Using Skipgrams, Bigrams, and Part of Speech Features for Sentiment Classification of Twitter Messages

Badr Mohammed Badr
Dept. of Comp Sci & Engg
University College of Engineering
Osmania University, Hyderabad
badr.md87@gmail.com

S. Sameen Fatima
Dept. of Comp Sci & Engg
University College of Engineering
Osmania University, Hyderabad
sameenf@osmania.ac.in

Abstract

In this paper, we consider the problem of sentiment classification of English Twitter messages using machine learning techniques. We systematically evaluate the use of different feature types on the performance of two text classification methods: Naive Bayes (NB) and Support Vector Machines (SVM). Our goal is threefold: (1) to investigate whether or not part-of-speech (POS) features are useful for this task, (2) to study the effectiveness of sparse phrasal features (bigrams and skipgrams) to capture sentiment information, and (3) to investigate the impact of combining unigrams with phrasal features on the classification's performance. For this purpose we conducted a series of classification experiments. Our results show that POS features are useful for this task while phrasal features could improve the performance of the classification only when combined with unigrams.

1 Introduction

Due to the rapid growth of online opinion-rich resources with the advent of Web 2.0, a new area of text mining has emerged: sentiment analysis and opinion mining. The research of sentiment analysis is concerned with developing computational techniques that extract, categorize, and summarize subjective information (such as attitudes and sentiments) in textual resources. One fundamental task of sentiment analysis is to analyze a text fragment and detect the author's attitude with respect to a topic (e.g., determining whether a movie review is positive or negative towards the movie).

Twitter is an online, highly social microblogging service that enables subscribers to communicate with short text messages called "tweets". Millions of people around the globe use Twitter on daily basis to publicly express their opinions regarding various aspects of their everyday lives. Thus, Twitter provides a massive data source which public and private organizations can harness to monitor the opinions of the crowd towards almost anything. Mining sentiments and opinions expressed in Twitter can provide indispensable information for various applications such as brand monitoring, customer management, and political forecasting.

Nevertheless, Twitter messages possess unique linguistic characteristics that make them distinguished from conventional user-generated content on the web (e.g., movie reviews). Tweets are short (maximum length is 140-character), cover a wide spectrum of topics, and written in an informal conversational style. The majority of tweets are poorly written and contain many non-standard lexical units such as emoticons (e.g., *_*), emojis (e.g., <3), neologisms (e.g., gr8), hashtags (e.g., #thingsilike), and acronyms (e.g., lol). In addition, most subscribers write their tweets using mobile devices, which increases the frequency of unintentional spelling mistakes. Petz et al. (2013) performed an empirical analysis on text samples collected from various social media, and they found misspellings ratio to be the highest in Twitter.

Because of the aforementioned reasons, sentiment analysis on Twitter text is considered much more challenging problem than other domains. The lexical variations of Twitter language affect the vector representation of the corpus. Saif et al. (2012) argued that data sparsity is the major challenge faced when dealing with Twitter text. They showed that Twitter text has much more low-frequency terms than movie reviews. This can be justified by the topic diversity and the large number of irregular words that occur in tweets. To get a closer look at this problem, we conducted a comparison between the Stanford Twitter Sen-

D S Sharma, R Sangal and E Sherly. Proc. of the 12th Intl. Conference on Natural Language Processing, pages 268–275,
Trivandrum, India. December 2015.

timent (STS) dataset[1] and movie reviews dataset[2] in terms of vocabulary growth. Figure 1 shows the vocabulary growth rate of the two datasets. From Figure 1, it can be observed that the vocabulary of Twitter grows at a higher rate than movie reviews as the corpus size gets larger (measured by the total number of words). We manually inspected the low-frequency words in the Twitter vocbulary. Most of these words were informal abbreviations (e.g., 4eva), elongated words (e.g., beeeeer), and hashtag phrases (e.g., #mayblowyourmind).

