2013 IEEE 14th International Conference on Mobile Data Management

Opinion Mining on Social Media Data

Po-Wei Liang
Department of Computer Science and Information Engineering, National Taiwan University of Science and Technology, Taipei, Taiwan, ROC.
Email: M10015085@mail.ntust.edu.tw

Bi-Ru Dai
Department of Computer Science and Information Engineering, National Taiwan University of Science and Technology, Taipei, Taiwan, ROC.
Email: brdai@csie.ntust.edu.tw

***Abstract*—Microblogging (Twitter or Facebook) has become a very popular communication tool among Internet users in recent years. Information is generated and managed through either computer or mobile devices by one person and is consumed by many other persons, with most of this user-generated content being textual information. As there are a lot of raw data of people posting real time messages about their opinions on a variety of topics in daily life, it is a worthwhile research endeavor to collect and analyze these data, which may be useful for users or managers to make informed decisions, for example. However this problem is challenging because a micro-blog post is usually very short and colloquial, and traditional opinion mining algorithms do not work well in such type of text. Therefore, in this paper, we propose a new system architecture that can automatically analyze the sentiments of these messages. We combine this system with manually annotated data from Twitter, one of the most popular microblogging platforms, for the task of sentiment analysis. In this system, machines can learn how to automatically extract the set of messages which contain opinions, filter out non-opinion messages and determine their sentiment directions (i.e. positive, negative). Experimental results verify the effectiveness of our system on sentiment analysis in real microblogging applications.**

Keywords*—*Microblogging; Sentiment analysis; Opinion Mining

I. INTRODUCTION

Microblogging websites have evolved to become a source of a diverse variety information, with millions of messages appearing daily on popular web-sites. Users can post real time messages about their life, share opinions on variety of topics and discuss current issues on these microblogging websites. Product reviewing has been rapidly growing in recent years because more and more products are selling on the Web. The large number of reviews allows customers to make an informed decisions on product purchases. However, it is difficult for product manufacturers or businesses to keep track of customer opinions and sentiments on their products and services. In order to enhance the customer shopping experiences a system is needed to help people analyze the sentiment content of product reviews.

The power of social media as a marketing tool has been recognized, and is being actively taken advantage of by people, governments, major corporations, and schools. Twitter is perhaps the most popular microblogging website where users create status messages called "tweets", which are short status updates and musings from Twitter's users that must be written in 140 characters or less. Tweets containing opinions are important because whenever people need to make a decision, they want to hear others' opinions. The same is true for organizations. However, many real-life applications require very detailed analyses in order to gather information from, for example, a product review, whose data could help users or managers make important product-related decisions. This approach is also being actively employed by governments or companies to collect and analyze feedback on their policies or products.

Most of the user-generated messages on microblogging websites are textual information, identifying their sentiments has become an important issue. The research in the field started with sentiment classification, which treated the problem as a text classification problem. Text classification using machine learning is a well studied field [1], and there is ample research of the effects of using various machine learning techniques (Naive Bayes (NB), Maximum Entropy (ME), and Support Vector Machines (SVM) [2]. After building and testing models using Naive Bayes, MaxEnt and Support Vector Machines (SVM), they reported that SVM showed the best performance. However most of the traditional research has focused on classifying long texts, such as reviews [2]. However since Microblogging messages that are short and colloquial, traditional algorithms do not perform as well as they do for long texts. For this reason, there has been in recent years a lot of research in the area of sentiment classification that has targeted microblogging data [11,13]. Go et al. (2009) focused on distant learning to acquire sentiment data, and use tweets containing positive emoticons like ":)", ":-)" to denote positive sentiment, and negative emoticons like ":(", ":-(" for negative emotional content. However, there are many contradictions, for example, "RT @MarieAFrnndz: The Twilight Saga is over :(I'm so sad, I'll miss it", in this example we can find that this message means positive but the emotion is negative. In this paper, to overcome these challenges, we aim to design a system which automatically combines supervised learning that is capable of extracting, learning and classifying tweets, with opinion expressions. The basic idea is to use domain-specific training data to build a generic classification model from social media data to help improve the performance. The experimental results demonstrate the effectiveness of the proposed system is work well.

