A report on

FEATURE BASED SENTIMENT ANALYSIS ON TWITTER

By,
Nibir Nayan Bora
School of Computer Engineering
KIIT University
Bhubaneswar, Orissa

Under the guidance of,
Samit Bhattacharya
Department of Computer Science & Engineering
Indian Institute of Technology Guwahati
Guwahati, Assam

A Summer Intership project at
Department of Computer Science & Engineering
Indian Institute of Technology Guwahati
Guwahati, Assam 781 039

May-July, 2011

Abstract

The objective of Sentiment Analysis is to identify any clue of positive or negative emotions in a piece of text reflective of the authors opinions on a subject. Machine Learning techniques applied to Sentiment Classification needs a labeled training set of considerable size. We introduce the approach of using words with sentiment value as noisy label in an distant supervised learning environment on Twitter corpus. We created a training set of such Tweets and used it to train a Naive Bayes Classifier. We also propose the method of using cross entropy based similarity measure to automatically select a test set which is representative of the training set. Finally, we test the accuracy of our classifier using a combination of minimum word frequency threshold and Catagorical Proportional Difference as the Feature Selection method.

Contents

List of Tables

List of Figures

Chapter 1

Objective

The objectives of this project are:

- To create a corpus of tweets.
- To suggest a method of automatic labeling this corpus.
- To use this corpus to train a sentiment classifier which can effectively classify tweets.
- Use feature selection methods to increase the accuracy of our classifier.
- To suggest a method of creating a test set which is 'representative' of the training set.

Note: The content of this project will be available at `http://mejaj.nibir.me`

Chapter 2

Introdution

2.1 Twitter

Twitter[1] is a microblogging website which has recently become very popular in among the internet community. Users update short messages (called Tweets) within a 140 character limit. They often share their personal opinion on various subjects, discuss current issues and write about events in their life. This platform is preferred because it is free from any political or economic restrictions and easily accessible by millions of people. As the number of users rise, microblogging platforms have become a rich pool of global opinions and sentiments.

2.2 What is Sentiment Classification?

Sentiment Classification, a sub topic of Sentiment Analysis, is the study of computationally determining whether a given piece of text is positive or negative. We usually apply machine learning techniques to sentiment classification, in which a classifier is required to be trained on a labeled training set. This is called supervised learning. However, owing to its nature and the number of tweets that can be collected, it is a challenging task to manually label a training set of such magnitude.

2.3 Why do we need to classify sentiment?

Sentiment classification, when performed on user generated textual content, find many applications. Knowing how users feel about a product or service

[1] http://twitter.com

can help in business decisions for corporates. Political parties and social organizations can collect feedback about there programs and legislation. Artists, musicians and other entertainment icons can reach out for their fans and access the quality of their work. Broadly, it can serve as an automatic polling system, relieving any manual intervention.

2.4 Why choose Twitter?

Twitter serves as a good platform for sentiment analysis because of its large user base from different sociocultural zones. Twitter contains a huge number of tweets, with millions added each day, which can be easily collected through its APIs (Application Program Interface), making it easy to build a large training set.

Chapter 3

Related Work

Previous research in sentiment classification has mostly been concerned with larger piece of text like blog posts [5] and reviews [7, 13]. However, blog posts and reviews differ from tweets because of Twitter's unique language model.

Pang et al. [7], had tested various machine learning techniques (Naive Bayes, maximum entropy and support vector machines) on movie review data. Using feature presence with unigram on a Naive Bayes classifier, they reported an accuracy of 81%. In a later study (Pang and Lee [8]), considering only subjective portions of text, their performance increased to 86.4%.

Our approach of using sentiment suggestive words is similar to the use of emoticons as noisy labels, first introduced by Read [10]. A noisy label is an element within the piece of text which provides information about the class to which the text belongs. This approach was later used by Go et al. [2] on a Twitter corpus. Go et al. reported an accuracy of 81.34% for their Naive Bayes classifier trained on a 1.6 million tweet training set. Duurkoop [1] used hashtags instead of emoticons as noisy label in his training data, and achieved an accuracy of around 70% for Naive Bayes.

