation of two words by statistics χ^2 method and we also chose Bigram multiple words which have high signification.

We extracted 10000 unigram, 100 bigram and 200 object-oriented features to built the list of feature sets. Because when we extracted more features, the accuracy did not change. The list of feature sets on the following steps: *a*) For Naive

Bayes classifier, we used Information Gian and statistics χ^2 to extract Unigam words and Bigram multiple words from Ravikiran Janardhana data set [13] and used AlchemyAPI data set to extract object words from Ravikiran Janardhana data set [13] which have sentiment polarity. *b*) For Support Vector Machine classifier, the process is same, we also used Information Gian and statistics χ^2 to extract Unigam words and Bigram multiple words from Sentiment Stanford data set [13] and used AlchemyAPI data set to extract object words from Sentiment Stanford data set [13] which have positive or negative signification . *c*) For the incorporation of object-oriented feature into Language model, we incorporated the object-oriented feature with the following augmentation method:

$$|V'| = |V| + |S| \tag{2}$$

In which: *i*) V': The new vocabulary set. *ii*) V: The original vocabulary set. *iii*) S: The additional vocabulary set.

d) After extracting features for classifiers, we standardized the feature sets following: *i*) For Unigram, Bigram, Object-oriented features extracted, We saved into two databases. The first database is words, multiple words which are labeled positive. The second database is words, multible words which are labeled negative. *ii*) We mixed the above-mentioned two databases into a database with (key, value) in which key is the vocabulary set of a sentence and value is positive or negative label of this vocabulary set. *iii*) Finally, we converted the vocabulary set into a dictionary set with binary occurences for training classifiers.

3.3 Classification Model

We used Twitter APIs as a library tool to collect tweets from internet for sentiment analysis and we built a system based on Naive Bayes (NB) and Support Vector Machine (SVM). We trained classifiers and we classified tweets collected from internet on the following steps: *a*) We built a Naive Bayes classifier to categorize subjective tweets and objective tweets. The subjective training set is sentences labeled subjective or objective and we applied Unigram, Bigram, Object-oriented features for training. *b*) For the SVM classifer, it classifies the subjective tweets into the positive class or the negative class. The sentiment training set is sentences labeled positive or negative and we applied Unigram, Bigram, object-oriented features for training. *c*) The system also draws a graph after the analysis of feelings. Picture 1 is sentiment analysis steps of the model based on Unigram, Bigram and Object-oriented features.

Naive Bayes (NB). Naive Bayes is a classification method based on statistics using Machine Learning. The idea of this approach is conditional probability among words, phrases and classes to predict statistics of them belonging to a class. A special point of this method is the appearance of words which is independent. Naive Bayes do not evaluate the dependance of words with any class. We used Multinomial Naive Bayes. Naive Bayes assign tweet d with class c. the formula is as follows:

$$C* = argmac_c P_{NB}(c|d) \tag{3}$$

$$P_{NB}(c|d) = \frac{(P(c)\sum_{i=1}^{m} P(f|c)^{n_i(d)})}{P(d)} \tag{4}$$

In which: *i*) f: describing a feature. *ii*) n_i(d): the number of features f_i are found in tweets. There are m features.

We used a Scikit-Learn tool[7] with a Multinomial Naive Bayes model to classify subjective tweets and objective tweets.

Support Vector Machine (SVM). Another technique which is popular in Machine Learning is Support Vector Machine. We also used a SVM Scikit-Learn tool with Linear kernel. A training set which is for Support Vector Machine are two vector sets with m size. m is all of features. An element of the vector describes the presence or the absence of that feature . For instance, A feature is a word found in tweet by the Information Gain method. If the feature is present, its value is 1. In contrary, its value is 0. Bigram and Object-oriented features are same. Fig.1 describes the system including Naive Bayes classifier and Support Vector Machine classifier for classifying tweets from Twitter.

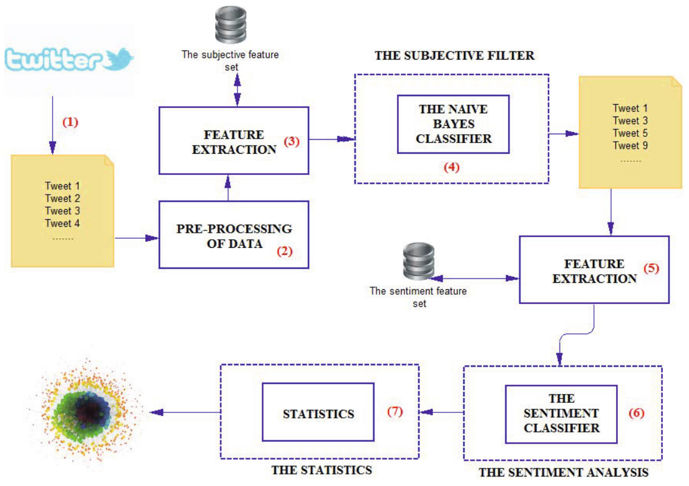

Fig. 1. The overview struture of the system

[7] http://scikit-learn.org/stable/

The operations of classifiers perform through the following steps:

