The Role of Pre-processing in Twitter Sentiment Analysis

Yanwei Bao[1], Changqin Quan[1], Lijuan Wang[1], and Fuji Ren[1,2]

[1] AnHui Province Key Laboratory of Affective Computing and Advanced Intelligent Machine, School of Computer and Information, HeFei University of Technology, Hefei, 230009, China
baoyanwei007@sina.com, quanchqin@gmail.com, wlj1034900987@163.com
[2] Faculty of Engineering, University of Tokushima, 2-1 Minami-Josanjima, Tokushima 770-8506, Japan
ren@is.tokushima-u.ac.jp

Abstract. Recently, increasing attention has been attracted to Social Networking Sentiment Analysis. Twitter as one of the most fashional social networking platforms has been researched as a hot topic in this domain. Normally, sentiment analysis is regarded as a classification problem. Training a classifier with tweets data, there is a large amount of noise due to tweets' shortness, marks, irregular words etc. In this work we explore the impact pre-processing methods make on twitter sentiment classification. We evaluate the effects of URLs, negation, repeated letters, stemming and lemmatization. Experimental results on the Stanford Twitter Sentiment Dataset show that sentiment classification accuracy rises when URLs features reservation, negation transformation and repeated letters normalization are employed while descends when stemming and lemmatization are applied. Moreover, we get a better result by augmenting the original feature space with bigram and emotions features. Comprehensive application of these measures makes us achieve classification accuracy of 85.5%.

Keywords: Twitter, Sentiment Analysis, Pre-processing.

1 Introduction

Twitter is one of the most fashional micro-blog service platforms where registered users can update their messages and follow others' statuses expediently. Twitter's contents (named tweets) include users' behaviors, states of mind, comments on certain topics etc, and a lot of these contents express the users' sentiments unavoidably. Twitter sentiment analysis is a significative topic because the research findings can feedback much important information we have never thought of (e.g. the trend of the stock market [3, 4]).

Under routine circumstances, sentiment analysis is researched as a problem of classification. This technique is also known as opinion mining aiming to identify the kind of emotions, mainly as positive, neutral or negative at present. There have been a large amount of researches in the area of sentiment classification. Applications utilizing sentiment classification technique might be classified into two main categories: trends prediction [3, 4, 6, 7, 8, 10] and products recommendation [11, 12, 13, 18, 19, 20, 21].

D.-S. Huang et al. (Eds.): ICIC 2014, LNAI 8589, pp. 615–624, 2014.

The works mentioned above are all based on sentiment analysis. However, these works didn't focus on pre-processing methods like [5] did, who discussed the role of pre-processing for reviews. Although pre-processing for Twitter data was used for dimensionality reduction in [1, 2, 16], there is no work specialising in researching the role of pre-processing in Twitter sentiment analysis. In this paper, we concentrate on exploring pre-processing methods to elevate the performance of sentiment analysis besides dimensionality reduction. In our work, we show the role of pre-processing in Twitter sentiment classification, including the effects of URLs, negation, repeated letters, stemming and lemmatization. Experimental results on the Stanford Twitter Sentiment Dataset show that sentiment classification accuracy rises when URLs features reservation, negation transformation and repeated letters normalization are employed while descends when stemming and lemmatization are applied. We also augment the original feature space with bigram and emotions features. All positive-impact methods are applied on the Stanford Twitter Sentiment Dataset comprehensively and we achieve the sentiment classification accuracy of 85.5%.

The rest of the paper is organized as follows. Section 2 outlines existing works on Twitter sentiment analysis. Section 3 describes the methodology we use in this paper. Experiments and results are discussed in Section 4. Finally, we conclude our work and outline future work in Section 5.

2 Related Work

A great deal of researches have been done in the field of sentiment analysis. Several works have pursued a study in this dimension with a particular emphasis on products reviews[2, 11, 12, 13].

The above works are focused on products reviews, which are always well-organized sentences and relate to a particular domain. In contrast, tweets, whose length is limitted to 140 characters, are casual in the language style and multifarious in fields and themes. Moreover tweets contain a large amount of noises, such as hashtags, URLs, and emotions. These characters make Twitter sentiment analysis a challenging assignment.

Researches [1, 3, 4, 6, 7, 8, 14, 15, 16] are working on Twitter sentiment classification. Researches [3, 4, 6, 7, 8] are good examples of Twitter sentiment classification based application. In these works, the sentiment orientations of online information like tweets and reviews were analyzed to predict the trends of the stock market and election or outcomes of the box office.