Pak and Paroubek [6], in their study, also used emoticons as noisy labels. However, they created their corpus by tweets from the Twitter accounts of 44 popular newspapers. They also discussed the use of an entropy based method for feature selection.

Yessenov and Misailović [13] tested the 'fitness' of a few supervised machine learning methods (Naive Bayes, Decision Trees, Maximum-Entropy) and unsupervised methods (K-Means clustering) on a corpus of movie review comments. In their model they have considered various feature extraction

and feature reduction methods, to yield an accuracy of around 67% for Naive Bayes. Considering only words that appear most frequently in the corpus, the performance was even better. Their results further showed that "distinction of polarity of comments when there are neutral comments as well may be harder problem than simpler binary polarity analysis". Our model resembles theirs in the feature reduction method, however, we have an added benefit of CPD along with a word frequency threshold.

Keefe and Koprinska [3] evaluated a range of feature selection methods using both Naive Bayes classifier and Support Vector Machine. They reported an accuracy of 81.5% for Naive Bayes when used along with Categorical Proportional Difference as a feature selection method on a movie review corpus. According to their study CPD is the best feature selector.

Chapter 4

The Experiment

4.1 Our Approach

We collected tweets using the Twitter Search API[1] to create a training set of 1.5 million tweets, with which we trained a Naive Bayes classifier, treating each tweet as a bag-of-unigrams feature. We propose a method of using words with sentiment value (e.g. cheerful, shocked, disappointed etc.) as noisy labels in the training set. The feature extraction process follows a series of cleaning steps on the tweet before tokenizing it. We also use a combination of a minimum word frequency threshold and Categorical Proportional Difference as our feature selection method, which is intended to increase the accuracy of the classifier. We finally test the accuracy of our classifier using a hand labeled test set.

4.2 Corpus collection

We collected a set of 40 words, each suggesting certain sentiment value and manually categorized them as positive and negative. Most of these words are in the past tense verb form, most likely to be used by an author to express his/her feelings, e.g. *I was ashamed of what I did*. The complete list of positive and negative sentiment words is given in Table 4.1. We label a tweet as positive if it contained any of the positive sentiment words or negative if it contained any of the negative sentiment word. For example, a tweet containing the word 'thrilled' will be labeled as a positive tweet, whereas a tweet containing 'annoyed' will be labeled as a negative tweet. Table 4.2 shows a few tweets labeled in this manner.

[1] http://apiwiki.twitter.com

Table 4.1: List of positive and negative sentiment words

Positive sentiment words	Negative sentiment words
amazed, amused, attracted, cheerful, delighted, elated, excited, festive, funny, hilarious, joyful, lively, loving, overjoyed, passion, pleasant, pleased, pleasure, thrilled, wonderful	annoyed, ashamed, awful, defeated, depressed, disappointed, discouraged, displeased, embarrassed, furious, gloomy, greedy, guilty, hurt, lonely, mad, miserable, shocked, unhappy, upset

The twitter search API allows to retrieve the most recent tweets based on a search query. However, it returns only 100 tweets per page and up to 15 such pages. The API also limits the number of requests per hour from an IP (Internet Protocol) address. We used the positive and negative sentiment words as the query term to collect tweets. Considering the hourly limits and allowing enough time for our requests to yield unique tweets, our system sent out a search request for each word every 30 minutes.

Our system accumulated tweets during the period from June 24, 2011 to July 5, 2011. We removed all retweets as they would cause an unwanted redundancy in our training set. Retweets are tweets that are posted by a user by copying another user's tweet. They are marked by the presence of the characters 'RT' in the tweet. We also removed the matching sentiment word found in each tweets. The tweets were then labeled according to the category in which the matching sentiment word falls. This way, we created a training set of 1.5 million (1,464,638, precisely) tweets consisting of 668,975 positive tweets and 795,661 negative tweets, which was used to train our classifier.