The remainder of this paper is organized as follows. We discuss several supervised learning algorithms and short text classification related to our work in Section 2. In Section 3, we

978-0-7695-4973-6/13 $26.00 © 2013 IEEE
DOI 10.1109/MDM.2013.73

introduce the proposed system architecture, and experimental evaluations are presented in Section 4. The conclusions are given in Section 5.

II. RELATED WORK

There has been a wide range of research done on sentiment analysis, from rule-based, bag-of-words approaches to machine learning techniques. Two main research directions of opinion mining operate on either the document level [2,3,4] or the sentence level [5,6,7]. Both document level and sentence level classification methods are usually based on the identification of opinion words or phrases. For this, there are basically two types of approaches: (1) lexicon-based approaches, and (2) rule-based approaches. In the former, a lexicon table will be built first, with each word in this table belonging to positive or negative evaluations. The echo count of positive words and negative words will then be calculated, or some formula which considers the distance between each opinion word and product feature will be used to determine the sematic direction. In rule-based approaches, parts-of-speech (POS) taggers will first be used to tag each word, and then co-occurrence patterns of words and tags will be found to determine the sentiments.

However in this paper, we focus on microblogging data like Twitter, on which users post real time reactions to and opinions about "everything". It is important to note that there are differences between product reviews and messages on microblogs. The messages on microblog are short, filled with colloquial and often people do not care about the grammar of their messages. In light of these characteristics, the use of traditional methods of sentiments analysis will yield poor results. A number of recent approaches on sentiment analysis take this into account, such as sentiment classification that classifies opinion texts or sentences as positive, negative or neutral [8-13].

III. SYSTEM ARCHITECTURE

In this paper, we will determine the data's category first, because we assume that different domains are associated with different customary terms and expressions, which will affect the accuracy of sentiment analysis. As such, we propose a new system architecture and combine various supervised learning methods to improve the final accuracy of sentiment classification. Because the labeling of data is very time-consuming, many researchers use data which contains emoticons to identify sentiment, and use these data as training data [12,14]. However, since emoticons are not always consistent with the sentiment, there will be many mistakes in the training data. Therefore, in this paper, we use the manually labeled data as the training data to build our model.

In Figure 1, we introduce our system architecture called Opinion Miner. First, we will crawl tweets from Twitter, and perform some pre-processing steps; then tweets which contain opinions are extracted. After that, tweets containing opinions are classified, because it is often the case that each of the different areas or categories of text data has its special terminology and common representation, and thus we hope that through text classification the overall accuracy rate can be improved. Training data of different categories are then used to build classifiers.

A. Preprocessing

First we will introduce various properties of messages that users post on Twitter. Some of the many unique properties include the following:

1) Usernames: Users often include Twitter usernames in their tweets in order to direct their messages. A de facto standard is to include the @ symbol before the username (e.g. @liang).

2) Hash Tags: Twitter allows users to tag their tweets with the help of a "hash tag", which has the form of "#<tag-name>". Users can use this to convey what their tweet is primarily about by using keywords that best represent the content of the tweet.

3) RT: If a tweet is compelling and interesting enough, users might republish that tweet, commonly known as re-tweeting, and twitter employs "RT" to represent re-tweeting (e.g. "RT @RodyRoderos: I love iphone 5 but i want samsung note 2 :(").

Second we eliminate tweets that:

- Are not in English.
- Have too few words (threshold is set as five).
- Have too few words apart from greeting words.
- Have just a URL.