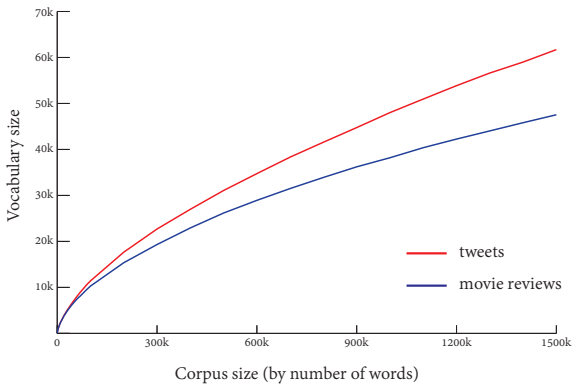

Figure 1: A comparison between the STS corpus and a movie reviews corpus in terms of vocabulary growth. Corpus size (x-axis) is plotted against the vocabulary size (y-axis).

In this paper, we address the problem of sentiment analysis of English tweets using machine learning-based classification techniques. We tackle this problem as a binary classification task where we have two classes: positive vs. negative.

2 Related Work

Using machine learning techniques to tackle the task of Twitter sentiment analysis has been the dominant technique in the literature (Go et al., 2009; Bifet and Frank, 2010; Pak and Paroubek, 2010; Kouloumpis et al., 2011; Asiaee T et al., 2012; Saif et al., 2012; Nakov et al., 2013). The pioneer work in this direction is Go et al. (2009) who used emoticons as noisy signals to obtain annotated training data. They experimented with SVM, Naive Bayes, and Maximum Entropy classifiers, and used unigrams, bigrams, and POS tags as features to build two-class sentiment classifiers. Their best classifier was Maximum Entropy trained on a combination of unigrams and bigrams with an accuracy of 83.0%. Their results give an indication that combining different levels of n-gram features has a positive impact on the classifiers performance.

The impact of using POS features for Twitter sentiment classification was hotly debated in the literature. While Go et al. (2009) and Kouloumpis et al. (2011) concluded that POS features are not useful at all for Twitter sentiment classification, Pak and Paroubek (2010) showed that the distribution of POS tags is not uniform among different sentiment classes, which suggests that they can be used as discriminating features to train sentiment classifiers. In the latter approach, POS tags were used in conjunction with n-gram features to train a Naive Bayes classifier. However, they did not investigate how much improvement was gained when POS features were added to unigrams.

Fernández et al. (2014) experimented with different levels of skipgrams extracted from tweets to train an SVM sentiment classifier. Their results show that skipgram features outperform conventional n-gram features. Nevertheless, no other experiments were conducted to evaluate the effectiveness of skipgrams on other learning algorithms nor the imapct of combining skipgrams with unigrams on the classification performance.

3 Approach

Our approach is to extract different types of features from tweets and build sentiment classifiers using two supervised machine learning algorithms: Naive Bayes (NB) and Support Vector machines (SVM). We investigate the effectiveness of four different sets of features: unigrams, unigrams with part of speech tags, bigrams, and skipgrams. First, each set of features is used as a standalone feature set to train the two classifiers. Then, different feature sets are combined in order to study the impact of combining sprase feature sets on the classifiers' performance.

3.1 Twitter Sentiment Data

For the research conducted in this paper, we used the STS corpus which was created by Go et al. (2009). The STS training data was collected using the Twitter API and consists of a balanced set of 1.6 million tweets (800,000 tweets in each sentiment category). Tweets in the training set were automatically labeled using emoticons as noisy sig-

[1]http://www.sentiment140.com
[2]http://ai.stanford.edu/ amaas/data/sentiment

nals to detect the author's sentiment. For example, a tweet is considered positive if it contains any of the emoticons `:)`, `:-)`, `:D`, or `=)`, and considered negative if it contains any of the emoticons `:(` or `:-(`. To filter out tweets of mixed sentiments, the authors removed tweets that contain both positive and negative emoticons. However, using emoticons as labels could produce noisy instances because the assumption of relating emoticons to the author's mood may not hold in all situations. Therefore, the work in this paper follows the Distant Supervision paradigm (Mintz et al., 2009).

A subset of the original STS training data was used to build the sentiment classifiers in this paper. We have randomly sampled a balanced set of 200,000 training instances from the original corpus. To evaluate the classifiers, the STS test dataset was used. The test data contains 182 positive and 177 negative tweets which were collected by querying the Twitter API with specific keywords including brands, products, and celebrities. Contrary to the training data, tweets in the test data were manually labeled by human annotators.