Go et al. [1] used emotions as noisy labels to label tweets as positive and negative. They set a corpus containing 1.6 million tweets (the Stanford Twitter Sentiment Dataset[1]), which can be obtained by public. Based on this dataset, they continued to train classifiers as Pang et la. [2] did. The best result was 83% reported by MaxEnt Classifier using a combination of unigrams and bigrams as features.

[1] http://twittersentiment.appspot.com/

Under normal circumstances, the first step of sentiment analysis is text pre-processing [1, 2, 5, 16], especially for tweets. Haddi et al. [5] explored the role of text pre-processing in online text sentiment analysis. Agarwal et al. [16] applied some novel methods to pre-process tweets. The result of their experiment illustrated that appropriate text pre-processing methods can significantly increase the classifier's performance. There have also been some researches like [14], who excavated the original features to improve the performance of sentiment classification.

3 Methodology

We follow the following procedures for Twitter sentiment classification.

3.1 Denoising

For visual differentiation, we call pre-processing of this step as denoising. We take the following measures to denoise:

Username There are usernames like "@Tonny", that starts with symbol "@". The usernames indicate who is the information pointing to, i.e. the target. We replace all usernames with a space.

Hashtags Hashtags, marked by symbol "#", mean that the tweets are associated with the particular topics. We replace all hashtags with a space as well.

Emotions Users are used to express sentiment with emotions, e.g. ":)" means happy or other positive affections. Emotions with obvious emotional colouring ar e persisted while others are replaced with a space also.

Others We delete all digital symbols, single letters, punctuation except "'" in "won't", "can't" and "n't" and other non-alphabetic symbols.

3.2 Pre-processing

In this step, five steps are proposed to process tweets, named as URLs features reservation, negation transformation, repeated letters normalization, stemming and lemmatization. We verify the role of URLs in Twitter sentiment analysis by observing the change of classification accuracy when URLs features are reserved or not. Considering the significant impact of negation on sentiment classification, we put forward four kinds of transformations to propose negation in tweets and explore which is the most effective way to improve Twitter sentiment classification accuracy.

Words with repeated letters, e.g. "cooooold", are common in tweets, and people tend to use this way to express their sentiments. Thus it is necessary to deal with these words to make them more formal. Here, repeated letter means a letter emerges more than 3 times consecutively in a word.

In this work, we propose a method based on lexicon to cope with these words. We first build a lexicon using words provided by WordNet[2] and other frequent terms.

[2] http://wordnet.princeton.edu/wordnet/

And then, we perform the following conditional statement. If a word contains repeated letters, we replace repeated letters with two occurrences and output the processed word if the lexicon contains the processed word, else we replace repeated letters with one occurrence and output the result.

Both of stemming and lemmatization are used to achieve feature reduction. Stemming is the process for reducing inflected words to their stem, base or root form, for example, "stemmer", "stemming", and "stemmed" all can be reduced into “stem”. Lemmatization is the process for converting inflected words to their root form, for example, “drove” can be converted into “drive”.

3.3 Features Selection

All of the unigram features and part of bigram features are used to structure the feature space. TF, IG, and $\chi 2$ Statistics are three standards we apply to select bigram features.

TF means the frequency of a feature appears in the corpus, and more frequent more important.

IG, i.e. information gain, is used to evaluate the importance of features on the classification system. The standard of IG is the amount of information feature contributes, more information, more important. We use the formula (1) to calculate the information gain of features:

$$\begin{aligned} &\mathrm{IG(T)=H(C)-H(C/T)=} \\ &-\sum_{i=1}^{n} P(C_i)\log_2^{P(C_i)} + P(t)\sum_{i=1}^{n} P(C_i/t)\log_2^{P(C_i/t)} + P(\bar{t})\sum_{i=1}^{n} P(C_i/\bar{t})\log_2^{P(C_i/\bar{t})} \end{aligned} \tag{1}$$

Where IG(T) is the information gain of feature t, H(C) is the entropy of class C, H(C/T) is the conditional entropy of C in the condition of feature T. $P(C_i)$ is the probability of class C_i occurrence in the corpus, P(t) is the probability of t, $P(C_i/t)$ is the probability of C_i in the condition of t occurrence, and $P(C_i/\bar{t})$ is the probability of C_i in the condition of t absence. What needs to be pointed out is that we use Laplace-like smoothing method when we calculate the probability mentioned above to avoid zero probability and zero denominator.

χ^2 Statistics is the measurement of interdependency between feature and class. Interdependency and χ^2 Statistics are proportional relations. The value of χ^2 Statistics between feature t and class C_i is calculated by the formula (2):

$$\chi^2(\mathrm{t}, C_i) = \frac{(AD-BC)}{(A+B)(C+D)} \tag{2}$$

Where N is the total number of tweets in the corpus, A represents the number of tweets belong to class C_i and contains feature t, B is the number of tweets don’t belong to class C_i but contain feature t, C is the number of tweets belong to C_i but don’t contain feature t, and D means the number of tweets when both are negative.