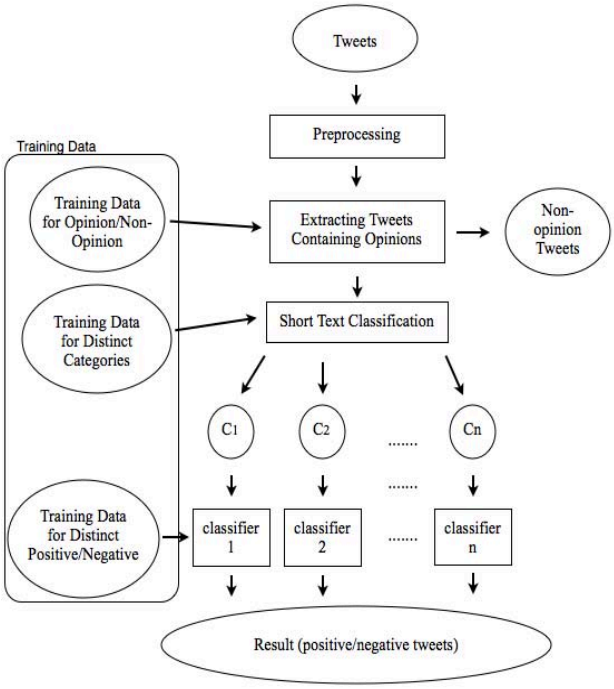

Fig. 1. System Artitecture of Opinion Miner

And then three resources, introduced in Agarwal, A.[15], are deployed for pre-processing twitter data: 1) a stop word dictionary, 2) an emoticon dictionary, and 3) an acronym dictionary. The stop word dictionary identifies stop words, and the emoticon dictionary identifies the 170 emoticons listed on Wikipedia with their emotional state to extract the emotion content in tweets. An online acronym dictionary, with translations for 5,184 acronyms, is used to re-store the acronyms. Thus, we pre-process all the remaining tweets as follows:

- All words are transformed into lower case.
- All the emoticons with a their sentiment polarity are extracted and saved with reference to the emoticon dictionary.
- Targets (e.g. "@liang") are replaced with "USER".
- Each word is enriched by Part-of-Speech (POS) tag (Verb, Noun, Adjective, etc.) in the learning corpus. To do this, we use the tool Tree Tagger [19], which automatically gives a Part-of-Speech tag for each word in a text.
- WordNet (Fellbaum 1998) is employed to determine whether the word is an English word or not.
- Each word is checked to ascertain if it is a stop word or not by referencing the stop word dictionary.
- Replace a sequence of repeated characters by one character, for example, convert cooooooool to col.
- According to previous preprocessing steps, all words are transformed into (word, POS tag, English_word, Stop_word) tuples, where English_word identifies whether this word is an English word or not (EN represents English Word, and NEN represents Non-English Word) and Stop_word identifies whether this word is a stop word or not (ST represents Stop Word, and NST represents Non-Stop Word). For example, (iPhone,NN,NEN,NST).

B. Extracting Tweets Containing Opinions

In real world, the tweets containing opinion is more valuable. So in this part we want to filter out the tweets without opinion. To do this, we use Naive Bayes (NB) classifier [18] on the training that data we labeled manually to classify tweets composed of opinions or non-opinions. Naive Bayes is a simple model which works well on text categorization [16]. We implemented a unigram Naive Bayes model [11,13] and employed the Naive Bayes simplifying independence assumption and class *c* is assigned to tweet *d*:

$$P(c|d) = \square(c)\ \prod_{1\le k\le n} P(t_k|c) \quad (1)$$

In this formula, t_k represents the *kth* token in a tweet, *P(c)* is the prior probability of a document occurring in class *c*, and *n* is the size of a tweet. In text classification, our goal is to find the best class for the document. As the best class in Naive Bayes classification is maximum a posteriori class *c*, we compute the *arg max* as follows:

$$\tilde{C} = arg\ max\ P(c|d) = arg\ max\ P\ (c) \prod_{1\le k\le n} P(t_k|c) \quad (2)$$

It is therefore better to perform the computation by adding logarithms of probabilities instead of multiplying probabilities. Hence, the maximization that is actually done in most implementations of NB is:

$$\tilde{C} = arg\ max\ [logP(c) + \sum_{1\le k\le n} logP(t_k|c)] \quad (3)$$

To eliminate zeroes in each term of count, we use add-one, which simply adds one to each count, and T_{ct} represents the count of each term in class *c*.

$$P(tk|c) = \frac{T_{ct}+1}{\sum_t (T_{ct}+1)} \quad (4)$$

In this step, the system can classify the tweets into opinion class and non-opinion class. Then the system passes the opinion part into the next step.

