Twitter Sentiment Classification using Distant Supervision

Alec Go
Stanford University
Stanford, CA 94305
alecmgo@stanford.edu

Richa Bhayani
Stanford University
Stanford, CA 94305
rbhayani@stanford.edu

Lei Huang
Stanford University
Stanford, CA 94305
leirocky@stanford.edu

ABSTRACT

We introduce a novel approach for automatically classifying the sentiment of Twitter messages. These messages are classified as either positive or negative with respect to a query term. This is useful for consumers who want to research the sentiment of products before purchase, or companies that want to monitor the public sentiment of their brands. There is no previous research on classifying sentiment of messages on microblogging services like Twitter. We present the results of machine learning algorithms for classifying the sentiment of Twitter messages using distant supervision. Our training data consists of Twitter messages with emoticons, which are used as noisy labels. This type of training data is abundantly available and can be obtained through automated means. We show that machine learning algorithms (Naive Bayes, Maximum Entropy, and SVM) have accuracy above 80% when trained with emoticon data. This paper also describes the preprocessing steps needed in order to achieve high accuracy. The main contribution of this paper is the idea of using tweets with emoticons for distant supervised learning.

Categories and Subject Descriptors

I.2 [**Artificial Intelligence**]: Natural Language Processing

General Terms

Algorithms

Keywords

Twitter, sentiment analysis, sentiment classification

1. INTRODUCTION

Twitter is a popular microblogging service where users create status messages (called "tweets"). These tweets sometimes express opinions about different topics. We propose a method to automatically extract sentiment (positive or negative) from a tweet. This is very useful because it allows feedback to be aggregated without manual intervention. Consumers can use sentiment analysis to research products or services before making a purchase. Marketers can use this to research public opinion of their company and products, or to analyze customer satisfaction. Organizations can also use this to gather critical feedback about problems in newly released products.

There has been a large amount of research in the area of sentiment classification. Traditionally most of it has focused on classifying larger pieces of text, like reviews [9]. Tweets (and microblogs in general) are different from reviews primarily because of their purpose: while reviews represent summarized thoughts of authors, tweets are more casual and limited to 140 characters of text. Generally, tweets are not as thoughtfully composed as reviews. Yet, they still offer companies an additional avenue to gather feedback. There has been some work by researchers in the area of phrase level and sentence level sentiment classification recently [11]. Previous research on analyzing blog posts includes [6].

Previous research in sentiment analysis like Pang et al. [9] have analyzed the performance of different classifiers on movie reviews. The work of Pang et al. has served as a baseline and many authors have used the techniques provided in their paper across different domains. Pang et al. also make use of a similar idea as ours, using star ratings as polarity signals in their training data. We show that we can produce comparable results on tweets with distant supervision.

In order to train a classifier, supervised learning usually requires hand-labeled training data. With the large range of topics discussed on Twitter, it would be very difficult to manually collect enough data to train a sentiment classifier for tweets. Our solution is to use distant supervision, in which our training data consists of tweets with emoticons. This approach was introduced by Read [10]. The emoticons serve as noisy labels. For example, :) in a tweet indicates that the tweet contains positive sentiment and :(indicates that the tweet contains negative sentiment. With the help of the Twitter API, it is easy to extract large amounts of tweets with emoticons in them. This is a significant improvement over the many hours it may otherwise take to hand-label training data. We run classifiers trained on emoticon data against a test set of tweets (which may or may not have emoticons in them).

We present the results of our experiments and our thoughts on how to further improve results. To help visualize the util-

ity of a Twitter-based sentiment analysis tool, we also have a web application with our classifiers[1]. This can be used by individuals and companies that may want to research sentiment on any topic.

1.1 Defining Sentiment

For the purposes of our research, we define sentiment to be "a personal positive or negative feeling." Table 1 shows some examples.

Many times it is unclear if a tweet contains a sentiment. For these cases, we use the following litmus test: If the tweet could ever appear as a frontpage newspaper headline or as a sentence in Wikipedia, then it belongs in the neutral class. For example, the following tweet is considered neutral because it could have appeared as a newspaper headline, even though it projects an overall negative feeling about General Motors: *RT @Finance_Info Bankruptcy filing could put GM on road to profits (AP) http://cli.gs/9ua6Sb #Finance*. In this research, we do not consider neutral tweets in our training or testing data. We only use positive or negative tweets. Many tweets do not have sentiment, so it is a current limitation of our research to not include the neutral class.