3.2 Linguistic Preprocessing

Twitter has its own conventions which makes tweets distinct from other online textual data. In addition, the raw text of tweets is noisy and usually contains non-standard lexical tokens (e.g., emoticons, acronyms, hashtags, etc.). Thus, tweets have to be processed using Twitter-specific tools followed by several steps of text normalization.

3.2.1 Tokenization and POS Tagging

In this step, we used CMU TweetNLP[3] to tokenize the tweets and produce the part-of-speech tags. TweetNLP is a Java-based microblogging-specific tool that was developed by Owoputi et al. (2013) to handle online conversational text such as Twitter messages. This POS tagger uses its own tag set which was crafted by Gimpel et al. (2011) for this kind of noisy and informal textual data. Figure 2 shows an example of a tweet that has been tagged with this tool.

3.2.2 Token Normalization

To reduce the number of word types, or the vocabulary size of the Twitter corpus, many preprocessing steps are necessary to normalize Twitter text. We use the same normalization steps as proposed by Saif et al. (2012). These steps include the following:

ikr	smh	he	asked	fir	yo	last
!	G	O	V	P	D	A
name	so	he	can	add	u	on
N	P	O	V	V	O	P
fb	lololol					
^	!					

Figure 2: An example of a tweet that has been tagged by TweetNLP. The original tweet is "*ikr smh he asked fir yo last name so he can add u on fb lololol*". This tweet can be formally writen as "*He asked for your last name so he can add you on Facebook*". Figure taken from (Owoputi et al., 2013).

- Case-folding: all tokens are first converted to lowercase letters.
- Username replacement: all user mentions are replaced by the token *MENTION*.
- Web links replacement: all URLs are replaced by the token *URL*.
- Punctuation replacement: all punctuation marks are replaced by the token *PUNCT*.
- Hashtags: the hash symbol (#) is removed from hashtags and they are treated as regular words afterwards.
- Digits: all digit characters are replaced by the token *DIGIT*.
- Word compression: any sequence of repeated letters is reduced to two letter. For example, "*coool*" and "*cooool*" are compressed as a single token "*cool*".

Table 1 summarizes the effect of the normalization steps on the vocabulary size. It can be observed that the number of word types has been reduced by 58.5% due to token normalization.

3.3 Feature Extraction

For the purpose of this research, four different types of features were extracted from the preprocessed tweets. These features are: Unigrams (individual word tokens separated by a whitespace or a punctuation mark), Unigrams–POS (unigrams with their POS appended to the token), Bigrams (pairs of adjacent word tokens), and Skipgrams

[3]http://www.ark.cs.cmu.edu/TweetNLP

Normalization step	Vocab size	Reduction(%)
No normalization	196,630	0%
Case-folding	169,362	13.7%
Usernames	106,020	37.4%
URLs	97,088	5.5%
Punctuations	96,157	0.5%
Digits	85,961	10.5%
Hashtags	85,022	0.6%
Word compression	81,454	4.1%
Total Reduction		**58.5%**

Table 1: Effects of normalization steps on the vocabulary size.

(pairs of word tokens that are not necessarily adjacent). For example, consider the following preprocessed tweet:

" i love my new smartphone "

The extracted features are shown in Table 2. Each extracted feature in the table is enclosed within square brackets "[ ]".

It is worth mentioning that the skipgram features extracted in this step are actually k-skip-bigrams, where k is the length of the tweet. In other words, we don't use a fixed window length to generate skipgram pairs, but the window length is allowed to expand for the entire text of the tweet. Using this approach to generate skipgram features will increase the size of the feature space by a considerable factor. In order to reduce the number of the skipgram features, we used the following procedure: (1) Tokens tagged with parts of speech G (foreign words or abbreviations), $ (digits and numbers), & (coordinating conjunction), U (URLs and emails), and P (pre- or post- postions) are removed from the set of candidate words for skipgrams generation. (2) We alphabetically arranged the words in each candidate pair. Thus, the features [i–love] and [love–i] are both represented as [i–love] regardless of the order in which each token appeared in the tweet. (3) Words that occur only once in the corpus (a.k.a. *hapaxes*) are excluded when generating features. Using this procedure allowed us to reduce the feature space substantially. Table 3 shows some summary statistics of the extracted features for each feature set.