a) Tweets are collected based on a topic from Twitter through Twitter APIs and saved into a database. For example, the topic is "Iphone", the system will downloads tweets from Twitter relating "Iphone". *b*) Pre-processing tweets collected. *c*) After the pre-processing step, the system extracts subjective features based on Information Gain, Bigram and use AlchemyAPI data set to extract the object-oriented feature. *d*) Thanks to the subjective feature set, the Naive Bayes classifier categorizes tweets into the subjective class or the objective class. Objective tweets are then removed. *e*) For the subjective tweets, the system extracts sentiment features based on Information Gain, Bigram and use AlchemyAPI data set to extract the object-oriented feature. *f*) The system puts tweets extracted features into the Support Vector Machine classifier and the classifier classify tweets into the negative class or the positive class. *g*) The system draws a graph based on tweets which is classified.

4 Experiments and Evalutions

4.1 Experiments

We trained Support Vector Machine classifier based on a Sentiment Stanford training set containing 200000 tweets in which 100000 tweets labeled positive and 100000 tweets labeled negative. Thanks to the training set 20000 tweet collected through Twitter APIs by Ravikiran Janardhana[13]. We labeled subjective or objective for training the Naive Bayes classifier. We extracted three types of features containing Unigram, Bigram, Object-oriented features for training and experiments. We used the testing data of Stanford University containing 500 tweets of neutral, negative and positive labeled manual. We built three types of the testing data set: *a*) A data set is extracted features through the Information Gain method with only unigram feature. *b*) A data set is extracted features through Bigram extraction with only Bigram feature. *c*) A data set is extracted three features. They are object-oriented extraction, Information Gain and Bigram. Table 4 describes the accuracy of the system on above-mentioned data sets.

In addition, we also evaluated the accuracy of classifiers on 200000 sentiment tweets of Standford University and 20000 subjective tweets of Ravikiran Janardhana[13] based on 10-fold method. The experiment shows that the Support Vector Machine classifier outferforms the Naive bayes classifier. However, we chose the Naive Bayes classifier to classify subjective tweets because the Naive Bayes classifier measures statistics on independent sentences, so the classifer easily removes the objective sentences of a tweet before the system puts them into the Support Vector Machine classifier for analyzing positive or negative.

4.2 Evalutions

We evaluated the accuracy of classifiers on 200000 sentiment tweets of Standford University and 20000 subjective tweets of Ravikiran Janardhana[13] to compare between classifiers. We used 10-fold method to test classifiers. However, we chose the training data of Ravikiran Janardhana[13] for classifying subjective. Because the training data of Ravikiran Janardhana[13] have subjective and objective tweets which are significant. Table 2 describes the accuracy of the classifiers on 20000 tweets of Ravikiran Janardhana[13].

Table 2. The accuracy of classifiers using Unigram, Bigram and Object-oriented features on 20000 tweets

Classifiers	Features	Accuracy (%)
Naive Bayes	Object-oriented, Unigram, Bigram features	80
Support Vector Machine	Object-oriented, Unigram, Bigram features	80

Table 3 describes the accuracy of the classifiers on 200000 tweets of Standford University.

Table 3. The accuracy of classifiers using Unigram, Bigram and Object-oriented features on 200000 tweets

Classifiers	Features	Accuracy (%)
Naive Bayes	Object-oriented, Unigram, Bigram features	79.54
Support Vector Machine	Object-oriented, Unigram, Bigram features	79.58

As can be seen, Support Vector Machine classifier outferforms the Naive bayes classifier on 200000 tweets of Standford University. Because Naive Bayes classifier measures statistics on independent sentences, but Support Vector Machine evaluates negative or positive on whole tweet which may have many sentences in. However, the classifiers just got 80 percent of accuracy. Because the grammar of tweet gathering from Twitter is not good. Many words are wrong and not necessary. Those have effects on feature selection. Table 4 describes the experiment result on three above-mentioned data sets using the system. The system take average of Naive Bayes and Support Vector Machine. The data set which is used Object-oriented, Unigram, Bigram features are better than others.

Table 4. The accuracy of the system on three data sets

Data sets	Features	Accuracy (%)
1	Object-oriented, Unigram, Bigram features	79.5
2	Unigram	77
3	Bigram	77

Table 5 and table 6 express the detail of Precision, Recall and Accuracy of classifiers on 200000 tweets of Standford University and 20000 tweets of Ravikiran Janardhana[13] using 10-fold method:

Table 5. Precision, Recall and F-Score of the Naive Bayes subjective classifier on 20000 tweets of Ravikiran Janardhana[13] using 10-fold method

Average			
Precision	Recall	F-score	Accuracy (%)
0.80	0.79	0.79	80

Table 6. Precision, Recall and F-Score of the SVM sentiment classifier on 200000 tweets of Standford University using 10-fold method

Average			
Precision	Recall	F-score	Accuracy (%)
0.79	0.79	0.79	79.5

The paper also compared the experiment result with other papers which also used data set of Standford University. Fig.2 describes the graph of accuracy of the papers.

