Comparative Experiments on Sentiment Classification for Online Product Reviews

Hang Cui *
Department of Computer Science
School of Computing
National University of Singapore
cuihang@comp.nus.edu.sg

Vibhu Mittal
Google Inc.
vibhu@google.com

Mayur Datar
Google Inc.
mayur@google.com

Abstract

Evaluating text fragments for positive and negative subjective expressions and their strength can be important in applications such as single- or multi- document summarization, document ranking, data mining, *etc*. This paper looks at a simplified version of the problem: classifying online product reviews into positive and negative classes. We discuss a series of experiments with different machine learning algorithms in order to experimentally evaluate various trade-offs, using approximately 100K product reviews from the web.

Introduction

A large amount of Web content is subjective and reflects peoples' opinions. With the rapid growth of the Web, more and more people write reviews for all types of products and services and place them online. It is becoming a common practice for a consumer to learn how others like or dislike a product before buying, or for a manufacturer to keep track of customer opinions on its products to improve the user satisfaction. However, as the number of reviews available for any given product grows[1], it becomes harder and harder for people to understand and evaluate what the prevailing/majority opinion about the product is.

Sentiment classification, also known as affect or polarity classification, attempts to address this problem by (i) presenting the user with an aggregate view of the entire data set, summarized by a label or a score, and (ii) segmenting the articles/text-fragments into two classes that can be further explored as desired. While many review sites, such as Epinions, CNet and Amazon, help reviewers quantify the positivity of their comments, sentiment classification can still play an important role in classifying documents that do not have explicit ratings. Often, web sites, such as personal blogs, have user reviews with personal experiences in using a particular product without giving any score. The review comments from these sites are valuable because they cover a lot more products than those formal review sites. For instance, in the case of the camera "Nikon D70", we get only 135 reviews from Froogle[2], which gathers user and editorial reviews from several sites, but get over 759,000 hits by searching "Nikon D70 user review" in Google. This demonstrates the need for algorithmic sentiment classification in order to digest this huge repository of "hidden" reviews. Robust classification of review polarities is the premise for subsequent mining tasks.

Sentiment classification has recently attracted much attention from the natural language processing community. Researchers have investigated the effectiveness of different machine learning algorithms using various linguistic features to develop sentiment classifiers. See the related work section for a list of references. However, we identify two drawbacks in the current research:

1. Large-scale, real-world data-sets: Most previous work was conducted on relatively small, experimental data sets employing at most a few thousand articles. Building sentiment classifiers for web-scale processing, with all the attendant problems of pre-processing, spelling/grammar mistakes, broken HTML, *etc*., brings in a whole set of other problems. Data sets are in the hundreds of thousands, sometimes millions, and are neither clean nor consistent. Algorithms need to be efficient and subtle language effects become visible.

2. Unigrams vs. n-grams: Current work mostly focuses on using unigrams and bigrams – rather than higher order n-gram models – to capture sentiments in the text. Pang *et al.* (2002) found that (surprisingly) unigrams beat other features in their evaluations. Similarly, Dave *et al.* (2003) experimentally showed that trigrams and higher failed to show consistent improvement. We conjecture that those experiments were hindered by the small training corpora, and thus were not able to show the effectiveness of high order n-grams ($n \geq 3$) in discerning subtleties in expressing sentiments. Unfortunately, lower order n-grams, such as unigrams and bigrams, are unable to capture longer range dependencies, and thus often lead to problems in classification accuracy. For instance, *excellent* is usually used for positive comments; sometimes, it may be used ironically as in *the CD player is an excellent noise maker*.

*This work was done by this author when he interned at Google Inc., Mountain View, CA.

Copyright © 2006, American Association for Artificial Intelligence (www.aaai.org). All rights reserved.

[1]For instance, there are several dozen reviews of the Nikon D-70 camera, just on the amazon.com website.