C. Short Text Classification

The intrinsic idea of this part is that we observed that a word may have different meanings in different domains. For example, "@htc I wouldn't know, no beats headphones came with my beats device :(sad" and "@teresa_fnts YES! they were. So sad that was the last twilight movie :(I loved wreck it Ralph. I took my nephew to see it :) he loved it!". We can see that "sad" means a negative word in the former example and the "sad" in the latter means a positive word. So the unigram Naïve Bayes classifier method is used together with the pre-labeled training data to build the multi-classifier. We use distinct categories of training data. However, since for unigram features, there are usually many different features, and as such it is helpful if we discard some useless features. (Even though our training set just contains thousands of sentences, it is still a large number of features for our training set.) In order to solving this problem, we try two different feature selection algorithms. The first is Mutual Information (MI) [11]. The idea of mutual information is that, for each class *C* and each feature *F*, there is a score to measure how much *F* can contribute to making a correct decision on class *C*. The formula of MI score is equation (5):

$$MI(C;F) = \sum\sum P(C,F) log \frac{P(C,F)}{P(C)P(F)} \quad (5)$$

In practice, we also use add 1 smoothing for each Count of *C* and *F* to avoid a denominator of zero. The second algorithm is X^2 Feature Selection [11]. The idea of X^2 Feature selection is similar to mutual information. In that for each feature and class, there is also a score to measure whether the feature and the class are independent of each other. For this reason, the X^2 test is employed, which is a statistic to determine to what degree two events are independent. It assumes the feature and class are independent and calculates the X^2 value with a larger score implying they are not independent. The formula of X^2 score is equation (6):

$$X^2(F,C) = \frac{N(N11-N00-N10N01)^2}{(N11+N01)(N11+N10)(N10+N00)(N10+N00)} \quad ,(6)$$

where N is the total number of training sentences. $N11$ is the number of co-occurrences of the feature F and the class C. $N10$ is the number of sentences containing the feature F but that are not in class C. $N01$ is the number of sentences in class C that do not contain feature F. $N00$ is the number of sentences not in C and that do not contain feature F.

In this part, the system gets the tweets containing opinion from the previous step, and classifies them according to the content of the tweets. Finally, the result will be sent to the next step to determine the orientation of a tweet.

D. Training Multiple Classifiers in Distinct Categories

We now reach the step of predicting the orientation of an opinion sentence, i.e., positive or negative. As we mentioned above, some words in different areas or categories can have different meanings. In order to improve the final identification accuracy, we need to first classify the short texts according to their domains, so that the classifier can automatically classify with greater performance the tweets as being either positive or negative. Unlike other researches using the identification of emotions is performed to ascertain positive and negative training data [11, 13], in this paper we use the training data that we labeled manually. In [11, 13], if a tweet contains positive emoticons like ":)" and ":-)", it is considered as positive training data. On the contrary, tweets with negative emoticons like ":(" and ":-(" are regarded as negative training data. However, it is likely that this method to determine the training data will result in many mistakes in the training data. For example, "RT @MarieAFrnndz: The Twilight Saga is over :(I'm so sad, I'll miss it" is not a negative tweet, but rather represents the user's feelings about how touching the movie is, which implies a positive category, so that is why we use labeled data to build the model. As shown in Figure 1, we will have different training data of positive and negative tweets for different categories. By using these training data and Naive Bayes method, we can build many binary classifiers of different categories to complete the system. However since labeling data is very time-consuming, the size of this training data is small. In the next section we will show the experiment results of using the two different types of training data.