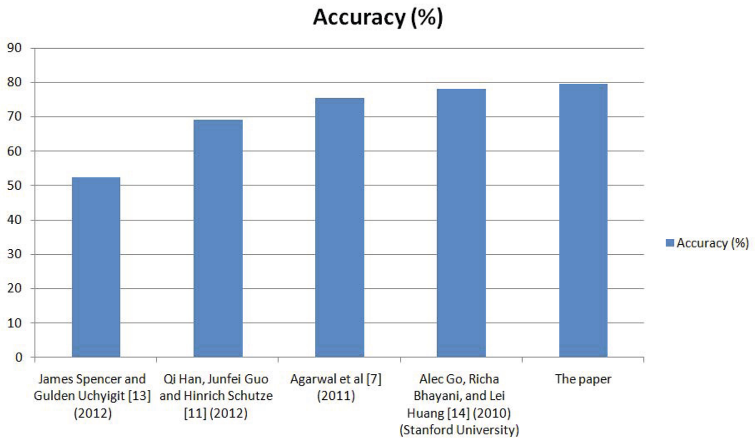

Fig. 2. Comparision with other papers

In our view, the paper outferforms others because we extracted fully significant features for sentiment analysis. We also resolved many comparison tweets. This helps to increase the accuracy of the classifiers. In addition, Fig. 3 describes a graph which the system create for expressing analysis tweets collecting from Twitter based on keyword in the most recent seven days.

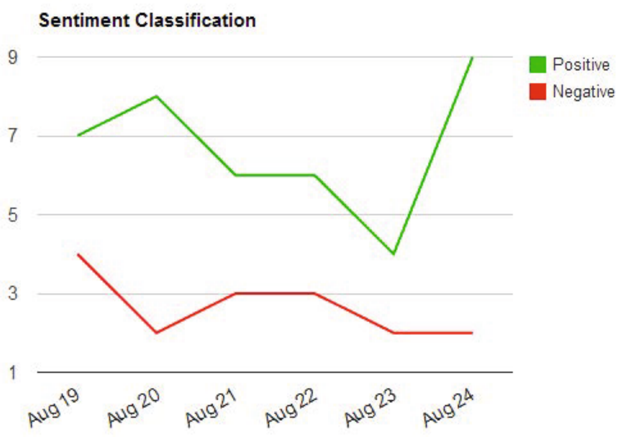

Fig. 3. A graph of analyzing tweets collected from Twitter on keyword "Iphone" in the most recent seven days

5 Conclusions

In conclusion, we built a model which analyzes sentiment on Twitter using Machine Leaning Techiques. Another consideration is that we applied Bigram, Unigram , Object-oriented features as an effective feature set for sentiment analysis. We used a good memory for resolving features better. However, we chose an effective feature set to enhance the effectiveness and the accuracy of the classifiers.

References

1. Pang, B., Lee, L.: A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In: Proceedings of the Association for Computational Linguistics (ACL), pp. 271–278 (2004)
2. Pang, B., Lee, L.: Opinion mining and sentiment analysis. Foundations and Trends in Information Retrieval 2(1-2), 1–135 (2008)
3. Liu, B.: Sentiment Analysis and Subjectivity. In: Handbook of Natural Language Processing, 2nd edn. (2010)
4. Mullen, T., Collier, N.: Sentiment Analysis using Support Vector Machines with Diverse Information Sources. In: Proc. of Conference on Empirical Methods in Natural Language Processing (EMNLP 2004) (2004)
5. Kumar, A., Sebastian, T.M.: Sentiment Analysis on Twitter. IJCSI International Journal of Computer Science Issues 9(4(3)) (2012)
6. Saif, H., He, Y., Alani, H.: Alleviating: Data Sparsity for Twitter Sentiment Analysis. In: Proceedings of the 2nd Workshop on Making Sense of Microposts (#MSM2012): Big Things Come in Small Packages: in Conjunction with WWW 2012 (2012)
7. Agarwal, A., Xie, B., Vovsha, I., Rambow, O., Passonneau: Sentiment analysis of twitter data. In: Proc. ACL 2011 Workshop on Languages in Social Media, pp. 30–38 (2011)

8. Go, A., Bhayani, R., Huang: Twitter sentiment classification using distant supervision. CS224N Project Report, Stanford (2009)
9. Barbosa, L., Feng, J.: Robust Sentiment detection on twitter from biased and noisy data. In: Proceedings of COLING, pp. 36–44 (2010)
10. Rizzo, G., Troncy, R.: Nerd: Evaluating named entity recognition tools in the web of data. In: Workshop on Web Scale Knowledge Extraction (WEKEX 2011), vol. 21 (2011)
11. Tang, H., Tan, S., Cheng, X.: A survey on sentiment detection of reviews. Expert Systems with Applications 36(7), 10760–10773 (2009)
12. Saif, H., He, Y., Alani, H.: Semantic Sentiment Analysis of Twitter. In: Cudré-Mauroux, P., et al. (eds.) ISWC 2012, Part I. LNCS, vol. 7649, pp. 508–524. Springer, Heidelberg (2012)
13. Janardhana, R.: Twitter Sentiment Analysis and Opinion Mining (2010)