[2]http://froogle.google.com

To remedy these problems, we make the following contributions in this paper:

- We conduct experiments on a corpus of over 200k online reviews with an average length of over 800 bytes crawled from the Web. Such a large-scale data set allows us not only to train language models and cull high order n-grams as features, but also to study the effectiveness and robustness of classifiers in a simulating context of the Web.
- We study the impact of higher order n-grams ($n \geq 3$). Compared to previous published work – we show that there is a benefit to use higher order n-grams beyond uni- and bi-grams. While we also conduct experiments using other features beyond n-grams, such as POS tags, parse trees, external thesauri, *etc*., we do not discuss them here in detail due to lack of significant improvements.
- We study multiple classification algorithms for processing large-scale data. We employ three algorithms: (i) Winnow (Nigam & Hurst 2004), (ii) a generative model based on language modeling, and (iii) a discriminative classifier that employs projection based learning (Shalev-Shwartz *et al.* 2004). While discriminative models, such as SVM, have been employed in previous sentiment classifiers (*e.g.* (Mullen & Collier 2004)), it was not clear to us whether they are capable of processing online documents incrementally. Our experimental results show that the discriminative model significantly outperforms the other two kinds of models.

The paper is organized as follows. We review the related work in the next section. We then discuss the classifiers and linguistic features. Finally, we report the experimental results and conclude the paper with future work.

Related Work

Sentiment classification aims to distinguish whether people like/dislike a product from their reviews. It has emerged as a proper research area. While it is still in its preliminary stage, there is much existing work in various disciplines. In this paper, restricted by our goal, we focus the related work only on sentiment classification for product reviews. We first review the work on classification of words according to their semantic orientation. The semantically oriented words could be considered evidence to article polarity. Next, we come to the work that classifies polarities on the article level.

Some researchers relate the polarity of an article to semantic orientation of the words in the article. Hatzivassiloglou and McKeown (1997) examined the semantic orientation of adjectives: they employed a supervised learning algorithm and a list of manually labeled seed adjectives to find out more adjectives that have the same or opposite semantic orientation according to textual patterns. While they evaluate on adjectives only, Turney and Littman (2003) extended the vocabulary to both adjectives and adverbs. They employed point-wise mutual information (PMI) and latent semantic analysis to calculate the semantic orientation of the extracted words according to their co-occurrences with the seed words, such as *excellent* and *poor*. They proposed determining the polarity of an article by averaging the semantic orientation of words in the article. However, this simple strategy fails as words are often used in articles that have the opposite orientation due to language variations.

Instead of limiting the analysis on the word level, another stream of research performs sentiment classification on the article level. Pang *et al.* (2002) tried to classify movie reviews into positive/negative by using three different classifiers – Naíve Bayes, Maximum Entropy and SVM. They tested different feature combinations including unigrams, unigrams+bigrams and unigrams+POS (part-of-speech) tags, *etc.* The experimental results showed that SVM combined with unigrams obtained the best performance. In their recent work (Pang & Lee 2004), they added in subjectivity detection to avoid the sentiment classifier from dealing with irrelevant "objective" sentences. To extend their work, we explore the use of high order n-grams in sentiment classification because we believe that longer phrases tend to be less ambiguous. In addition, they found SVM performs the best. As reported in their work, the classification performance of product reviews is worse than that of normal topical text categorization. One of the main difficulties is that people typically use both positive and negative words in the same review, regardless of the rating score. As such, we hypothesize that a discriminative classifier could gain more strength in differentiating the mixed sentiments. We experimentally compare a discriminative model with a generative model by language modeling to verify this hypothesis. Nigam and Hurst (2004) applied simple online classifier Winnow to classifying polarity of documents. They showed that human agreement can merely achieve 75%-80% of precision and recall on polarity prediction. The recall obtained by Winnow is very poor, achieving only 43% for positive reviews and 16% for negative reviews. We also include Winnow as a classifier in our evaluations.

As is to be expected, others have previously also worked with data crawled from the Web (though not at a comparable level of scale). Dave *et al.* (2003) conducted experiments on reviews gathered from CNet and Amazon. To gain better performance, they tried to employ more complex strategies, such as proper name substitution, dependency parsing and negating modifications. However, the experimental results indicate that except stemming – which improves the unigram baseline – the linguistic features inversely hurt the performance. While they showed that n-grams (up to trigrams) can improve the performance of the classifier in one of the tests, it is not clear if the improvement is consistent when scaling up to larger corpus and if higher order n-grams are still effective. We try to address these unanswered questions in this study. Hu and Liu (2004) intended to classify sentences based on product attributes. They mainly classify the polarity of sentences based on the semantic orientation of words, determined by the method introduced in (Hatzivassiloglou & McKeown 1997). Improving upon the work of Hu and Liu (2004), Popescu and Etzioni (2005) concerned extracting product features and opinions from reviews. A salient feature of this work is the use of search engine hit-counts to compute point-wise mutual information (PMI) between candidate phrases to select product features and the use of *relaxation labeling* technique to assign polarity. Similarly, Wilson *et al.* (2005) focused on the task of phrase-level sen-

timent analysis. They followed a two-tiered approach: detecting whether a phrase is polar or neutral; and in the case of a polar phrase, trying to disambiguate its polarity.