3.4 Classification

In this work, we employ Liblinear[3] to train a linear classifier. Liblinear is able to handle large-scaled dataset, especially for text classification, whose features space is extremely sparse. And it runs really faster than Libsvm because it doesn't have to compute the kernel for any two points. For optimization, trust region Newton method is applied, which is the combination of trust region method and truncated Newton method. We can see the mathematics foundations of Liblinear in [17].

4 Experiments and Results

In this section, we present the results obtained when several kinds of pre-processing methods are applied individually and jointly. After that, we compare the results with the existing approaches.

4.1 Corpus

The experiments are carried on the Stanford Twitter Sentiment Dataset. The dataset was collected by using emotions as noisy labels to label tweets as positive and negative. For the training set, there are 800,000 positive tweets and 800,000 negative tweets. A set of 177 negative tweets and 182 positive tweets were manually marked, for a total of 359 test tweets.

4.2 Experiments

In the experimental stage, we take the safe-launch mode to carry out the experiment. Alleged safe-launch mode means that we implement the experiment step by step, and we choose the better result in each step. Then the next step will be conducted based on the previous better result. If the effect of a certain pre-processing method is not so good, we will abandon this method.

We use WEKA[4] to perform Liblinear classification. We set the parameter S to be 0 meaning L2-regularized logistic regression, which is appropriate for binary classification. Other parameters are set to be the default values.

We start the experiments with URLs features reservation and negation transformation. In order to verify the effectiveness of processing, we establish four datasets using unigram features named as: **UN0** for the dataset after denoising. **UN1** when dataset contains URLs features and negation is processed, **N** when dataset contains URLs features while negation is not processed, **U** when all URLs features are deleted and negation is not processed.We use the method in **UN0** as our baseline. Here, the method used to process negation is transforming “won’t”,”can’t” and “n’t” into “will not”, “can not” and “ not”, respectively.

[3] http://www.csie.ntu.edu.tw/~cjlin/liblinear/
[4] http://weka.wikispaces.com/

Table 1. The role of URLs and negation

	U	**UN1**	**N**	**UN0**
Accuracy	82.73%	83.01%	81.62%	81.62%

Table 1 indicates the effect of URLs and negation. The result does be better when URLs and negation are processed. The accuracy grows by 0.28% when URLs features are utilized. That is because URLs may contain some helpful features. The effect of negation is more powerful for the accuracy drops down to 81.62% when negation is not processed. Note that, the result of **UN1** is similar to Go [1] while we use 308306 unigram features only.

In consideration of the effect of negation, we propose another three methods to deal with negation:

UN2 transforming all verb negatived by "not" or its abbreviations into "verb not", e.g. "didnt" to "did not".

UN3 transforming "won't","can't" and "n't" into "willnot", "cannot" and "not", respectively.

UN4 transforming "won't","can't" and "n't" into "wont", "cant" and "nt", respectively.

Table 2 shows the result of four datasets when unigram features are used only. **UN3** and **UN4** get the best result, followed by **UN1**, and **UN2** gets the worst result. This result may be due to the fact that **UN3** and **UN4** transform one negation word into another negation word while **UN1** and **UN2** transform one negation word into two words. We conjecture that **UN3** and **UN4** may enhance the unigram features while **UN1** and **UN2** weaken the unigram features, or enhance the bigram features in another way. We name this conjecture as **conjecture1**.

Table 2. Compare of four negation datasets

	UN1	**UN2**	**UN3**	**UN4**
# of features	308306	308284	308452	308332
Accuracy	83.01%	82.73%	83.57%	83.57%

In order to verify **conjecture1**, we augment the original unigram feature space with bigram features. Here we select part of bigram features to be used. So we carry out the subsequent experiment based on **UN1** to ascertain the standard of selecting bigram features and the best number of bigram features. We select features with TF, IG, and χ^2 Statistics. For each method, we perform four experiments to contrast the impact established by the number of bigram features.

Table 3. Accuracy of different feature sets based on **UN1**

Accuracy	# of bigram features			
	200	300	500	1000
Unigram	83.01%			
Unigram+Bigram(TF)	84.40%	83.57%	83.01%	82.73%
Unigram+Bigram(IG)	84.40%	**84.68%**	84.12%	83.29%
Unigram+Bigram(χ^2)	**84.68%**	**84.68%**	84.12%	82.73%

Table 3 presents the results of different feature sets based on **UN1**. We get better results after bigram features are affiliated with feature space. Also IG and χ^2 are more effective than TF when used to select features. We experiment with different number of bigram features and get the best accuracy of 84.68% at 300.