IV. EXPERIMENT

In this section, we evaluate the whole system and present results for predicting the semantic orientations on Twitter. The main task of the system is to classify tweets to positive versus negative versus non-opinion. Next we will show the results for each part, and then for the whole system.

The datasets used in our experiments are described in part A. Experimental result of each step and discussions are presented in part B, and the final result of Opinion Miner and the comparison with an existing method are shown in part C.

A. Data Sets

There is no large public available data set of Twitter tweets with sentiment, so we use Twitter API to help us collect data. However since the Twitter API has a limit of 100 tweets in a response for any request, we crawl tweets of three distinct categories (camera, mobile phone, and movie) as our training set from the time period between November 1, 2012 to January 31, 2013. The Twitter API has a parameter that specifies which language to retrieve tweets in. We set this parameter to English, and to test our system on tweets in English. We believe that our system also can be extended to work in other languages. Then we manually labeled the tweets crawl from Twitter. Because labeling the data is a time consuming task, the amount of training data set is not very much currently. The details are presented in Table 1. Table 2 represents the details of our testing data which was also manually labeled.

TABLE I. TRAINING DATA

	Positive	**Negative**	**Non-opinion**
Camera	449	297	446
Mobile Phone	472	724	603
Movie	798	168	485
Total	1719	1189	1534

TABLE II. TESTING DATA

	Positive	**Negative**	**Non-opinion**
Camera	62	20	36
Mobile Phone	116	35	59
Movie	140	11	31
Total	318	66	126

B. Results

In this subsection we will show the result of each part of the proposed system called Opinion Miner.

1) Extracting Tweets Containing Opinions: This part of results are shown in Table 3. In this paper we treat the positive and negative tweets as opinion, and others as non-opinion.

TABLE III. RESULT OF EXTRACTING TWEETS CONTAINING OPINION

	Opinion (384)	**Non-opinion (126)**
Opinion	346	70
Non-opinion	38	56

As shown in Table 3, the total accuracy is 76.8%. In order to eliminating the influence of unbalanced training data, we try to use the Mutual Information feature selection algorithm introduced in (5). However, as shown in Figure 2, the accuracy is not improved significantly. Therefore, how to deal

with unbalanced training data is still a challenge that we need to solve in our future research.

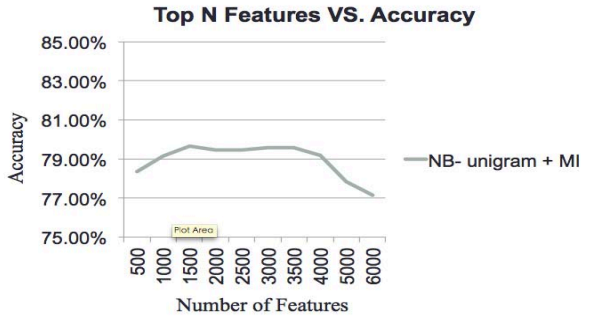

Fig. 2. Effect of Feature Size on Naive Bayes Classifier for Extracting Tweets Containing Opinions

2) Short Text Classificaton: In short text classification, we use Naive Bayes classifier and feature selection to train the muti-classifier. If we only use Naive Bayes, the accuracy is 91%. Because the performance of x^2 Feature selection is very similar to that of mutual information in our experiement, both methods increase accuracy. We choose the mutual information as our feature selection method. After calculating mutual information (MI) score, only top N features with highest scores will be picked for the feature set to test.