4.3 The Naive Bayes classifier

Naive Bayes classifier is a probabilistic model which estimates the probability that a tweet belongs to a specific class (positive or negative class, in our experiment). We used Naive Bayes because of its low computational overhead and ease of use, yet performing well in many NLP tasks, as indicated by Lake [4]. We run our classifier against a test set of hand labeled tweets to check its accuracy. In general, the probability that a tweet T, belongs to class C,

Table 4.2: Example of a few labeled tweets

Tweet	Label	Matched sentiment word
`Arrived in Basel. Brilliant sunny weather. I'll go to the botanical gardens first. It's a wonderful place to think and write.`	positive	wonderful
`I'm really unhappy with myself at this point and i'm seriously at a breaking point and i'm on the verge of relapse i'm trying so hard`	negative	unhappy
`Toy Story 3 is hilarious. I love the scene where Ken is modelling for Barbie! Fab stuff :-)`	positive	hilarious
`i should really eat something, but i'm just not a fan when it's this hot and miserable. :/`	negative	miserable

can be calculated using Bayes Theorem, as follows

$$P(C|T) = \frac{P(C).P(T|C)}{P(T)} \tag{4.1}$$

Since each class is equally probable, we can reduce equation 4.1 to

$$P(C|T) = P(T|C) \tag{4.2}$$

According to the naive independence assumption each token $w_i \in T$ is independent of each other, therefore,

$$P(T|C) = \prod_i P(w_i|C) \tag{4.3}$$

where $P(w_i|C)$ is the probability of the token w_i occurring in class C. We finally assign the class C to the tweet T, whichever yields the maximum $P(C \mid T)$.

4.4 Unigram feature extractor

Because of Twitter's 140 character limit, tweets are more likely to contain spelling and grammatical errors. Users, sometimes, strips off letters from words (e.g. whr) to fit their message within this limit. Twitter users also use a lot of Internet slang and abbreviations (e.g. OMG, ASAP, etc.) or emoticons.

We apply the following steps to obtain unigrams from the Tweets.

- Remove URLs (e.g. http://bit.ly/aDkhG) and user mentions. User mention is a way to tag a user in a tweet. It is represented by a '@' symbol followed by the username (e.g. @nibirbora).
- Replace emoticons with a token representing the emotion expressed by it. Emoticons, also known as *smileys* are glyphs constructed using the characters available on a standard keyboard, representing a facial expression of emotion. Table 4.3 is a list of few emoticons and their meanings.

Table 4.3: Emoticons and their meanings

Emoticons	Meaning
>:] :-) :) :o) :] :3 :c) :> =] 8) =) :} :^)	smile
>:D :-D :D 8-D 8D x-D xD X-D XD =-D =D =-3 =3	laugh
:'(;*(:_(	cry
>:[:-(:(:-c :c :-< :< :-[:[:{	frown
>;] ;-) ;) *-) *) ;-] ;] ;D	wink
>:o >:O :-O :O	surprise
D:< >:(>:-C >:C >:O D-:< >:-(:-@ :@ ;('_' D< :L	angry

- Tokenize words by splitting the tweet at spaces and punctuation marks.
- Remove stop words (e.g. the, is, at, which, on).
- Replace words with repeated letters. Users sometimes arbitrarily repeat certain letters in a word to put more emphasis on it (e.g. happppppyyyyyy). We designed an algorithm which replaces a word with any letter occurring more than twice with two words, one in which the repeated letter once and twice in the second. For example, the word

happppppyyyyyy will be replaced with four words - *hapy*, *hapyy*, *happy*, *happyy*.

- Stem the words. We use the Porter Stemming algorithm [9] to stem words to their roots. However, we even analyze our results without Porter Stemming.