Classifiers and Features

In this section, we discuss in detail the three classifiers and n-grams as linguistic features. To adapt the classifiers to the Web, we conduct feature selection to reduce feature dimensionality.

Classifiers

Passive-Aggressive (PA) Algorithm Based Classifier Introduced by Shalev-Shwartz *et al.* (2004), the Passive-Aggressive (PA) algorithms are a family of margin based online learning algorithms for binary classification. PA algorithms work similarly to support vector machines (SVM) and can be viewed as an online version of a SVM. PA algorithms try to find a hyperplane that separates the instances into two half-spaces. The margin of an example is proportional to the example's distance to the hyperplane. When making errors in predicting examples, PA algorithm utilizes the margin to modify the current classifier. The update of the classifier follows the constraints: the new classifier should be a close proximity to the current one (passive update) while achieve at least a unit margin on the most recent example (aggressive update). We illustrate the PA algorithm in Figure 1.

The reason that we choose the PA algorithm, instead of SVM, to construct the discriminating classifier is two-fold: (1) The PA algorithm follows an online learning pattern, *i.e.*, it updates the hypothesis (or the separating hyperplane) upon seeing the next example and ensures the update does not degrade the performance of the classifier. This property is very attractive to Web applications because once the classifier is deployed, it will see samples in a sequential manner and should be able update itself along with seeing more samples. (2) The PA algorithm has a theoretical loss bound, which makes the performance of the classifier predictable.

We follow the default setting of the PA algorithm and set the number of iterations to 10 due to cross-validation experiments on a small data set.

INITIALIZE: $w_1 = (0 \ldots 0)$ as parameters of the classifier
For $t = 1, 2, \ldots$

- receive instance: $x_t \in R^n$
- predict: $\hat{y_t} = sign(w_t \cdot x_t)$
- receive correct label: $y_t \in \{-1, +1\}$
- suffer loss: $l_t = \max\{0, 1 - y_t(w_t \cdot x_t)\}$
- update:
  1. set: $\tau_t = \frac{l_t}{\|x_t\|^2}$
  2. update: $w_{t+1} = w_t + \tau_t y_t x_t$

Figure 1: Outline of the passive-aggressive algorithm for binary classification

Language Modeling (LM) Based Classifier Language modeling (LM) (Manning & Schütze 1999) is a generative method that calculates the probability of generating a given word sequence, or string. In n-gram language modeling, the probability of string s is represented by the product of the conditional probabilities of its n-grams, whose probabilities are conditional on the preceding $n-1$ words. As such, given string s as $w_1 \ldots w_l$, the probability of s is

$$P(s) = \prod_{i=1}^{l} P(w_i | w_1^{i-1})$$

where w_i^j denotes the words $w_i \ldots w_j$.

Due to the limitations of training data, n-gram language modeling often suffers from data sparseness as some n-grams in the test data, especially higher-order n-grams, are never seen during training. Maximum likelihood estimation fails when encountering such unseen n-grams. Smoothing is one technique to combat with data sparseness. In our experiments, we adopted Good-Turing estimation. Good-Turing states that an n-gram that occurs r times should be treated as if it had occurred $r*$ times, where

$$r* = (r+1)\frac{n_{r+1}}{n_r}$$

and n_r denotes the number of n-grams that occur r times in the training data.

We employ the CMU-Cambridge Language Modeling Toolkit (Clarkson & Rosenfeld 1997) for the training and testing of language models. We train separate language models for positive and negative reviews, respectively. The polarity of a test sample is determined by the ratio of its generative probabilities by the positive and negative language models.