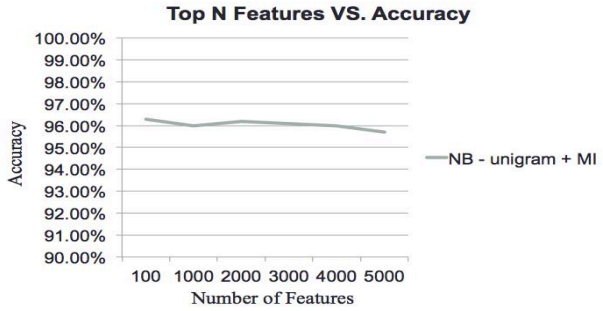

Fig. 3. Effect of Feature Size on Naive Bayes Classifier for Short Text Classification

In Figure 3, the best accuracy is 96.6%, and Table 4 is shown the detail for short text classification.

3) Training Multiple Classifiers in Distinct Categories: In this part, we want to determine the semantic orientation of a tweet (positive/negetive). We use unigram Naive Bayes to bulid the model, and the result is shown in Table 5. In this part, we do not consider the tweets of non-opinion, because it were already wrong when they were not be filtered out through the previous process. We only focus on positive and negative data, so the accuracy is 90.17%. However the result is affected by unbalanced testing data, we observed the weight of positive data is higher than negative data, so the result tends to the positive one.

TABLE IV. RESULT OF SHORT TEXT CLASSIFICATION

	Camera (118)	**Mobile Phone (210)**	**Movie (182)**
Camera	107	6	0
Mobile Phone	11	204	0
Movie	0	0	182

TABLE V. RESULT OF TRAINING CLASSIFIER

	Positive (297)	**Negative (49)**	**Non-opinion (70)**
Positive	282	28	64
Negative	15	21	6

On the other hand, we also to generated another training data using the emotions to denote positive or negative tweets [11,13]. We want to compare the effect of two distinct training data sets. A new training data set is shown in Table 6. Then we use the new training data in Table 6 to train the classifier in this part, and the result is shown in Table 7.

TABLE VI. TRAINING DATA USING EMOTIONS

	Positive	**Negative**
Camera	1484	1201
Mobile Phone	1819	2242
Movie	2330	1938
Total	5663	5381

TABLE VII. RESULT OF TRAINING CLASSIFIER USING THE NEW TRAINING DATA

	Positive (297)	**Negative (49)**	**Non-opinion (70)**
Positve	106	9	27
Negatvie	191	40	43

Similarly we still do not consider the non-opinion part, and we get the accuracy in Table 7 of 58.65%, which is not very good. Besides the unbalanced data problem, we observed that there are many types of sentence “I want xxx :(” in the Twitter. For example, “I want my grandma's htc.... :(”, “Bored of my Olympus. I want a Lumix :(”, and “I want to watch Wreck It Ralph so bad :(”. We think these examples are all positive data, so using the emotions to collect training data is not always correct.

C. Comparison of models

Finally we combine the result of Table 3 and Table 5 and show the final result of the whole system called Opinion Miner in Table 8.

TABLE VIII. RESULT OF OPINION MINER

	Positive (318)	**Negative (66)**	**Non-opinion (126)**
Positive	282	28	64
Negative	15	21	6
Non-opinion	21	17	56

We use a unigram model for comparison. The unigram feature extractor is the simplest way to retrieve features from a tweet, and researchers report well performance for sentiment analysis on Twitter data using a unigram model [11, 13]. The result is shown in Table 9, and we can see the accuracy is 67.58%. We think the reason is still the unbalanced training data. As shown in Table 9 most of testing data are classified to be positive, which result in low accuracy.

TABLE IX. RESULT OF UNIGREAM MODEL

	Positive (318)	**Negative (66)**	**Non-opinion (126)**
Positive	250	19	59
Negative	20	40	13
Non-opinion	48	7	54

Finally, we show the result of two models in Table 10. We can observe that with the same training data, the opinion miner works better than the unigram model.

TABLE X. ACCURACY OF UNIGRAM MODEL AND OPINION MINER

	Accuracy
Unigram Model	67.58%
Opinion Miner	70.39%

V. CONCLUSION AND FUTURE WORK

We designed a system called opinion miner which integrated machine learning techniques and domain-specific data, and experimental results demonstrated the effectiveness of the whole system.