Table 4. Results of using unigram and bigram features

Accuracy	**UN1**	**UN2**	**UN3**	**UN4**
Unigram	83.01%	82.73%	83.57%	83.57%
Unigram+Bigram(IG)	**84.68%**	84.40%	84.40%	84.40%
Unigram+Bigram(χ^2)	**84.68%**	**84.68%**	84.40%	84.40%

To go on verifying **conjecture1**, we carry on the next experiments on **UN2**, **UN3** and **UN4** where 300 bigram features, selected by IG and χ^2 respectively, are used to augment unigram features set. Results are shown inTable 4.

As shown from Table 4, the accuracy increases more on **UN1** and **UN2** than **UN3** and **UN4**. That means bigram features are more effective when be used in **UN1** and **UN2**, and **conjecture1** is verified commendably.

Table 5 shows the results after repeated letters normalization. We get a better result while the number of features is even less (282087). This shows that normalization of repeated letters is effective. We achieve 84.96% sentiment classification accuracy.

Table 5. Results of repeated letters normalization

Accuracy	**UN1**	**UN2**
# of unigram features	281787	281837
Unigram	83.29%	83.01%
Unigram+Bigram(IG)	**84.96%**	84.68%
Unigram+Bigram(χ^2)	**84.96%**	84.40%

To continue reducing features from the original feature space, we introduce stemming and lemmatization to process the corpus. Table 6 shows the effect of stemming and lemmatization. There is a sharp decline in classification accuracy though the feature space is reduced. The results illustrate that stemming and lemmatization are not helpful for Twitter sentiment classification.

Table 6. Effect of Stemming and Lemmatization

Accuracy	**UN1**	Stemming	Lemmatization
# of unigram features	281787	234446	257364
Unigram	83.29%	82.45%	81.62%
Unigram+Bigram(IG)	**84.96%**	81.89%	81.62%
Unigram+Bigram(χ^2)	**84.96%**	81.62%	80.78%

Continually, we augment the Unigram-Bigram feature space with emotions. Here we define emotions such as ":)", ":-)", ":)" etc. as positive emotions, and define emotions ":(", ":-(", ": (" etc. as negative emotions. Once an instance contains one of the positive emotions, the "Positive" feature would be "1" or else it would be "0". "Negative" feature meets the parallel situation. Another 641 instances (321 positive instances and 320 negative instances) selected randomly from training set are removed from the training set and appended to test set.

Table 7 reveals the effect of emotions features. The result based on **UN1** doesn't get better when emotions features are put in. Analysising the wrongly classified instances, we can see that all these instances do not contain emotions in their content. However, the rise of accuracy based on **newdata** illustrates the effectiveness of emotions features and we achieve 85.5% sentiment classification accuracy, which outperforms the baseline by 3.9%. Moreover, our result is better than [1], who reported the best accuracy of 83%, while their number of features is more than 364464.

Table 7. Results of using emotions features

Accuracy	**UN1**	**newdata**
# of unigram features	281789	281780
Unigram+Bigram(IG)	84.96%	85.1%
Unigram+Bigram(χ^2)	84.96%	85.2%
Unigram+Bigram(χ^2)+Emotions	84.96%	**85.5%**
UN0 baseline	81.62%	
Go et al.,2009	83%	

5 Conclusions and Future Work

Twitter has been one of the most popular social networking platforms and it is a really meaningful topic to research Twitter sentiment analysis. However, length limitation, multi-topics, casual language, and rich in symbols, all of these characteristics of tweets make Twitter sentiment analysis a challenging assignment. Pre-processing can improve the classification accuracy besides reducing the original feature space.

In this paper, we conduct a series of experiments to verify the effectiveness of several pre-processing methods. Our experiments results on Stanford Twitter Sentiment Dataset show that URLs features reservation, negation transformation and repeated letters normalization have a positive impact on classification accuracy while stemming and lemmatization have a negative impact.

We also augmented the original feature space with bigram features and emotions simultaneously to improve Twitter sentiment classification appearance. Compared to the existing researches, we get a better result by using fewer features.

There are some possible directions we could try in the future. First, for preprocessing, spelling correction can be used to make tweets more regular. Second, different sets of features should be exploited to enhance the classification performance, and hashtags might be useful features in topic-based sentiment analysis. Finally, semi-supervised classification algorithm should be the focus of our research.