1.2 Characteristics of Tweets

Twitter messages have many unique attributes, which differentiates our research from previous research:

Length The maximum length of a Twitter message is 140 characters. From our training set, we calculate that the average length of a tweet is 14 words or 78 characters. This is very different from the previous sentiment classification research that focused on classifying longer bodies of work, such as movie reviews.

Data availability Another difference is the magnitude of data available. With the Twitter API, it is very easy to collect millions of tweets for training. In past research, tests only consisted of thousands of training items.

Language model Twitter users post messages from many different media, including their cell phones. The frequency of misspellings and slang in tweets is much higher than in other domains.

Domain Twitter users post short messages about a variety of topics unlike other sites which are tailored to a specific topic. This differs from a large percentage of past research, which focused on specific domains such as movie reviews.

2. APPROACH

Our approach is to use different machine learning classifiers and feature extractors. The machine learning classifiers are Naive Bayes, Maximum Entropy (MaxEnt), and Support Vector Machines (SVM). The feature extractors are unigrams, bigrams, unigrams and bigrams, and unigrams with part of speech tags. We build a framework that treats classifiers and feature extractors as two distinct components. This framework allows us to easily try out different combinations of classifiers and feature extractors.

[1]The URL is http://twittersentiment.appspot.com/. This page has a link to our training data and test data. It is also a public tool that other researchers can use to build their own data sets.

2.1 Query Term

We normalize the effect of query terms. Table 1 lists example query terms along with corresponding tweets. Our assumption is that users prefer to perform sentiment analysis *about* a product and not *of* a product. When a user enters a query 'XYZ', we normalize the sentiment carried by 'XYZ' itself. For example, the tweet *XYZ is hardly interesting* should be classified as negative. If the word "XYZ" by itself has a positive sentiment, it would bias the results. Our approach is to represent each query term as a QUERY_TERM equivalence class, which allows us to normalize the effect it has on classification.

2.2 Emoticons

Since the training process makes use of emoticons as noisy labels, it is crucial to discuss the role they play in classification. We will discuss in detail our training and test set in the Evaluation section.

We strip the emoticons out from our training data. If we leave the emoticons in, there is a negative impact on the accuracies of the MaxEnt and SVM classifiers, but little effect on Naive Bayes. The difference lies in the mathematical models and feature weight selection of MaxEnt and SVM.

Stripping out the emoticons causes the classifier to learn from the other features (e.g. unigrams and bigrams) present in the tweet. The classifier uses these non-emoticon features to determine the sentiment. This is an interesting side-effect of our approach. If the test data contains an emoticon, it does not influence the classifier because emoticon features are not part of its training data. This is a current limitation of our approach because it would be useful to take emoticons into account when classifying test data.

We consider emoticons as noisy labels because they are not perfect at defining the correct sentiment of a tweet. This can be seen in the following tweet: *@BATMANNN :(i love chutney.....* Without the emoticon, most people would probably consider this tweet to be positive. Tweets with these types of mismatched emoticons are used to train our classifiers because they are difficult to filter out from our training data.

2.3 Feature Reduction

The Twitter language model has many unique properties. We take advantage of the following properties to reduce the feature space.

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. @alecmgo). An equivalence class token (USERNAME) replaces all words that start with the @ symbol.

Usage of links Users very often include links in their tweets. An equivalence class is used for all URLs. That is, we convert a URL like "http://tinyurl.com/cvvg9a" to the token "URL."

Table 1: Example Tweets

Sentiment	Query	Tweet
Positive	jquery	dcostalis: Jquery is my new best friend.
Neutral	San Francisco	schuyler: just landed at San Francisco
Negative	exam	jvici0us: History exam studying ugh.

Table 2: Effect of Feature Reduction

Feature Reduction	# of Features	Percent of Original
None	794876	100.00%
Username	449714	56.58%
URLs	730152	91.86%
Repeated Letters	773691	97.33%
All	364464	45.85%

Repeated letters Tweets contain very casual language. For example, if you search "hungry" with an arbitrary number of u's in the middle (e.g. huuuungry, huuuuuuungry, huuuuuuuuuungry) on Twitter, there will most likely be a nonempty result set. We use preprocessing so that any letter occurring more than two times in a row is replaced with two occurrences. In the samples above, these words would be converted into the token *huungry*.