4.5 Feature selection

Feature selection, used in order to improve the performance of the classifier, is a method to select a portion of the feature set, generated by the trainer, which is most likely to serve in classification. We use Categorical Proportional Difference along with a minimum word frequency threshold as our feature selection method. Categorical Proportional Difference has been successfully studied by Simeon and Hilderman [11] for text catagorization, and by Keefe and Koprinska [3] for sentiment analysis on a movie review corpus. Yessenov and Misailović [13] considered features with frequencies above a minimum threshold in their study of sentiment analysis on movie review comments. However, none of these methods have been tried on a corpus of tweets. The two step feature selection process is discussed next.

4.5.1 Minimum word frequency threshold

In the first step, we remove all features with frequency below a minimum threshold frequency from our feature set. The threshold is set as a percentage calculated on the maximum frequency of any feature in the feature set. We check the accuracy of our classifier at various minimum threshold percentages.

4.5.2 Categorical Proportional Difference

Categorical Proportional Difference (CPD), a measure of how equal two numbers are, can use this to find the features that occur mostly in either one of positive or negative class of tweets. We use the positive and negative frequencies of a feature, to calculate it's CPD, by an equation suggested by Keefe and Koprinska [3] as follows:

$$CPD = \frac{|Positive_f - Negative_f|}{Positive_f + Negative_f} \tag{4.4}$$

If a feature is prevalent in either positive tweets or in negative tweets then it's CPD will be close to one, whereas if it occurs almost evenly in both

positive and negative tweets then its CPD will be close to zero. A high CPD indicates that the feature is worth using considering in classification. For example if the word "wife" appears in exactly as many positive tweets as negative tweets then finding the word "wife" would not contribute in classifying new tweet and its CPD score will be zero. Conversely, if the word "birthday" appears in only positive tweets then finding the word "birthday" in a new tweet would give us a good clue that the tweet is positive, and it would have a CPD score of one.

We use CPD as for feature selection by removing any features whose CPD score is less than some threshold value. We check the accuracy of our classifier at various threshold value.

Table 4.4 and 4.5 shows the effect of the feature selection process on our feature set. The values indicate the feature size after applying feature selection. We later present the accuracies for each of these values.

Table 4.4: Effect of feature selection on the feature set (Using Porter Stemming). Initial feature size = 319719

CPD thresholds	Minimum word frequency threshold			
	1.000%	0.100%	0.010%	0.001%
0.000	1357	6616	27487	319719
0.125	883	4553	21042	296861
0.250	533	2902	15417	287159
0.375	258	1643	10409	270971
0.500	128	903	6919	26615

4.6 Test set

To create our test set, we used the concept of Cross Entropy based similarity measure. Assuming our training set and test set to be two random variables, say X_1 and X_2, generating character, we can calculate their cross entropy as follows:

$$H(X_1, X_2) = -\sum_i X_1(x_i) log_2 X_2(x_i) \tag{4.5}$$

Table 4.5: Effect of feature selection on the feature set (Without Porter Stemming). Initial feature size = 371915

CPD thresholds	Minimum word frequency threshold			
	1.000%	0.100%	0.010%	0.001%
0.000	1336	7975	35772	371915
0.125	896	5662	27582	343808
0.250	537	3639	20253	331427
0.375	267	2166	13810	311278
0.500	613	1206	9188	304950

This cross entropy value is then compared to the entropy value of the training set, calculated as follows:

$$H(X_1) = -\sum_i X_1(x_i) log_2 X_1(x_i) \tag{4.6}$$

A test set having cross entropy value closest to the entropy of the training set is supposed to be a good representation of the training set. This way, we can select a test set from any number of randomly created test sets.

We randomly selected 500 tweets from our training set, from which we scraped out tweets which do not indicate a positive or negative sentiment. We created five such sets, and calculated cross entropy for each. The set which had the cross entropy value closest to the entropy value of the training set was selected as our test set. This test set was then manually labeled. Our final test set consisted of 198 positive sentiment tweets and 204 negative sentiment tweets.

