"The Worst Blender I've Ever Bought!": Leveraging Natural Language Processing On Sentiment Analysis of Product Reviews

Paul Elliott
University of Oregon
paule@cs.uoregon.edu

Kanika Sood
University of Oregon
kanikas@cs.uoregon.edu

***Abstract*—Consumer-generated product reviews on commercial sites such as Amazon.com have become an important factor in the business world as they increasingly influence purchasing decisions. Extracting consumer sentiments from these online reviews would be a boon to manufacturers and companies. However, the complexity of human language makes extracting information from unfiltered natural language texts a difficult task. In this work, we leverage natural language processing to parse raw consumer reviews into a machine-readable corpus, and then leverage machine learning algorithms to classify the sentiment of each review. We evaluate three classifiers–Naive Bayes, Decision Tree, and Support Vector Machine–and analyze the effect of parametric preprocessing on the performance of all three. Our results demonstrate that a simple bag of words approach to feature extraction and minor restrictions on the feature space enable accurate sentiment classification (up to 95% of the test corpus) that outperforms similar work.**

I. Introduction

It is the twenty-first century, and so we can safely state, "the Internet means business." With the rise of online commercial marketplaces, retail giants such as Amazon.com have reshaped consumer culture. In particular, the proliferation of online, consumer-generated product reviews on such sites has become an influential factor in the market. Recent surveys suggest that consumer reviews are considered far more trustworthy than manufacturer reviews–nearly 12 times more so suggests one poll–and influence a large majority of purchasing decisions [13]. Such implications on buying habits make online product reviews an important resource for improving marketing, boosting sales, and so forth.

Given the above trends, there exists a strong incentive to extract these valuable consumer sentiments–positive or negative views, attitudes, emotions, or appraisals–and leverage them toward improving business. However, the difficulty of extracting sentiments from a corpus of reviews is somewhat self-explanatory: raw human-generated text, even when written in a common language, is nontrivial to parse by machines. Grammar and syntactic errors aside, human language is sufficiently complex enough to pose a host of issues for any parser. Consider, for example, words that change meaning between sentence contexts (i.e., "the *watch* I got for Christmas" and "the game I will *watch* tomorrow"), or phrases that shift interpretations depending on broader cultural contexts. These ambiguities can be difficult to comprehend by even human readers, let alone a computer program.

One solution to this problem involves natural language processing, or rather leveraging machine learning techniques to analyze human language media. One particular subfield of natural language processing, **sentiment analysis**, is directly applicable to the problem at hand. By extracting features from a given corpora of reviews and training a machine learning classifier to predict positive or negative labels from such features, sentiment analysis provides one way to automatically extract consumer sentiments from product reviews.

Our project involves leveraging NLTK [1], a natural language processing toolkit, for the purpose of sentiment analysis on user-generated product reviews. Using a corpus of reviews extracted from Amazon.com, we use NLTK to extract features from the reviews. We then compare the accuracy of three learners (Naive Bayes, Decision Trees, and Support Vector Machines) on predicting the polarity of reviews given their features using a range of tuning parameters. We then analyze the relative performance of each algorithm, and furthermore the influence of different tuning parameters on classification.

Our results show that we can achieve high in-domain classification accuracy of product review sentiments using a simple bag of words approach to feature extraction. Our best classification accuracy comes from a Naive Bayes classifier using minute restrictions on the feature space to reduce overfitting. This best-case performance greatly outperforms our baseline, which we draw from a previous work on our dataset [2]. We show Support Vector Machines to perform decently in the classification task as well. Beyond a comparison of accuracy, our results also illustrate the influence of tuning parameters such as filtering common words and including punctuation characters. Taken together, we demonstrate that a well-defined feature space, reasonable restrictions on feature extraction, and a simple bag of words approach is well-suited for sentiment analysis of consumer-generated product reviews.

The rest of the paper is organized as follows: Section II is an overview of the topics of natural language processing and sentiment analysis; Section III details our methodology for this work; Section IV covers our experiments and analyzes their results; and in Section V we draw our conclusions.

II. Background

A. Natural Language Processing

Natural language processing (NLP) has its roots in the early nineteen fifties with Alan Turing and the Turing test, but our work is better identified with the modern approach to NLP from the field of machine learning. This body of more recent work generally focuses on leveraging computer systems to understand and manipulate natural language text or speech for useful purposes [3]. In practice, such tasks involve running machine learning algorithms on features extracted from a natural language corpus, where the goal is to derive the proper rulesets from the given features.