Acknowledgments. This research has been partially supported by National Natural Science Foundation of China under Grant No. 61203312 and the National High-Tech Research & Development Program of China 863 Program under Grant No. 2012AA011103 and the Scientific Research Foundation for the Returned Overseas Chinese Scholars, State Education Ministry, and Key Science and Technology Program of Anhui Province, under Grant No. 1206c0805039.

References

1. Go, A., Bhayani, R., Huang, L.: Twitter sentiment classification using distant supervision. CS224N Project Report, Stanford, pp. 1–12 (2009)
2. Pang, B., Lee, L., Vaithyanathan, S.: Thumbs up?: sentiment classification using machine learning techniques. In: Proceedings of the ACL 2002 Conference on Empirical Methods in Natural Language Processing, vol. 10, pp. 79–86. Association for Computational Linguistics (2002)
3. Zhang, X., Fuehres, H., Gloor, P.: Predicting Stock Market Indicators Through Twitter "I hope it is not as bad as I fear". Procedia - Social and Behavioral Sciences 26, 55–62 (2011)
4. Bollen, J., Mao, H., Zeng, X.: Twitter mood predicts the stock market. Journal of Computational Science 2(1), 1–8 (2011)
5. Haddi, E., Liu, X., Shi, Y.: The Role of Text Pre-processing in Sentiment Analysis. Procedia Computer Science 17, 26–32 (2013)
6. Asur, S., Huberman, B.A.: Predicting the future with social media. In: 2010 IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology (WI-IAT), vol. 1, pp. 492–499. IEEE (2010)
7. Stieglitz, S., Dang-Xuan, L.: Political Communication and Influence through Microblogging-An Empirical Analysis of Sentiment in Twitter Messages and Retweet Behavior. In: 2012 45th Hawaii International Conference on System Science (HICSS), pp. 3500–3509. IEEE (2012)
8. Tumasjan, A., Sprenger, T.O., Sandner, P.G., et al.: Predicting Elections with Twitter: What 140 Characters Reveal about Political Sentiment. ICWSM 10, 178–185 (2010)
9. Williams, C., Gulati, G.: What is a social network worth? Facebook and vote share in the 2008 presidential primaries. American Political Science Association (2008)
10. Mishne, G., Glance, N.S.: Predicting Movie Sales from Blogger Sentiment. In: AAAI Spring Symposium: Computational Approaches to Analyzing Weblogs, pp. 155–158 (2006)
11. Aciar, S., Zhang, D., Simoff, S., et al.: Informed recommender: Basing recommendations on consumer product reviews. IEEE Intelligent Systems 22(3), 39–47 (2007)
12. Aguwa, C.C., Monplaisir, L., Turgut, O.: Voice of the customer: Customer satisfaction ratio based analysis. Expert Systems with Applications 39(11), 10112–10119 (2012)
13. Kang, H., Yoo, S.J., Han, D.: Senti-lexicon and improved Naïve Bayes algorithms for sentiment analysis of restaurant reviews. Expert Systems with Applications 39(5), 6000–6010 (2012)
14. Saif, H., He, Y., Alani, H.: Alleviating data sparsity for twitter sentiment analysis. In: The 2nd Workshop on Making Sense of Microposts (2012)
15. Speriosu, M., Sudan, N., Upadhyay, S., et al.: Twitter polarity classification with label propagation over lexical links and the follower graph. In: Proceedings of the First workshop on Unsupervised Learning in NLP, pp. 53–63. Association for Computational Linguistics (2011)

16. Agarwal, A., Xie, B., Vovsha, I., et al.: Sentiment analysis of twitter data. In: Proceedings of the Workshop on Languages in Social Media, pp. 30–38. Association for Computational Linguistics (2011)
17. Lin, C.J., Weng, R.C., Keerthi, S.S.: Trust region newton method for logistic regression. The Journal of Machine Learning Research 9, 627–650 (2008)
18. Quan, C.Q., Ren, F.J.: Target Based Review Classification for Fine-grained Sentiment Analysis. International Journal of Innovative Computing, Information and Control 10(1) (2014)
19. Quan, C.Q., Ren, F.J.: Unsupervised Product Feature Extraction for Feature-oriented Opinion Determination. Information Sciences (2014), doi: http://dx.doi.org/10.1016/j.ins.2014.02.063
20. Quan, C.Q., Wei, X.Q., Ren, F.J.: Combine Sentiment Lexicon and Dependency Parsing for Sentiment Classification. In: SII 2013 (December 2013)
21. Quan, C.Q., Ren, F.J., He, T.T.: Sentimental Classification Based on Kernel Methods. International Journal of Innovative Computing, Information and Control 6(6) (2010)