Feature set	**Extracted features**
Unigrams	[i] [love] [my] [new] [smartphone]
Unigrams–POS	[i–O] [love–V] [my–D] [new–A] [smartphone–N]
Bigrams	[i–love] [love–my] [my–new] [new–smartphone]
Skipgrams	[i–love] [i–my] [i–new] [i–smartphone] [love–my] [love–new] [love–smartphone] [my–new] [my–smartphone] [new–smartphone]

Table 2: The extracted features from the sample tweet: "*i love my new smartphone*".

Feature set	**# Features**	**Hapaxes (%)**
Unigrams	81,454	76.84%
Unigrams–POS	97,084	78.23%
Bigrams	656,440	87.57%
Skipgrams	3,084,656	92.04%

Table 3: Summary statistics for the extracted features.

3.4 Classification Methods

In this paper, we use machine learning techniques to build predictive models that can classify a tweet as positive or negative based on the sentiment expressed in the tweet. In other words, we tackle this problem as a binary text classification problem. We experimented with two learning algorithms that have been widely used for text classification tasks: Naive Bayes (NB) and Support Vector Machines (SVM).

In order to implement the aforementioned classification methods on our Twitter data sample, we adapted the same bag-of-features approach as in Pang et al. (2002) to convert extracted features from each tweet into a feature vector. We consider each tweet as a text document. Then, we extract a set of m features $\{f_1, \ldots, f_m\}$ from the entire Twitter corpus as shown in section 3.3. These features might include, for example, the unigram [*good*], the bigram [*really–good*], or the skipgram [*my–good*]. Finally, each tweet is represented as a feature vector $\mathbf{t} = (n_1(t), \ldots, n_i(t) \ldots, n_m(t))$, where $n_i(t)$ as the number of times a feature f_i occurs in a tweet $\mathbf{t}$.

3.4.1 Multinomial Naive Bayes

Naive Bayes classification is a probabilistic classification method which has been formulated based on Bayes theory. In this work, we used a multinomial Naive Bayes (MNB) model for text classification (Manning et al., 2008). To classify a tweet **t**, a MNB classifier assigns a class $c*$ as

$$c* = argmax_c P_{NB}(c|t)$$

$$P_{NB}(c|t) = \frac{P(c) \cdot P(t|c)}{P(t)}$$

The probability $P(t)$ is fixed for each class and has no role in estimating $c*$. Therefore, by applying the conditional independence of features assumption, $P_{NB}(c|t)$ can be estimated as

$$P_{NB}(c|t) = P(c)(\prod_{i=1}^{m} P(f_i|c)^{n_i(t)})$$

The probability $P(f_i|c)$ represents the relative frequency of the feature f_i in tweets belonging to class c and estimated through maximum likelihood. To deal with unknown features, the Laplace (add-one) smoothing technique is used to avoid zero-valued probabilities. Thus, $P(f_i|c)$ is estimated as

$$P(f_i|c) = \frac{count(f_i,c)+1}{\sum_{f \in V} count(f,c)+m}$$

where m represents the total number of features of a feature set.

3.4.2 Support Vector Machines

Joachims (1998) has shown that Support Vector Machines (SVM) is a highly effective classification technique for sparse text data. The motivation for using SVM for text classification problems originates from the capability of this algorithm to effectively handle sparse and highly dimensional data representations. Given a binary classification problem and a set of training examples, the aim of the training phase is to construct a hyperplane, denoted as a vector $\vec{w}$, that represents the largest possible separator between each class of the training examples. Thus, SVM classifiers are usually refereed to as maximum margin classifiers. When this largest margin separator is learned, deciding the class of unknown text documents is simply predicting which side of the hyperplane they might fall on.

The mathematical details of the SVM algorithm is beyond the scope of this paper. In our research, we used the LIBLINEAR package which includes a linear implementation of the SVM algorithm developed by Fan et al. (2008).