Table 2 shows the effect of these feature reductions. These three reductions shrink the feature set down to 45.85% of its original size.

3. MACHINE LEARNING METHODS

We test different classifiers: keyword-based, Naive Bayes, maximum entropy, and support vector machines.

3.1 Baseline

Twittratr is a website that performs sentiment analysis on tweets. Their approach is to use a list of positive and negative keywords. As a baseline, we use Twittratr's list of keywords, which is publicly available[2]. This list consists of 174 positive words and 185 negative words. For each tweet, we count the number of negative keywords and positive keywords that appear. This classifier returns the polarity with the higher count. If there is a tie, then positive polarity (the majority class) is returned.

3.2 Naive Bayes

Naive Bayes is a simple model which works well on text categorization [5]. We use a multinomial Naive Bayes model. Class $c*$ is assigned to tweet d, where

$$c* = argmac_c P_{NB}(c|d)$$

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

In this formula, f represents a feature and $n_i(d)$ represents the count of feature f_i found in tweet d. There are a total of m features. Parameters $P(c)$ and $P(f|c)$ are obtained through maximum likelihood estimates, and add-1 smoothing is utilized for unseen features.

[2]The list of keywords is linked off of http://twitrratr.com/. We have no association with Twittratr.

3.3 Maximum Entropy

The idea behind Maximum Entropy models is that one should prefer the most uniform models that satify a given constraint [7]. MaxEnt models are feature-based models. In a two-class scenario, it is the same as using logistic regression to find a distribution over the classes. MaxEnt makes no independence assumptions for its features, unlike Naive Bayes. This means we can add features like bigrams and phrases to MaxEnt without worrying about features overlapping. The model is represented by the following:

$$P_{ME}(c|d,\lambda) = \frac{\exp[\Sigma_i \lambda_i f_i(c,d)]}{\Sigma_{c'} \exp[\Sigma_i \lambda_i f_i(c,d)]}$$

In this formula, c is the class, d is the tweet, and λ is a weight vector. The weight vectors decide the significance of a feature in classification. A higher weight means that the feature is a strong indicator for the class. The weight vector is found by numerical optimization of the lambdas so as to maximize the conditional probability.

We use the Stanford Classifier[3] to perform MaxEnt classification. For training the weights we used conjugate gradient ascent and added smoothing (L2 regularization).

Theoretically, MaxEnt performs better than Naive Bayes because it handles feature overlap better. However, in practice, Naive Bayes can still perform well on a variety of problems [7].

3.4 Support Vector Machines

Support Vector Machines is another popular classification technique [2]. We use the SVM^{light} [4] software with a linear kernel. Our input data are two sets of vectors of size m. Each entry in the vector corresponds to the presence a feature. For example, with a unigram feature extractor, each feature is a single word found in a tweet. If the feature is present, the value is 1, but if the feature is absent, then the value is 0. We use feature presence, as opposed to a count, so that we do not have to scale the input data, which speeds up overall processing [1].

4. EVALUATION

4.1 Experimental Set-up

There are not any large public data sets of Twitter messages with sentiment, so we collect our own data. Twitter has an Application Programming Interface (API)[4] for programmatically accessing tweets by query term. The Twitter

[3]The Stanford Classifier can be downloaded from http://nlp.stanford.edu/software/classifier.shtml.

[4]More information about the Twitter API can be found at http://apiwiki.twitter.com/.

Table 3: List of Emoticons

Emoticons mapped to :)	Emoticons mapped to :(
:)	:(
:-)	:-(
:)	: (
:D	
=)	

API has a parameter that specifies which language to retrieve tweets in. We always set this parameter to English. Thus, our classification will only work on tweets in English because our training data is English-only.

There are multiple emoticons that can express positive emotion and negative emotion. For example, :) and :-) both express positive emotion. In the Twitter API, the query ":)" will return tweets that contain positive emoticons, and the query ":(" will return tweets with negative emoticons[5]. The full list of emoticons can be found in Table 3.