While NLP can be used for a number of ends, some of the primary tasks include:

- Automatic summarization, or "abstracting";
- Automatic translation;
- Natural language generation and understanding;
- Sentence parsing, part-of-speech tagging, named entity recognition (NER);
- Sentiment analysis; and
- Speech recognition and segmentation.

Our work focuses on sentiment analysis, which we will highlight in the following section.

B. Sentiment Analysis

Sentiment analysis–the computational study of opinions, sentiments, subjectivity, evaluations, attitudes, appraisal, affects, views, emotions, etc., expressed in text–is a growing body of research in natural language processing [5], [11], [8]. Work in this area covers a broad range of corpora from news articles to movie reviews to Twitter feeds ([4], [9], [12]), and leverages a variety of strategies ([7], [14], [6], [10]). In general, the focus of such work involves extracting positive or negative sentiments from the given corpus with consistent accuracy.

Our work is a specific follow-up of Blitzer et al's sentiment analysis study on product reviews [2]. Their research focuses on leveraging structured correspondence learning (SCL), a means of relating contextual features to more universal features, and A-distance to enable accurate sentiment classification across different product domains (i.e., electronics, housewares, books, etc). Their methodology demonstrably improves the accuracy of cross-domain predictions over the baseline, and achieves comparable performance to in-domain classification accuracy for certain domain pairs. While we did not have time to consider cross-domain predictions, we use Blitzer et al's experimental setup and gold standard classifier as a baseline.

III. Methods

In this section we cover our methodology for this work. We will highlight the dataset we used and how we handled them, the feature space we decided to explore and the classifiers we chose to focus on.

A. Product Review Dataset

The dataset we used is the same set of Amazon.com product reviews as in [2]. It contains unordered lists of reviews within a range of product categories. Each review consists of a rating (0-5 stars), a reviewer name and location, a product name, review title, date and the review text.

In order to use Blitzer et al's results as a baseline, we handled the dataset in the same fashion as their work. From the set of product categories, we selected the same four categories:

- $books$;
- dvd;
- $electronics$; and
- $kitchen\ \&\ housewares$ (referred to as $kitchen$).

For each category, we labeled reviews according to Blitzer et al: those with a rating of less than 3 stars were labeled negative, those with ratings greater than 3 stars were labeled positive, and the remainder were discarded due to their ambiguity. Each category was left with 2000 reviews, 1000 positive and 1000 negative. Once labeled, we then extracted all the review texts from the XML-like review files to create the final labeled instances of our corpus.

B. Features

With the corpus ready, we turn to our work on identifying useful features for the classification task. Whereas our baseline leverages SCM for feature extraction in order to improve cross-domain classifications, we are interested primarily in in-domain classification. We determined that it may be possible to achieve more accurate classifications using a mundane approach to feature extraction *if the right features are selected*. Following this, we resolved to use a bag of words approach, which essentially represents review text as a collection of word features irrespective of order, to extract features from the reviews.

In order to determine the best possible bag of words features, we chose to test the following tuning parameters:

- *N-grams* - We determined to test unigram, bigram, and trigram features (representing single word and consecutive word pairs and triples, in that order).
- *Restrictions on feature counts* - We limit the number of features used in two important ways. First, we may specify a minimum value for information gain that a given feature must achieve to be included in the feature set. Second, we may specify an upper bound n on the number of features, and select the top n features with respect to information gain.
- *Stopwords filtering* - Filtering stopwords involves preprocessing the review text to remove common words such as "the," "is," etc. This tuning parameter may help remove irrelevant features from the feature set.
- *Punctuation* - With similar justification as stopwords filtering, we optionally keep punctuation characters in the review text, which may indicate more salient features (i.e., "good!" as opposed to "good").

In addition to these tuning parameters, we also discarded a couple other possibilities for our experiments. First, we determined not to test int-value features (a tuple of $(feature, count)$ where $count$ represents the number of times $feature$ appears in the review text). In our preliminary tests, we found that int-value features generally led to far worse performance than boolean features, likely as a result of overfitting (i.e., treating $(cat, 1)$ and $(cat, 2)$ as separate features). We also determined not to use cross-fold validation. In preliminary tests, cross-fold averages of training error tended to be very similar to non-folded calculations of training error, and took considerably longer to run. Since our dataset is fairly sizeable, and in the interest of time, we excluded cross-fold validation in our training procedure.

With the dataset and features defined, we turn to the machine learning algorithms that we focus on in this work.