4 Evaluation

4.1 Experimental Set-up

We used a sample of the STS dataset described in section 3.1 as training data. We sampled 100,000 positive and 100,000 negative tweets from the original corpus. The STS testing dataset was used to evaluate the classifiers. Then, the same linguistic preprocessing and feature extraction steps were applied to both training and testing datasets. First, token normalization steps were applied to the output of TweetNLP tool using regular expressions. Second, we used a Python script to extract each of the feature sets described in section 3.3 from the preprocessed tweets. Features that occur only once were removed form the final feature sets. For classification, we used the multinomial Naive Bayes algorithm in-built in *WEKA*[4], the open source data mining software developed by Hall et al. (2009). The LIBLINEAR package was used for SVM with L2-regularized L2-loss and the cost parameter c was tuned to get the best performance of the SVM. We found that setting $c = 0.01$ works best for moderate number of features (Unigrams and Unigrams–POS), and $c = 0.005$ works best for large number of features (Bigrams, Skipgrams, and combinations of feature sets).

4.2 Experimental Results

In this section, we present the classification results, both from using each feature set in isolation and combining unigram features with phrasal features.

4.2.1 Individual Feature Sets

First, each set of features described in section 3.3 is used as a standalone feature set to train the classifiers. Table 4 shows the performance of each classifier as measured by accuracy.

From the results summarized in Table 4, it can be observed that SVM outperforms MNB when unigram features (both Unigrams and Unigrams–POS) are used to train the the classifiers. However, MNB outperforms SVM when phrasal features (both Bigrams and Skipgrams) are used. For Bigrams, and compared to the unigrams baseline, MNB accuracy has improved from 74.7% to 76.0% while SVM accuracy declined from 79.9% to 76.0%. The same can be observed for skipgram features, where MNB accuracy improved to 78.8%

[4] http://www.cs.waikato.ac.nz/ml/weka

Feature Set	MNB	SVM
Unigrams	74.7%	79.9%
Unigrams–POS	76.0%	**80.5%**
Bigrams	77.1%	76.0%
Skipgrams	**78.8%**	74.7%

Table 4: Classifications' accuracy for each set of features individually. Bold-faced accuracy scores denote the top-performing classifier for each learning algorithm.

Feature Set	MNB	SVM
Unigrams + Bigrams	78.3%	82.3%
Unigrams + Skipgrams	79.1%	83.0%
Unigrams–POS + Bigrams	78.6%	81.1%
Unigrams–POS + Skipgrams	**80.2%**	**83.6%**

Table 5: Classifications' accuracy for combined feature sets. Bold-faced accuracy scores denote the top-performing classifier for each learning algorithm.

while SVM declined to 74.7%. These results are comparable to those of Go et al. (2009) who observed similar behavior of their MNB and SVM classifier for unigram and bigram features. Therefore, one can conclude that the performance of SVM degrades as the sparsity degree of the feature representation increases, given that bigrams are sparser than unigrams, and skipgrams are sparser than bigrams. On the other hand, the sparsity degree does not negatively affect the performance of MNB.

Moreover, our results show that POS tags improve classification performance when they are attached to the unigrams. Compared to the unigrams baseline, accuracy of MNB improved from 74.7% to 76.0% while SVM accuracy slightly improved from 79.9% to 80.5%. These results contradict with the findings of Go et al. (2009) and Kouloumpis et al. (2011) who concluded that POS features are not useful for Twitter sentiment analysis. In both works, a general-purpose POS tagger was used to produce the POS tags. Therefore, our results provide evidence that using microblogging-specific POS tagger to produce POS features provide informative features for machine learning-based sentiment classifiers.

4.2.2 Combined Feature Sets

Next, we investigate the impact of combining unigram features with phrasal features on the classification's performance. To combine different feature sets, we merge two sets of features to generate a single feature set. For example, if the Unigrams feature set has n features is combined with the Bigrams feature set which has m features, the resulting feature space from the merge will have $n + m$ number of features. Table 5 shows the classification's accuracy for each combined features set.

From Table 5, one can observe the following: First, adding phrasal features to unigram features consistently enhance the classifiers' performance and yield an improvement over the classifiers trained on individual feature sets. Second, SVM classifiers consistently beat MNB when trained on combined features. Therefore, it can be concluded that using unigram features in combination with phrasal features improves the classification performance of learning algorithms. Moreover, skipgram features seem to be more capable of capturing informative sentiment terms than bigram features. Finally, combining Unigrams–POS and Skipgrams produced the optimal feature set of both learning algorithms as it achieved top performance (83.6% for SVM and 80.2% for MNB).