For the training data, we use a scraper that queries the Twitter API. Periodically, the scraper sends a query for :) and a separate query for :(at the same time. This allows us to collect tweets that contain the emoticons listed in Table 3.

The Twitter API has a limit of 100 tweets in a response for any request. The scraper has a parameter that allows us to specify the frequency of polling. We found an interval of 2 minutes is a reasonable polling parameter. The tweets in our training set are from the time period between April 6, 2009 to June 25, 2009.

The training data is post-processed with the following filters:

1. Emoticons listed in Table 3 are stripped off. This is important for training purposes. If the emoticons are not stripped off, then the MaxEnt and SVM classifiers tend to put a large amount of weight on the emoticons, which hurts accuracy.

2. Any tweet containing both positive and negative emoticons are removed. This may happen if a tweet contains two subjects. Here is an example of a tweet with this property: *Target orientation :(But it is my birthday today :).* These tweets are removed because we do not want positive features marked as part of a negative tweet, or negative features marked as part of a positive tweet.

3. Retweets are removed. Retweeting is the process of copying another user's tweet and posting to another account. This usually happens if a user likes another user's tweet. Retweets are commonly abbreviated with "RT." For example, consider the following tweet: *Awesome! RT @rupertgrintnet Harry Potter Marks Place in Film History http://bit.ly/Euxi :).* In this case,

[5]At the time of this writing, the Twitter API query ":(" returns messages with ":P", which does not usually express a negative sentiment. Messages with :P are filtered out from our training data.

Table 4: List of Queries Used to Create Test Set

Query	Negative	Positive	Total	Category
40d		2	2	Product
50d		5	5	Product
aig	7		7	Company
at&t	13		13	Company
bailout	1		1	Misc.
bing	1		1	Product
Bobby Flay		6	6	Person
booz allen	1	2	3	Company
car warranty call	2		2	Misc.
cheney	5		5	Person
comcast	4		4	Company
Danny Gokey		4	4	Person
dentist	9	3	12	Misc.
east palo alto	1	2	3	Location
espn	1		1	Product
exam	5	2	7	Misc.
federer		1	1	Person
fredwilson		2	2	Person
g2		7	7	Product
gm	16		16	Company
goodby silverstein		6	6	Company
google	1	4	5	Company
googleio		4	4	Event
india election		1	1	Event
indian election		1	1	Event
insects	5	1	6	Misc.
iphone app	1	1	2	Product
iran	4		4	Location
itchy	5		5	Misc.
jquery	1	3	4	Product
jquery book		2	2	Product
kindle2	1	16	17	Product
lakers		4	4	Product
lambda calculus	2	1	3	Misc.
latex	5	3	8	Misc.
lebron	4	14	18	Person
lyx		2	2	Misc.
Malcolm Gladwell	3	7	10	Person
mashable		2	2	Product
mcdonalds	1	5	6	Company
naive bayes	1		1	Misc.
night at the museum	3	12	15	Movie
nike	4	11	15	Company
north korea	6		6	Location
notre dame school		2	2	Misc.
obama	1	9	10	Person
pelosi	4		4	Person
republican	1		1	Misc.
safeway	5	2	7	Company
san francisco	3	1	4	Location
scrapbooking		1	1	Misc.
shoreline amphitheatre		1	1	Location
sleep	3	1	4	Misc.
stanford		7	7	Misc.
star trek		4	4	Movie
summize	2		2	Product
surgery	1		1	Misc.
time warner	33		33	Company
twitter		1	1	Company
twitter api	6	2	8	Product
viral marketing	1	2	3	Misc.
visa		1	1	Company
visa card	1		1	Product
warren buffet		5	5	Person
wave s&box		1	1	Product
weka	1		1	Product
wieden		1	1	Company
wolfram alpha	1	2	3	Product
world cup		1	1	Event
world cup 2010		1	1	Event
yahoo	1		1	Company
yankees		1	1	Misc.
Total	177	182	359	-

Table 5: Categories for Test Data

Category	Total	Percent
Company	119	33.15%
Event	8	2.23%
Location	18	5.01%
Misc.	67	18.66%
Movie	19	5.29%
Person	65	18.11%
Product	63	17.55%
Grand Total	359	

the user is rebroadcasting rupertgrintnet's tweet and adding the comment *Awesome!*. Any tweet with RT is removed from the training data to avoid giving a particular tweet extra weight in the training data.