C. Classifiers

We decided to compare three separate learning algorithms for our product review sentiment analysis task. The three learners we selected are:

- *Naive Bayes* - Naive Bayes learning is a well-documented approach for classifying text using a bag of words approach. It is quite robust to irrelevant features as the insignificant features can nullify each other without drastic impact on the results.
- *Support Vector Machine (SVM)* - SVMs are a standard choice in binary classification tasks, and in particular, they perform well on datasets with numerous attributes (such as a bag of words approach to text classification).
- *Decision Trees* - Decision trees are simple to implement and can represent any boolean function, therefore making them a good baseline comparison to the other two learning algorithms.

IV. Experiments

In the following sections, we detail the experiments we conducted and analyze the results we found, both between learners and between various tuning parameters.

A. Experimental Procedure

We proceeded with our experiments in the following manner. For each of the four product categories ($books$, dvd, $electronics$, $kitchen$), we split the 2000 reviews (1000 positive and 1000 negative) randomly into a training set of 1600 reviews and a test set of 400 reviews. We then conducted the following tests:

1) *Select learner*: Naive Bayes, Decision Tree, SVM;
2) *Select feature*: unigrams, bigrams, trigrams;
3) *Select feature restrictions*: all features or top 10,000 features, and minimum information gain $i \in \{2, 4, 8, 16\}$ or no minimum;
4) *Select additional tuning parameters*: retain punctuation characters, remove English stopwords, a combination of the two, or neither;

Category	N-grams	Min-score	Max-feats	Use-punc	Filter-stop
books	3	N/A	10,000	No	No
dvd	2	N/A	10,000	No	Yes
electronics	3	2	10,000	No	No
kitchen	2	N/A	10,000	No	No

TABLE I
CONFIGURATIONS FOR BEST PERFORMANCE USING NAIVE BAYES.

All told, this amounts to:
$4\ categories * 3\ learners * 3\ feature\ types * 10\ feature\ restrictions * 4\ additional\ options = 1440\ tests$.

For each test, we trained the given classifier on the training data using the appropriate features and tuning parameters, and then tested the resulting classifier on the test data. We record the learner's accuracy on the classification task, as well as precision and recall for positive and negative labels. We analyze the results of these tests in the following sections.

B. Learner Comparison

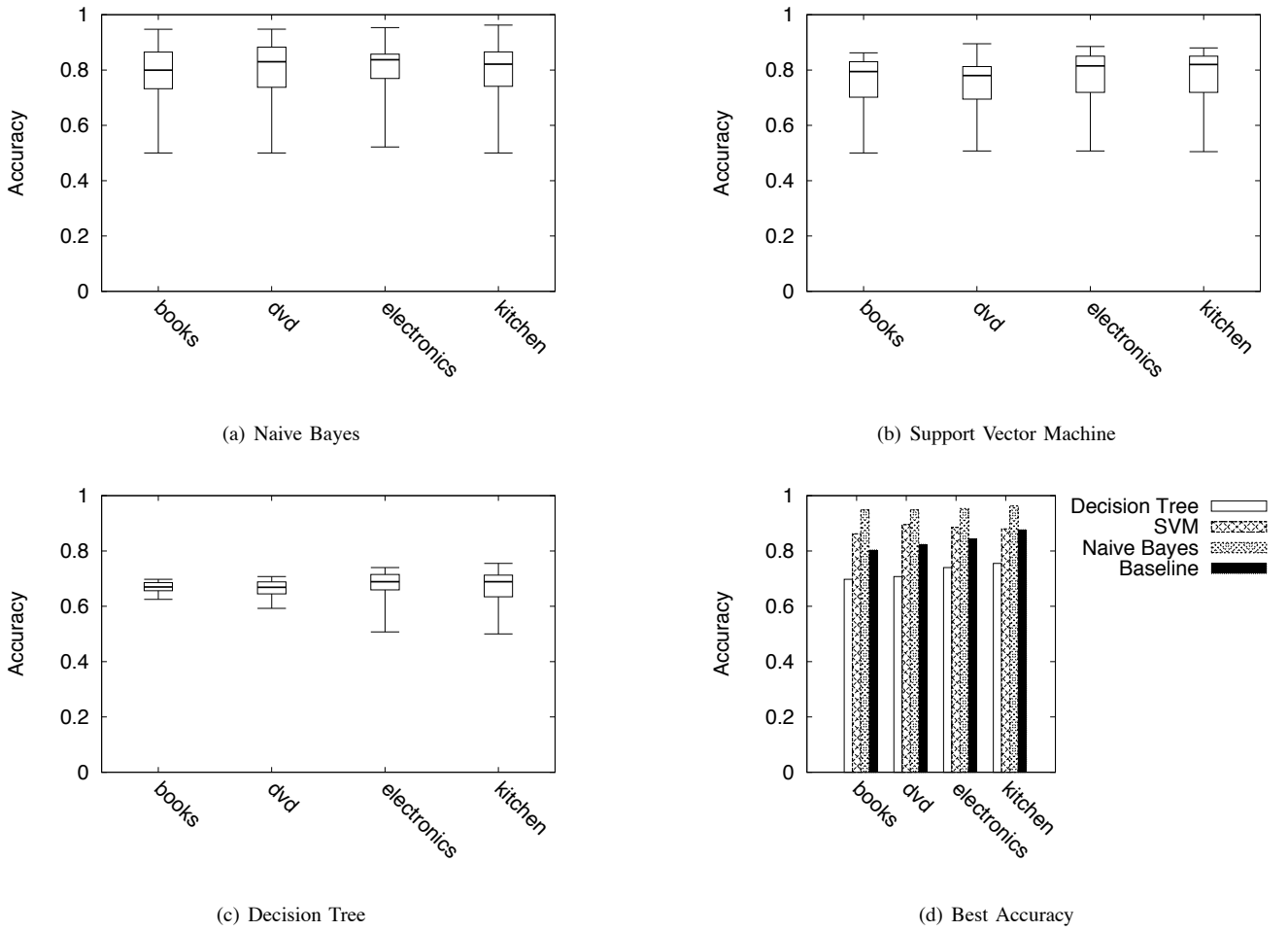
As mentioned, we compare classification accuracy of three separate learners. Figure 1 depicts the relative accuracy of each classifier across all tuning parameter configurations (including feature types and restrictions). In general, we see that Naive Bayes classification outperforms our SVM, which generally outperforms our Decision Tree. These results suggest that for N-grams features, Naive Bayes may be best suited for product review sentiment analysis.