5 Discussion

Our classification results show that the two learning algorithms (MNB and SVM) behave differently in response to the sparsity degree of the text representation. Interestingly, the MNB classifier trained on skipgrams alone was optimal among the the other MNB classifiers even though the skipgram feature set is much more sparser than the other feature sets. Likewise, the MNB classifier trained on bigrams showed better performance than the unigram baseline. However, the increase in the sparsity degree hampers the performance of SVM as both classifiers trained on bigrams and skipgrams performed worse than the unigram baseline. Regarding POS features, and contrary to previous works in the literature, we found that adding POS features to unigrams improves the performance of the classification for both algorithms. We justify the difference in our result to the choice of the POS tagger, since the previous works used a general-purpose POS tagger that was trained on different domains other than social media text.

In addition, our results show evidence that combining unigram features with phrasal features always yields better performance than using feature sets in isolation. Sparse feature sets, which were less effective for SVM classifiers when used as standalone features, do not negatively impact the performance of SVM when combined with unigram features. Thus, we conclude that combining sparse phrasal feature sets with ungirams can be considered as a smoothing technique to alleviate the the sparsity problem and provide more informative features for sentiment classifiers.

In our experiments, we used a subset of the STS data as our training data (200,000 tweets). However, our optimal classifier in this research, the SVM trained Unigrams–POS and Skipgrams features combined, achieves 83.6% accuracy. This result outperforms all the classifiers of Go et al. (2009) which were trained on the entire original training dataset ($\sim$1.6M tweets) and evaluated on the same testing set. Their top performing classifier was the maximum entropy classifier trained on a combination of unigrams and bigrams (they reported 83.0% accuracy). This finding suggests that using skipgrams as features in combination with unigrams is a very good, yet simple solution to consider when the available training data is limited in size.

6 Future Work

In this paper, we showed that combining unigrams with skipgram features consistently improves the classification performance. Our skipgram features are actually k-skip-bigrams where k is the length of the tweet. For future work, one can investigate the impact of using fixed values for k on the classification performance as well as the sparsity degree.

Furthermore, we aim to expand the work conducted in this paper by considering the neutral class since not all tweets convey a polar sentiment. To tackle this task, the problem of Twitter sentiment classification would be reformulated as 3-class classification problem. The major challenge would be how to collect neutral tweets for training. One way to collect neutral tweets was suggested by Pak and Paroubek (2010) who used the Twitter feed of some newspapers and magazines as a source of objective tweets. Because newspapers headlines usually convey facts not opinions, they have considered Twitter news as neutral tweets.

7 Conclusions

The emergence of the social media, such as Twitter, has provided an open medium where people express their feelings and opinions at liberty. Twitter stream generates an enormous number of opinionated messages that cover all aspects of our daily lives. Effective sentiment analysis of Twitter messages can reveal high quality information regarding the concerns and preferences of the general public. However, the use of informal language and slang in Twitter, coupled with the high frequency of misspellings, makes this task more challenging than other domains where the text is well-edited. We showed that the vocabulary size of a Twitter text corpus grows in a higher rate than movie reviews corpus. By means of linguistic preprocessing, we were able to reduce the vocabulary size in our Twitter dataset by nearly 58%. Still, up to 76% of the words occur only once in the corpus. A manual inspection of the low-frequecnt terms has revealed that the vast majority of the words that occur only once are actually hashtag phrases, elongated words, and irregular acronyms.

In this paper, we systematically evaluated the effectiveness of four different feature sets: unigrams, unigrams with part of speech tags, bigrams, and skipgrams on the performance of MNB and SVM classifiers. First, each set of features is used as a standalone feature set to train the two classifiers. Our experimental results show that appending Twitter-specific POS tags to the unigrams consistently improves the performance of the classifiers compared to the unigrams baseline. We also find that MNB and SVM behaves differently in response to the sparsity degree of phrasal features (bigrams and skipgrams). While these features improve the performance of the MNB classifier compared to the unigram baseline, they hamper performance of SVM. To the best of our knowledge, this finding is unprecedented.