Winnow Classifier Winnow is an online learning algorithm and has been employed by Nigam and Hurst (2004) for sentiment classification. Winnow learns a linear classifier from bag-of-words of documents to predict the polarity of a review x:

$$h(x) = \sum_{w \in V} f_w c_w(x)$$

where $c_w(x) = 1$ if word w appears in review x and 0 otherwise. f_w represents the weight of w. If $h(x) > V$, Winnow predicts the document as positive and negative otherwise. Winnow is trained by the procedure:

1. Initialize all f_w to 1.
2. For each labeled review x in the training set:
   - Calculate $h(x)$.
   - If the review is positive but Winnow predicts it as negative (*i.e.* $h(x) < V$), update the weight f_w where $c_w(x) = 1$ by $f'_w = f_w \times 2$.
   - If the review is negative but Winnow predicts it as positive (*i.e.* $h(x) > V$), update the weight f_w where $c_w(x) = 1$ by $f'_w = f_w \ / \ 2$.

It is worth noting that according to Nigam and Hurst (2004), the threshold V is defined as $|V|$, which is the size of

the vocabulary which consists of non-trivial words from the training corpus. However, in our experiments, we use up to 6-grams in the vocabulary. Given the multitude of our training data, $|V|$ is too high as a threshold to update the feature weights properly. According to our experiments on a validation data set, we set V to $|V|/2$ for training and $|V|/128$ for testing to get the best performance.

N-grams as Linguistic Features

One of the main contributions of this work is that we explore the role of high order n-grams as features in discerning the subtleties reflecting sentiments. The underlying motivation stems from the fact that high order n-grams ($n \geq 3$) are more precise and deterministic expressions than unigrams or bigrams. For instance, given "recommend" or "highly recommend", it is still hard to tell whether they are positive because a widely used phrase in negative reviews is "*highly recommend staying away from it*". In addition, while it is difficult to model positive and negative expressions by lexico-syntactic patterns due to extreme variety, it would be helpful to employ high order n-grams to approximate surface patterns to capture the sentiment in text.

Note that when we mention n-grams, we refer to a set of all $1 \ldots n$ order n-grams. While this violates the independence between features, we take it because: (1) As suggested by Pang *et al.* (2002), combining unigrams and bigrams gives much higher performance than using only bigrams. (2) The classifiers we employ do not require independent features.

In this study, we set $n = 6$ for n-grams. From the training data, we extracted in total 3.02M n-grams after removing those appearing in less than $m (m = 20)$ reviews. To reduce its computational complexity so as to be adapted in Web applications, we pare down the feature vector by performing feature selection. We calculate χ^2 scores for each n-gram (Yang & Pedersen 1997). Given term t and class c, let A be the number of times t and c co-occur, B be the number of times t occurs without c, C be the number of times c occurs without t, D be the number of times neither t nor c occurs, and N be the number of documents. χ^2 is calculated by:

$$\chi^2(t,c) = \frac{N \times (AD - CB)^2}{(A+C) \times (B+D) \times (A+B) \times (C+D)}$$

The n-grams are sorted in descending order by their χ^2 scores. We take the top M ranked n-grams as features in the classification experiments.

Evaluations

We have three questions to answer in the evaluations: (1) With a large-scale corpus, would high order n-grams compensate for the ambiguity brought by using only unigrams, such as to improve the classification performance? (2) Which kind of classifier is most appropriate for sentiment classification for Web documents? (3) Would fewer features after feature selection decrease the performance?

Before coming to the results, we first discuss the setup of the experiments.

Evaluation Setup

Data Set We accumulate reviews about electronic products like digital cameras, laptops, PDAs, MP3 players, *etc.* from Froogle. Each review comes with the full text and the rating score by the reviewer. The size of the whole corpus is around 0.4GB, including a total of over 320k product reviews about over 80k unique products. These reviews are crawled from prominent sites, such as cnet.com, ciao.co.uk and shopping.yahoo.com. The average length of the reviews is 875 bytes. We construct the training and test data by adopting the following strategy: Let R be the highest score for each rating scheme ($R = 5$ *or* 10) and the lowest score is 1. Let r be the rating from the user. We take those reviews with $r = R$ as positive examples and those with $r = 1$ as negative examples to form the training data. We take those reviews with $r = R - 1$ as positive instances and those with $r = 2$ as negative instances for testing. We intentionally make the training data more pure than the test data because in real applications, the review articles from less authorative web sites tend to be much more noisier. We are conservative in selecting training and testing instances in order to ensure the integrity of the data set. We list the statistics of the data set in Table 1.