We also compare our results to a baseline of the gold standard in-domain classification accuracy from [2]. Figure 1(d) illustrates this comparison. Here we see that our best iteration using the Naive Bayes classifier outperforms the gold standard in all four categories. A well-tuned Naive Bayes classifier, therefore, is an appropriate choice for extracting sentiments from user-generated product reviews. For the remainder of this analysis, we focus on Naive Bayes results.

C. Analysis of Best Performance

Our best performance accuracies with the Naive Bayes classifier stem from a small range of features and tuning parameters. Table I illustrates these configurations, where *Min-score* is the minimum information gain allowed for a given feature, *Max-feats* is the number of n best features used, *Use-punc* indicates whether we used punctuation characters, and *Filter-stop* indicates whether we filtered English stopwords.

Among the similarities between the four cases, we note that each caps the number of features at the top 10,000 with respect to information gain. We believe that this helps prevent our model from overfitting to irrelevant features. In addition, all cases filter punctuation characters from the feature set, and all but one case includes English stopwords as well. While this runs contrary to our hypothesis that including punctuation and filtering stopwords may increase performance, we conclude that for these best cases, filtering punctuation may have reduced overfitting and including stopwords retained



(a) Naive Bayes

(b) Support Vector Machine

(c) Decision Tree

(d) Best Accuracy

Fig. 1. **Classification accuracy for each category across all tuning parameter configurations, and a comparison of each classifier's best accuracy against the gold standard in [2] (note: Decision Tree tests were incomplete for the books and dvd category, though suggested comparable trends)**

some valuable features. We return to the general effect of these parameters in a later section.

Our best performance tests also demonstrate the importance of using larger than unigram features for our Naive Bayes bag of words classifier. All four of our best case scenarios leverage bigram or trigram features. We believe that this helps capture more semantic meaning from the given corpus, allowing meaningful features such as "best directing" rather than more ambiguous unigrams (for example, "best" as in "this is a C-list cast at *best*"). Deriving more semantic meaning from features contributes to more accurate classifications.