Furthermore, we investigated the impact of combining unigram features with phrasal features on the classification's performance. Interestingly, the results of combining different feature sets show improvements not only compared to the unigram baseline, but also over all individual feature sets. Therefore, we conclude that using a combination of unigrams and phrasal features is a very good, yet simple solution to improve the performance of Twitter sentiment classifiers, specially if the training data is limited.

References

Amir Asiaee T, Mariano Tepper, Arindam Banerjee, and Guillermo Sapiro. 2012. If you are happy and you know it... tweet. In *Proceedings of the 21st ACM international conference on Information and knowledge management*, pages 1602–1606. ACM.

Albert Bifet and Eibe Frank. 2010. Sentiment knowledge discovery in twitter streaming data. In *Discovery Science*, pages 1–15. Springer.

Rong-En Fan, Kai-Wei Chang, Cho-Jui Hsieh, Xiang-Rui Wang, and Chih-Jen Lin. 2008. Liblinear: A library for large linear classification. *The Journal of Machine Learning Research*, 9:1871–1874.

Javi Fernández, Yoan Gutiérrez, José M Gómez, and Patricio Martınez-Barco. 2014. Gplsi: Supervised sentiment analysis in twitter using skipgrams. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014), number SemEval*, pages 294–299.

Kevin Gimpel, Nathan Schneider, Brendan O'Connor, Dipanjan Das, Daniel Mills, Jacob Eisenstein, Michael Heilman, Dani Yogatama, Jeffrey Flanigan, and Noah A Smith. 2011. Part-of-speech tagging for twitter: Annotation, features, and experiments. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: short papers-Volume 2*, pages 42–47. Association for Computational Linguistics.

Alec Go, Richa Bhayani, and Lei Huang. 2009. Twitter sentiment classification using distant supervision. *CS224N Project Report, Stanford*, 1:12.

Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H Witten. 2009. The weka data mining software: an update. *ACM SIGKDD explorations newsletter*, 11(1):10–18.

Thorsten Joachims. 1998. *Text categorization with support vector machines: Learning with many relevant features*. Springer.

Efthymios Kouloumpis, Theresa Wilson, and Johanna Moore. 2011. Twitter sentiment analysis: The good the bad and the omg! *Icwsm*, 11:538–541.

Christopher D Manning, Prabhakar Raghavan, Hinrich Schütze, et al. 2008. *Introduction to information retrieval*, volume 1. Cambridge university press Cambridge.

Mike Mintz, Steven Bills, Rion Snow, and Dan Jurafsky. 2009. Distant supervision for relation extraction without labeled data. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2-Volume 2*, pages 1003–1011. Association for Computational Linguistics.

Preslav Nakov, Zornitsa Kozareva, Alan Ritter, Sara Rosenthal, Veselin Stoyanov, and Theresa Wilson. 2013. Semeval-2013 task 2: Sentiment analysis in twitter.

Olutobi Owoputi, Brendan O'Connor, Chris Dyer, Kevin Gimpel, Nathan Schneider, and Noah A Smith. 2013. Improved part-of-speech tagging for online conversational text with word clusters. Association for Computational Linguistics.

Alexander Pak and Patrick Paroubek. 2010. Twitter as a corpus for sentiment analysis and opinion mining. In *LREC*, volume 10, pages 1320–1326.

Bo Pang, Lillian Lee, and Shivakumar Vaithyanathan. 2002. Thumbs up?: sentiment classification using machine learning techniques. In *Proceedings of the ACL-02 conference on Empirical methods in natural language processing-Volume 10*, pages 79–86. Association for Computational Linguistics.

Gerald Petz, Michał Karpowicz, Harald Fürschuß, Andreas Auinger, Václav Stříteskỳ, and Andreas Holzinger. 2013. Opinion mining on the web 2.0–characteristics of user generated content and their impacts. In *Human-Computer Interaction and Knowledge Discovery in Complex, Unstructured, Big Data*, pages 35–46. Springer.

Hassan Saif, Yulan He, and Harith Alani. 2012. Alleviating data sparsity for twitter sentiment analysis. CEUR Workshop Proceedings (CEUR-WS. org).