Table 1: Statistics of the Data Set

# Reviews	Positive	Negative
Training	159,558	27,366
Testing	112,020	22,490

System Configurations In our evaluations, we vary the parameters of the classifiers and the features of n-grams. In particular, the configurations are:

1. PA classifier
   - Varying the number of n-gram features – Recall that we calculate χ^2 scores as the feature selection metric. We experiment with the top 50k, 100k and 200k n-grams with the highest χ^2 scores as features.
   - Varying the n of n-grams – We fix the number of features to 100k while varying the n of n-grams ($n = 1 \ldots 6$). Note that we denote n-grams as all n-grams with orders less than or equal to n.
2. LM classifier
   - LM – N-gram language models are trained on the whole review articles in the training data. Similar to the configuration of the PA classifier, we vary the value of n ($n = 3, 4, 6$) in language modeling.
   - LM-FILTER – This classifier uses the same language model as LM ($n = 6$), but it first tests on individual sentences in a review to filter out those possible objective sentences, *i.e.* the ratio between a sentence's positive and negative probabilities is below a threshold. After sentence filtering, the classifier predicts the polarity of the new review article that excludes those objective sentences.

3. Winnow classifier – We follow the configuration of the Winnow classifier as described in the previous section and employ n-grams ($n = 6$) as features.

Results and Discussions

We list the evaluation results in Table 2, which illustrates the performance of each classifier on positive/negative instances respectively, as well as their overall performance. We draw the following observations:

1. High order n-grams improve the performance of the classifiers, especially the performance on the negative instances. As Table 2 shows, using high order n-grams ($n = 3, 4, 5, 6$) as features for the PA classifier universally outperforms that using unigrams and bigrams. It is also true for language modeling. When using the 6-gram language model, the F_1 measure is augmented by 10.14% ($p - value < 0.01$ by paired t-test) than that using the trigram model. This observation verifies our hypothesis that given the enormous data on the Web, an online classification system should make use of high order n-grams as features, which are more deterministic, to achieve more precise results. This result is some contradictory to the conclusion drawn by Pang *et al.* (2002). We attribute it to that we put the sentiment classification problem in the context of the Web, and thus we are able to demonstrate the effectiveness of more complex linguistic features thanks to the large corpus.

   One may argue that the improvement seems not so significant in the overall F_1 measures for the PA classifier. This could be attributed to the fact that we make the test instances less positive or less negative, according to their ratings given by reviewers, than the training instances. This brings more difficulties to differentiate between test reviews, and thus inevitably narrows down the margin between the performance scores by various systems. Moreover, it is worth noting that our data set is very much skewed to positive instances as shown in Table 1. While it does not make obvious difference in the performance over the positive instances, we observe a significant increase of 10.72% ($p - value < 0.01$ by paired t-test) in F_1(Neg) for the negative instances when using n-grams ($n = 6$) over that using only unigrams.

2. Discriminative models are more appropriate for sentiment classification than generative models. The best F_1 measure obtained by the PA algorithm is significantly better than that by the language modeling method with a margin of over 0.04. It shows that since sentences with opposite polarities are often mixed together in the reviews, generative models are likely to be confused by the mixtures. In contrast, discriminating models mainly rely on the features on the boundaries. Moreover, the PA model is an online learning algorithm, which adjusts the separating hyperplane (or hypothesis) along with seeing more examples. As such, the PA model is less sensitive to the reviews which mix positive and negative sentences.

3. The performance of the PA classifier is not sensitive to the number of features, given that appropriate feature selection method is applied. We deem it imperative for a Web application to maintain low computational complexity in order to be deployed and updated efficiently. We observe that using the top 50k, 100k and 200k n-gram features selected by the χ^2 scores do not make obvious difference in the classification performance. It shows that we do not degrade the performance while significantly reducing the dimensionality of the feature vector.

4. Filtering out objective sentences does not show obvious advantage for our data set. As Pang and Lee (2004) suggested, removing objective sentences improves the classification of full reviews. However, in our evaluations, we see that incorporating a sentence subjectivity classifier with the language modeling classifier does not improve the performance, but decreases the F_1 measure by 1.9%. We conjecture that this is due to that we employ reviews about electronics while Pang and Lee worked on movie reviews. In movie reviews, there is large amount of description about movie plots. Therefore, it is helpful to incorporate a subjectivity classifier. However, it is not the case in electronics reviews, which are mainly subjective in terms of user comments on different attributes.