Chapter 5

Results

Our classifier reached a maximum accuracy of 83.33% on our hand labeled test set, when tested using Porter Stemming. This value was achieved when we set the minimum word frequency threshold to 0.001% and the CPD threshold to 0.25, reducing feature set size to 287,159 features, which is about 90% of the original size (319,719 features). The same accuracy was achieved when tested without Porter Stemming, setting the minimum word frequency threshold at 0.001% and the CPD threshold at 0.125. This reduced the feature set size to 343,808 features, which is about 92% of the original size (371,915 features).

The accuracy of the classifier at various minimum word frequency thresholds and CPD thresholds are presented in Figure 5.1 and 5.2, and Table 5.1 and Table 5.2.

When tested without any feature selection method, our classifier showed an accuracy of 77.61% (using Porter Stemming) and 82.83% (without Porter Stemming). We notice that, while our feature selection method considerably increased the accuracy when used along with Porter Stemming, it does not show much difference when used without Porter Stemming.

We also observe that considering features with frequency above a minimum threshold frequency did not contribute much to the accuracy of the classifier. So, we tested our classifier with only CPD as our feature selection method. The results are shown in Figure 5.3 and Table 5.3.

Figure 5.1: Accuracy of the classifier at different minimum word frequency thresholds and CPD thresholds. (Using Porter Stemming)

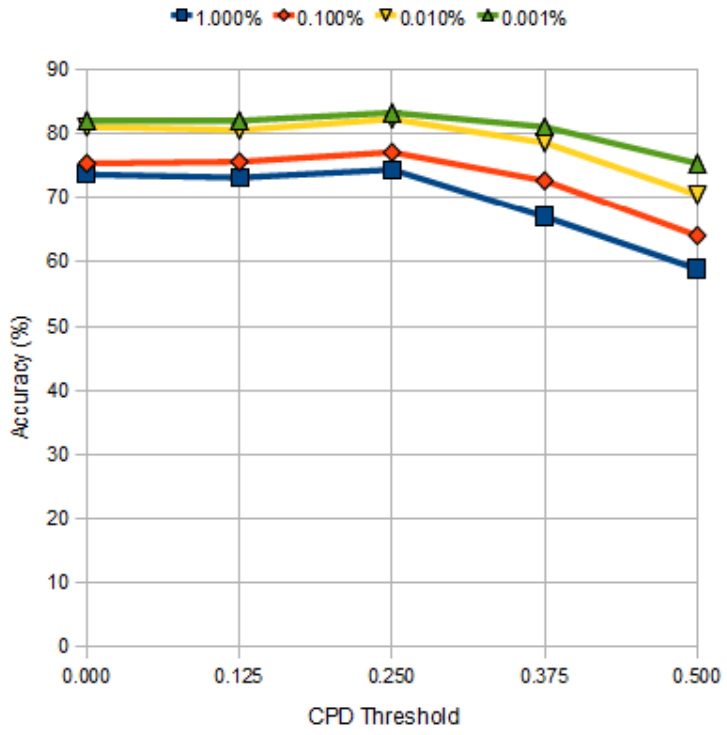

Table 5.1: Accuracy values (in %) at different minimum word frequency thresholds and CPD thresholds. (Using Porter Stemming)

CPD thresholds	Minimum word frequency threshold			
	1.000%	0.100%	0.010%	0.001%
0.000	73.63	75.37	81.09	82.08
0.125	73.13	75.62	80.59	82.08
0.250	74.37	77.11	82.33	**83.33**
0.375	66.91	72.63	78.60	81.09
0.500	58.70	63.93	70.39	75.37

Figure 5.2: Accuracy of the classifier at different minimum word frequency thresholds and CPD thresholds. (Without Porter Stemming)