4. Tweets with ":P" are removed. At the time of this writing, the Twitter API has an issue in which tweets with ":P" are returned for the query ":(". These tweets are removed because ":P" usually does not imply a negative sentiment.

5. Repeated tweets are removed. Occasionally, the Twitter API returns duplicate tweets. The scraper compares a tweet to the last 100 tweets. If it matches any, then it discards the tweet. Similar to retweets, duplicates are removed to avoid putting extra weight on any particular tweet.

After post-processing the data, we take the first 800,000 tweets with positive emoticons, and 800,000 tweets with negative emoticons, for a total of 1,600,000 training tweets.

The test data is manually collected, using the web application. A set of 177 negative tweets and 182 positive tweets were manually marked. Not all the test data has emoticons. We use the following process to collect test data:

1. We search the Twitter API with specific queries. These queries are arbitrarily chosen from different domains. For example, these queries consist of consumer products (40d, 50d, kindle2), companies (aig, at&t), and people (Bobby Flay, Warren Buffet). The query terms we used are listed in Table 4. The different categories of these queries are listed in Table 5.

2. We look at the result set for a query. If we see a result that contains a sentiment, we mark it as positive or negative. Thus, this test set is selected independently of the presence of emoticons.

4.2 Results and Discussion

We explore the usage of unigrams, bigrams, unigrams and bigrams, and parts of speech as features. Table 6 summarizes the results.

Unigrams The unigram feature extractor is the simplest way to retrieve features from a tweet. The machine learning algorithms clearly perform better than our keyword baseline. These results are very similar to Pang and Lee [9]. They report 81.0%, 80.4%, and 82.9% accuracy for Naive Bayes, MaxEnt, and SVM, respectively. This is very similar to our results of 81.3%, 80.5%, and 82.2% for the same set of classifiers.

Bigrams We use bigrams to help with tweets that contain negated phrases like "not good" or "not bad." In our experiments, negation as an explicit feature with unigrams does not improve accuracy, so we are very motivated to try bigrams.

However, bigrams tend to be very sparse and the overall accuracy drops in the case of both MaxEnt and SVM. Even collapsing the individual words to equivalence classes does not help. The problem of sparseness can be seen in the following tweet: *@stellargirl I loooooooovvvvvveee my Kindle2. Not that the DX is cool, but the 2 is fantastic in its own right.* MaxEnt gave equal probabilities to the positive and negative class for this case because there is not a bigram that tips the polarity in either direction.

In general using only bigrams as features is not useful because the feature space is very sparse. It is better to combine unigrams and bigrams as features.

Unigrams and Bigrams Both unigrams and bigrams are used as features. Compared to unigram features, accuracy improved for Naive Bayes (81.3% from to 82.7%) and MaxEnt (from 80.5 to 82.7). However, there was a decline for SVM (from 82.2% to 81.6%). For Pang and Lee, there was a decline for Naive Bayes and SVM, but an improvement for MaxEnt.

Parts of speech We use part of speech (POS) tags as features because the same word may have many different meanings depending on its usage. For example, "over" as a verb may have a negative connotation. "Over" may also be used as a noun to refer to the cricket over, which does not carry a positive or negative connotation.

We found that the POS tags were not useful. This is consistent with Pang and Lee [9]. The accuracy for Naive Bayes and SVM decreased while the performance for MaxEnt increased negligibly when compared to the unigram results.

5. FUTURE WORK

Machine learning techniques perform well for classifying sentiment in tweets. We believe that the accuracy could still be improved. Below is a list of ideas we think could help in this direction.

Semantics Our algorithms classify the overall sentiment of a tweet. The polarity of a tweet may depend on the perspective you are interpreting the tweet from. For example, in the tweet *Federer beats Nadal :)*, the sentiment is positive for Federer and negative for Nadal. In this case, semantics may help. Using a semantic role labeler may indicate which noun is mainly associated with the verb and the classification would take place accordingly. This may allow *Nadal beats Federer :)* to be classified differently from *Federer beats Nadal :)*.