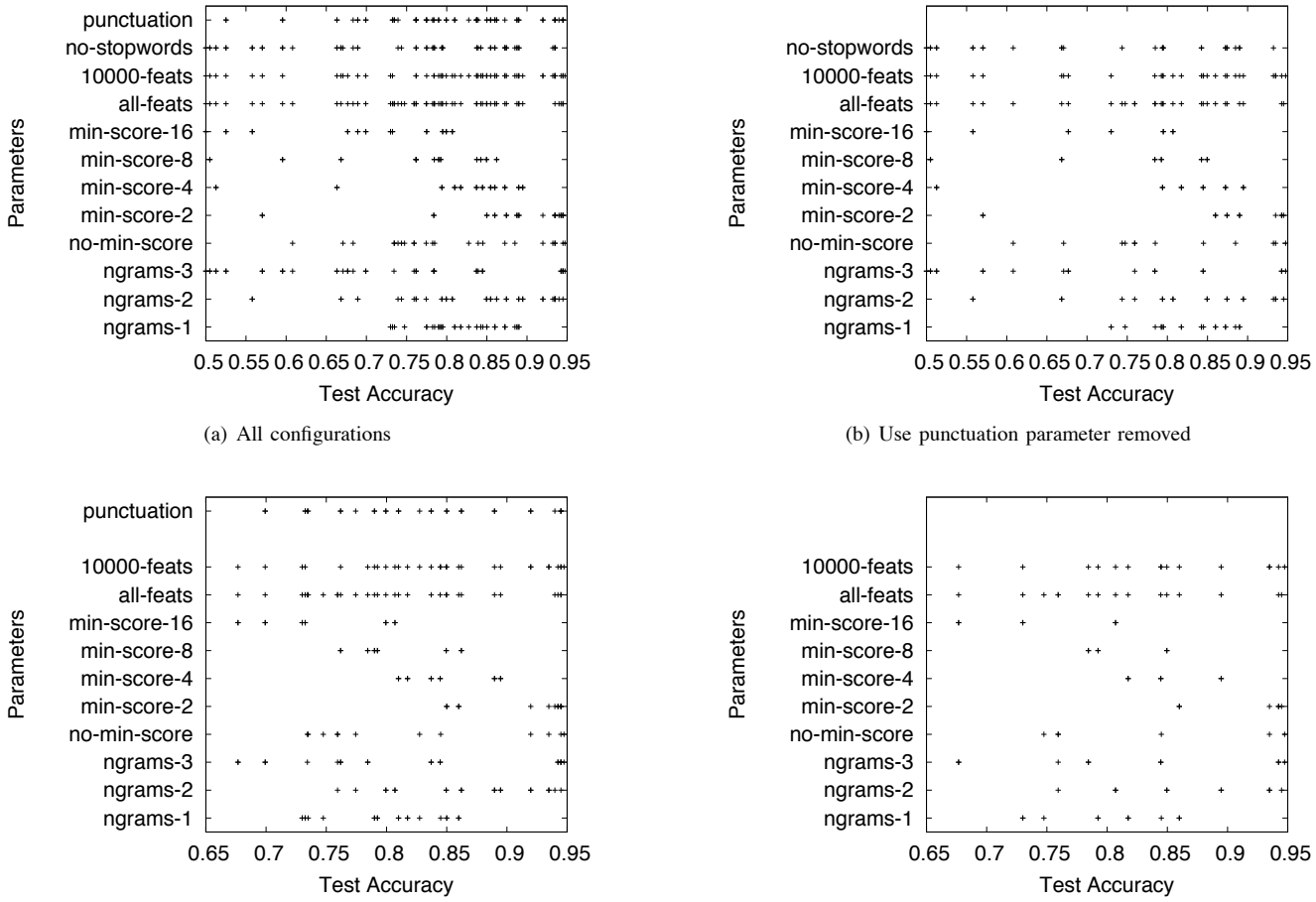
Lastly, we note that in general, our learner preferred features associated with negative sentiment labels over those associated with positive labels. In all instances, information gain for negative features ("don't buy", "waste of money", "bad acting") was significantly higher than for positive features. In addition, both the precision and recall for negative labels outpaced the same for positive labels by 4-5% on average across the four cases. We attribute this pattern to the nature of our corpus; negative sentiments are likely to be more overt and less ambiguous in product reviews, and therefore represent more informative features.

D. Analysis of Worst Performance

Our worst-case performances in the classification task are trivial to explain, but we cover them here for the sake of completeness. In all cases, worst-case performance stems from an over-reduction of the feature space. As restrictions on the maximum number of features and the minimum score for features increase, the classifier can unwittingly purge out nearly all useful features from the review text. Filtering stopwords compounds this problem, leaving even fewer features to select. We conclude that while restrictions on feature space help reduce overfitting up to a point, beyond that threshold they hinder classification. We show the degradation of performance as these parameters increase too far in the next section.