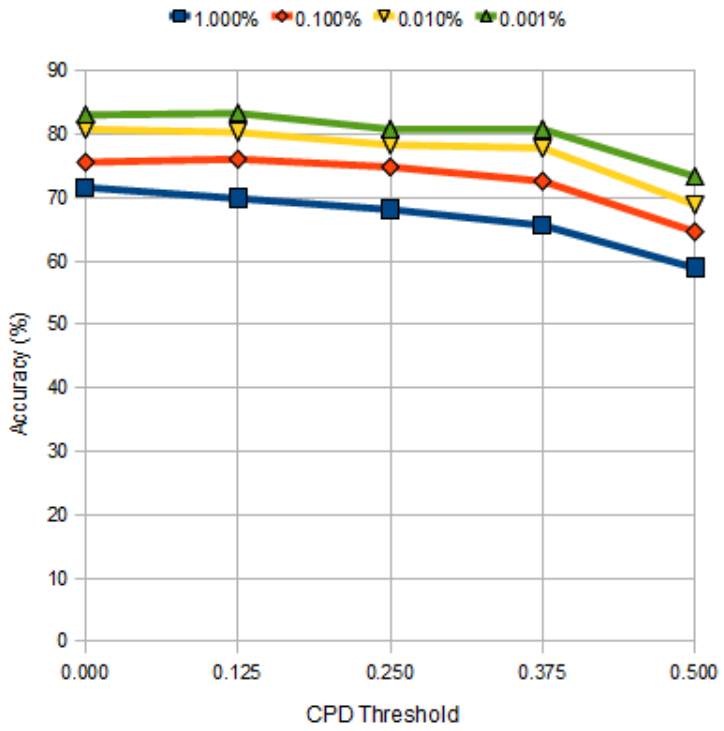

Table 5.2: Accuracy values (in %) at different minimum word frequency thresholds and CPD thresholds. (Without Porter Stemming)

CPD thresholds	Minimum word frequency threshold			
	1.000%	0.100%	0.010%	0.001%
0.000	71.64	75.62	80.84	83.08
0.125	69.90	76.11	80.34	**83.33**
0.250	68.15	74.87	78.35	80.84
0.375	65.67	72.63	77.86	80.84
0.500	58.95	64.67	68.90	73.38

Figure 5.3: Accuracy of the classifier at different CPD thresholds. (Without minimum word frequency threshold)

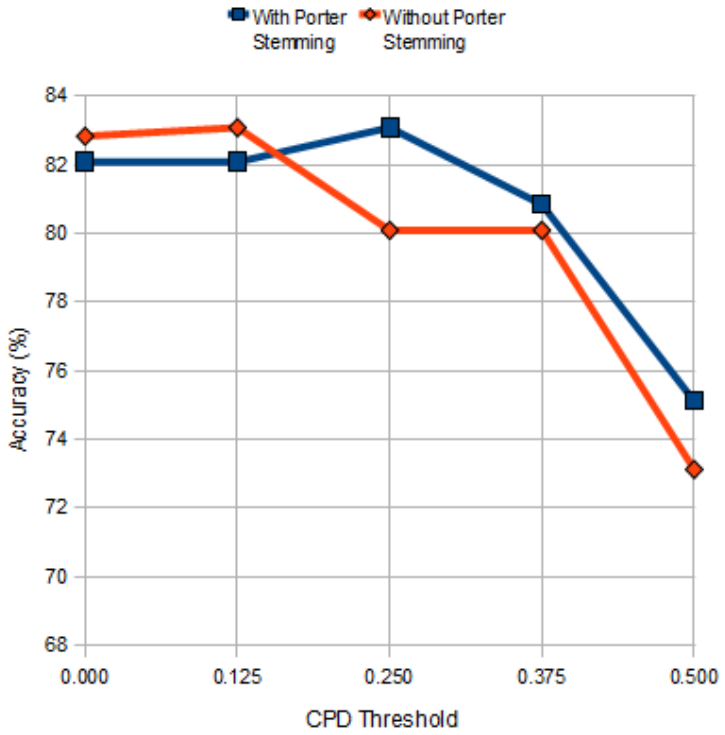

Table 5.3: Accuracy values (in %) at different CPD thresholds. (Without minimum word frequency threshold)