Machine learning performed well in the classification of sentiments in tweets. We believe that their accuracy can still be improved. In this paper, we demonstrated the use of domain-specific training data to build the model, and obtained a very positive performance. In our future work, we plan to further improve and refine our techniques in order to enhance the accuracy of the system. With this in mind, the following is a list of possible research directions. First, emoticon data can be used to check the results of the classification, and if the results conflict with the emotion meaning of the tweet, another method can be employed to once more determine the direction of the semantic content. Otherwise remain the same result. Second distinct machine learning techniques can be strategically deployed in different parts, to analyze which method is more suitable. Finally, rule-based models or methods of natural language processing can be incorporated into our system.

REFERENCES

[1] Fabrizio Sebastiani. Machine learning in automated text categorisation. Technical Report IEI-B4-31-1999, Istituto di Elaborazione dell'Informazione, 2001.

[2] B. Pang, L. Lee, and S. 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.

[3] P. Turney. Thumbs Up or Thumbs Down? Semantic Orientation Applied to Unsupervised Classification of Reviews. ACL'02, 2002.

[4] K. Dave, S. Lawrence, and D. Pennock. Mining the Peanut Gallery: Opinion Extraction and Semantic Classification of Product Reviews. WWW'03, 2003.

[5] M. Gamon, A. Aue, S. Corston-Oliver, and E. K. Ringger. Pulse: Mining customer opinions from free text. IDA'2005.

[6] M. Hu and B. Liu. Mining and summarizing customer reviews. KDD'04, 2004.

[7] S. Kim and E. Hovy. Determining the Sentiment of Opinions. COLING'04, 2004.

[8] A-M. Popescu and O. Etzioni. Extracting Product Features and Opinions from Reviews. EMNLP-05, 2005.

[9] Kunpeng Zhang, Yu Cheng, Yusheng Xie, Daniel Honbo, Ankit Agrawal, Diana Palsetia , Kathy Lee , Wei-keng Liao , Alok Choudhary, SES: Sentiment Elicitation System for Social Media Data, Proceedings of the 2011 IEEE 11th International Conference on Data Mining Workshops, p.129-136, December 11-11, 2011.

[10] Apoorv Agarwal, Boyi Xie, Ilia Vovsha, Owen Rambow, and Rebecca Passonneau. 2011. Sentiment analysis of twitter data. In Proceedings of the Workshop on Languages in Social Media, pages 30–38. Association for Computational Linguistics.

[11] Go, A., Huang, L., Bhayani, R.: Twitter sentiment classification using distant supervision. In: CS224N Project Report, Stanford (2009).

[12] X. Ding, B. Liu, and P. S. Yu, "A holistic lexicon-based approach to opinion mining," Proceedings of the Conference on Web Search and Web Data Mining (WSDM), 2008.

[13] Alexander Pak and Patrick Paroubek. 2010. Twitter as a corpus for sentiment analysis and opinion mining. Proceedings of LREC.

[14] Wei Jin, Hung Hay Ho, Rohini K. Srihari, OpinionMiner: a novel machine learning system for web opinion mining and extraction, Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining, June 28-July 01, 2009, Paris, France.

[15] J. 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.

[16] AGARWAL, A., XIE, B., VOVSHA, I., RAMBOW, O., AND PASSONNEAU, R. Sentiment analysis of twitter data. In Proceedings of the ACL 2011 Workshop on Languages in Social Media (2011).

[17] C. D. Manning and H. Schutze. Foundations of statistical natural language processing. MIT Press, 1999.

[18] Dumais, Susan, et al. "Inductive learning algorithms and representations for text categorization." Proceedings of the seventh international conference on Information and knowledge management. ACM, 1998.

[19] H. Schmid. Treetagger. In TC project at the Institute for Computational Linguistics of the University of Stuttgart, 1994.