E. Tuning Parameters

Finally, we conclude our discussion of experiments with an analysis of the tuning parameters and their effect on classification performance. Figure 2(a) illustrates the accuracy



(a) All configurations

(b) Use punctuation parameter removed

(c) Filter English stopwords parameter removed

(d) Both parameters removed

Fig. 2. **Naive Bayes classification accuracy given the presence of tuning parameters for the books category.**

distribution of Naive Bayes classification with respect to given tuning parameters (only the *books* category is shown for brevity–other categories demonstrate similar results).

This graph depicts broad trends in the effect of certain tuning parameters on classification performance. For example, we see that bigrams leverage the other tuning parameters better than unigram features, allowing better accuracy with bigrams in some cases. Trigrams also achieve fairly good performance, but in general the average performance is much lower (this could be a unique interaction with trigrams and other tuning parameters). We also note that maintaining a minimum score for information gain on features hits a peak in performance around a score of two, and worsens rapidly thereafter.

Finally, we make particular note of the effect of using punctuation and filtering stopwords in our tests. The remaining three graphs in Figure 2 illustrate the resulting classification accuracy if one or both of these features were not used. While the best case performances still remain, a considerable amount of the *reasonably good* tests are lost (including many of those near or above the baseline accuracy). We conclude that in general, using punctuation characters and filtering stopwords are good parameters for bag of words sentiment analysis.

V. Conclusion

In this work we compare the performance of three learning algorithms across a variety of parameters on the task of sentiment analysis of consumer-generated product reviews. Our results indicate that a simple bag of words approach to feature extraction enables Naive Bayes and SVM classifiers to achieve very high accuracy on our corpus, particularly with the inclusion of minor tuning parameters. We further detail the effect of these tuning parameters on overall accuracy, demonstrating that preprocessing such as removing stopwords and including punctuation characters has great effect.

We conclude that much more work could be done in this area. Our future work would involve a more in-depth analysis of our best case performances and a thorough investigation of other possibilities in feature extraction. Had time permitted, we would have liked to investigate meta natural language features such as part of speech and sentence structure. In addition, we could continue this work with a look at cross-domain classification. For now, we are satisfied in laying an important groundwork for future study.

REFERENCES

[1] S. Bird, E. Klein, and E. Loper. *Natural language processing with Python*. O'Reilly Media, Incorporated, 2009.

[2] J. Blitzer, M. Dredze, and F. Pereira. Biographies, bollywood, boomboxes and blenders: Domain adaptation for sentiment classification. In *Annual Meeting-Association For Computational Linguistics*, volume 45, page 440, 2007.

[3] G. G. Chowdhury. Natural language processing. *Annual Review of Information Science and Technology*, 37(1):51–89, 2003.

[4] N. Godbole, M. Srinivasaiah, and S. Skiena. Large-scale sentiment analysis for news and blogs. In *Proceedings of the International Conference on Weblogs and Social Media (ICWSM)*, volume 2, 2007.

[5] B. Liu. Sentiment analysis and opinion mining. *Synthesis Lectures on Human Language Technologies*, 5(1):1–167, 2012.

[6] A. L. Maas, R. E. Daly, P. T. Pham, D. Huang, A. Y. Ng, and C. Potts. Learning word vectors for sentiment analysis. In *Proceedings of the 49th annual meeting of the association for computational Linguistics (acL-2011)*, 2011.

[7] T. Mullen and N. Collier. Sentiment analysis using support vector machines with diverse information sources. In *Proceedings of EMNLP*, volume 4, pages 412–418, 2004.

[8] T. Nasukawa and J. Yi. Sentiment analysis: Capturing favorability using natural language processing. In *Proceedings of the 2nd international conference on Knowledge capture*, pages 70–77. ACM, 2003.

[9] A. Pak and P. Paroubek. Twitter as a corpus for sentiment analysis and opinion mining. In *Proceedings of LREC*, volume 2010, 2010.

[10] B. Pang and L. Lee. A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In *Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*, page 271. Association for Computational Linguistics, 2004.

[11] B. Pang and L. Lee. *Opinion mining and sentiment analysis*. Now Pub, 2008.

[12] B. Pang, L. Lee, and S. Vaithyanathan. 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, 2002.

[13] Various. Online reputation monitoring, 2012.

[14] C. Whitelaw, N. Garg, and S. Argamon. Using appraisal groups for sentiment analysis. In *Proceedings of the 14th ACM international conference on Information and knowledge management*, pages 625–631. ACM, 2005.