CPD thresholds	Using Porter Stemming	Without Porter Stemming
0.000	82.08	82.83
0.125	82.08	**83.08**
0.250	**83.08**	80.09
0.375	80.84	80.09
0.500	75.12	73.13

Chapter 6

Conclusion

We built a sentiment classification tool which can accurately find the polarity (positive or negative) of a tweet. We showed that sentiment suggestive words can be effectively used as noisy label for a Twitter corpus. We create a training set of 1.5 million tweets in this manner to train a Naive Bayes classifier. We discussed the use a combination of minimum word frequency threshold and Categorical Proportional Difference (CPD) as feature selection method. We found that while CPD successfully increased the efficiency of the classifier, setting a minimum word frequency threshold did not. We proposed a method to automatically create a test set from the training set using cross entropy based similarity measure. We finally tested our classifier to achieve a maximum accuracy of 83.33%.

Chapter 7

Future Work

Following is a list of ways in which the study can be extended in the future:

- Build a classification tool that can classify various emotions and not just classify positive and negative sentiment.
- Use such a tool to create an user interface which is reflective of the user's mood.
- Find a way to eliminate tweets from the training set that does not indicate any sentiment value.
- Perform grammatical semantic analysis on the tweets to possibly increase the accuracy of classification.

References

[1] Jorik Duurkoop. Real-Time Happiness. Bachelor thesis, University of Amsterdam, Faculty of Science, Faculty of Science, Science Park 904, 1098 XH Amsterdam, June, 2010.

[2] Alec Go, Richa Bhayani, and Lei Huang. Twitter sentiment classification using distant supervision. CS224N Project Report, Stanford, 2009.

[3] Tim O Keefe and Irena Koprinska. Feature Selection and Weighting Methods in Sentiment Analysis. *The 14th Australasian Document Computing Symposium*, 2009.

[4] Thomas Lake. Twitter Sentiment Analysis, Western Michigan University, Kalamazoo, MI, For client William Fitzgerald, April, 2011.

[5] Gilad Mishne and Maarten de Rijke. Capturing Global Mood Levels using Blog Posts. In *AAAI 2006 Spring Symposium on Computational Approaches to Analysing Weblogs*, The AAAI Press, Menlo Park, CA/Stanford University, CA, pp. 145152, August, 2006.

[6] Alexander Pak and Patrick Paroubek. Twitter as a corpus for sentiment analysis and opinion mining. In *Proceedings of the Seventh conference on International Language Resources and Evaluation (LREC10), European Language Resources Association (ELRA)*, Valletta, Malta, pp. 19-21, May 2010.

[7] Bo Pang, Lillian Lee, and Shivakumar Vaithyanathan. Thumbs up? Sentiment classification using machine learning techniques. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 79-86, 2002.

[8] Bo Pang and Lillian Lee. A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In *Proceedings of the ACL*, pages 271-278. ACL, 2004.

[9] Martin F. Porter. An algorithm for suffix stripping, *Program*, 14(3) pp. 130-137, 1980.

[10] Jonathan Read. Using emoticons to reduce dependency in machine learning techniques for sentiment classification. In *Proceedings of ACL-05, 43nd Meeting of the Association for Computational Linguistics. Association for Computational Linguistics*, 2005.

[11] Mondelle Simeon and Robert Hilderman. Categorical Proportional Difference: A Feature Selection Method for Text Categorization. *The Australasian Data Mining Conference (Aus DM)*, pp. 201-208, Nov. 2008.

[12] Yiming Yang and Jan O. Pedersen. A comparative study on feature selection in text categorization. In *Proceedings of the 14th International Conference on Machine Learning (ICML97)*, Nashville, U.S.A., pp. 412-420, 1997.

[13] Kuat Yessenov and Saša Misailović. Sentiment Analysis of Movie Review Comments. 6.863 Spring 2009 final project, CSAIL, MIT, 2009.