Domain-specific tweets Our best classifier has an accuracy of 83.0% for tweets across all domains. This is a very

Table 6: Classifier Accuracy

Features	Keyword	Naive Bayes	MaxEnt	SVM
Unigram	65.2	81.3	80.5	82.2
Bigram	N/A	81.6	79.1	78.8
Unigram + Bigram	N/A	82.7	83.0	81.6
Unigram + POS	N/A	79.9	79.9	81.9

large vocabulary. If limited to particular domains (such as movies) we feel our classifiers may perform better.

Handling neutral tweets In real world applications, neutral tweets cannot be ignored. Proper attention needs to be paid to neutral sentiment.

Internationalization We focus only on English sentences, but Twitter has many international users. It should be possible to use our approach to classify sentiment in other languages.

Utilizing emoticon data in the test set Emoticons are stripped from our training data. This means that if our test data contains an emoticon feature, this does not influence the classifier towards a class. This should be addressed because the emoticon features are very valuable.

6. RELATED WORK

There has been a large amount of prior research in sentiment analysis, especially in the domain of product reviews, movie reviews, and blogs. Pang and Lee [8] is an up-to-date survey of previous work in sentiment analysis. Researchers have also analyzed the brand impact of microblogging [3]. We could not find any papers that use machine learning techniques in the specific domain of microblogs, probably because these services have become popular only in recent years.

Text classification using machine learning is a well studied field [5]. Pang and Lee [9] researched the performance of various machine learning techniques (Naive Bayes, maximum entropy, and support vector machines) in the specific domain of movie reviews. We modeled much of our research from their results. They were able to achieve an accuracy of 82.9% using SVM with an unigram model.

Read [10] shows that using emoticons as labels for positive and sentiment is effective for reducing dependencies in machine learning techniques. We use the same idea for our Twitter training data.

7. CONCLUSIONS

We show that using emoticons as noisy labels for training data is an effective way to perform distant supervised learning. Machine learning algorithms (Naive Bayes, maximum entropy classification, and support vector machines) can achieve high accuracy for classifying sentiment when using this method. Although Twitter messages have unique characteristics compared to other corpora, machine learning algorithms are shown to classify tweet sentiment with similar performance.

8. ACKNOWLEDGMENTS

The authors would like to thank Christopher Manning, Nate Chambers, and Abhay Shete for providing feedback on this research.

9. REFERENCES

[1] D. O. Computer, C. wei Hsu, C. chung Chang, and C. jen Lin. A practical guide to support vector classification chih-wei hsu, chih-chung chang, and chih-jen lin. Technical report, 2003.

[2] N. Cristianini and J. Shawe-Taylor. *An Introduction to Support Vector Machines and Other Kernel-based Learning Methods*. Cambridge University Press, March 2000.

[3] B. J. Jansen, M. Zhang, K. Sobel, and A. Chowdury. Micro-blogging as online word of mouth branding. In *CHI EA '09: Proceedings of the 27th international conference extended abstracts on Human factors in computing systems*, pages 3859–3864, New York, NY, USA, 2009. ACM.

[4] T. Joachims. Making large-scale support vector machine learning practical. In B. Schölkopf, C. J. C. Burges, and A. J. Smola, editors, *Advances in kernel methods: support vector learning*, pages 169–184. MIT Press, Cambridge, MA, USA, 1999.

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

[6] G. Mishne. Experiments with mood classification in blog posts. In *1st Workshop on Stylistic Analysis Of Text For Information Access*, 2005.

[7] K. Nigam, J. Lafferty, and A. Mccallum. Using maximum entropy for text classification. In *IJCAI-99 Workshop on Machine Learning for Information Filtering*, pages 61–67, 1999.

[8] B. Pang and L. Lee. Opinion mining and sentiment analysis. *Foundations and Trends in Information Retrieval*, 2(1-2):1–135, 2008.

[9] 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.

[10] 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.

[11] T. Wilson, J. Wiebe, and P. Hoffmann. Recognizing contextual polarity in phrase-level sentiment analysis. In *Proceedings of Human Language Technologies Conference/Conference on Empirical Methods in Natural Language Processing (HLT/EMNLP 2005)*, Vancouver, CA, 2005.