Conclusions and Future Work

In this paper, we presented the experiments we have done on sentiment classification using large-scale data set. Our experimental results show that a discriminating classifier combined with high order n-grams as features can achieve comparable, or better performance than that reported in academic papers. More importantly, this paper shows that sentiment classification is possible to be learned from online product reviews, even with very disparate products and authors. In addition, we have shown that high order n-grams do help in discriminating the articles' polarity in the mixture context. This observation based on large-scale data set has never been testified before.

In future work, a better feature selection scheme should be examined because there are a large amount of n-grams which are likely to be noisy. Previous work in classification has shown that feature selection is crucial to the success of classification task. Another promising extension of this work would be looking at classifying reviews to different scales, rather than just positive and negative. Instead of performing only multi-label classification, Pang and Lee (2005) addressed this problem by exploring the similarity among reviews by taking their positive sentences percentage (PSP) as a metric. They combine the use of SVM with nearest neighbors in determining scales of a review.

Acknowledgments

The authors are grateful to Tat-Seng Chua and Min-Yen Kan for their valuable discussions and comments on the paper. They would also like to thank Yoram Singer for his advice on classification algorithms.

References

Clarkson, P., and Rosenfeld, R. 1997. Statistical language modeling using the CMU–cambridge toolkit. In *Proc. Eurospeech '97*, 2707–2710.

Table 2: Performance comparison – P denotes precision and R denotes recall. F_1 is defined as $F_1 = \frac{2PR}{P+R}$. *Pos* stands for positive instances and *Neg* represents negative instances.

	P(Pos)	R(Pos)	F_1(Pos)	P(Neg)	R(Neg)	F_1(Neg)	P	R	F_1
PA-50k ngrams($n = 6$)	0.9329	0.9480	0.9404	0.7437	0.6536	0.6958	0.9013	0.8988	0.9000
PA-200k ngrams($n = 6$)	0.9352	0.9452	0.9402	0.7354	0.6672	0.6996	0.9018	0.8987	0.9003
PA-100k ngrams($n = 6$)	0.9350	0.9460	**0.9405**	0.7386	0.6657	**0.7003**	0.9022	0.8991	**0.9007**
$n = 5$	0.9348	0.9460	0.9404	0.7383	0.6651	0.6998	0.9019	0.8990	0.9004
$n = 4$	0.9350	0.9455	0.9402	0.7367	0.6659	0.6995	0.9018	0.8987	0.9003
$n = 3$	0.9347	0.9460	0.9403	0.7383	0.6640	0.6992	0.9018	0.8988	0.9003
$n = 2$	0.9360	0.9424	0.9392	0.7263	0.6722	0.6982	0.9009	0.8972	0.8990
$n = 1$	0.9176	0.9464	0.9317	0.7112	0.5696	0.6325	0.8830	0.8833	0.8832
LM ($n = 6$)	0.8766	0.9749	0.9231	0.7166	0.3162	0.4388	0.8499	0.8648	0.8573
LM ($n = 4$)	0.8774	0.9739	0.9231	0.7090	0.3186	0.4396	0.8492	0.8643	0.8567
LM ($n = 3$)	0.8384	0.9243	0.8793	0.3516	0.1872	0.2443	0.7570	0.8010	0.7784
LM-FILTER ($n = 6$)	0.8744	0.9631	0.9166	0.6152	0.2988	0.4022	0.8311	0.8520	0.8414
Winnow	0.8688	0.8021	0.8341	0.2867	0.3966	0.3328	0.7715	0.7343	0.7524

Dave, K.; Lawrence, S.; and Pennock, D. M. 2003. Mining the peanut gallery: Opinion extraction and semantic classification of product reviews. In *Proceedings of WWW-03, 12th International Conference on the World Wide Web*, 519–528. Budapest, HU: ACM Press.

Hatzivassiloglou, V., and McKeown, K. R. 1997. Predicting the semantic orientation of adjectives. In *Proceedings of the eighth conference on European chapter of the Association for Computational Linguistics*, 174–181. Morristown, NJ, USA: Association for Computational Linguistics.

Hu, M., and Liu, B. 2004. Mining and summarizing customer reviews. In *Proceedings of KDD '04, the ACM SIGKDD international conference on Knowledge discovery and data mining*, 168–177. Seattle, US: ACM Press.

Manning, C. D., and Schütze, H. 1999. *Foundations of Statistical Natural Language Processing*. Cambridge (Massachusetts) and London (England): The MIT Press.

Mullen, T., and Collier, N. 2004. Sentiment analysis using support vector machines with diverse information sources. In Lin, D., and Wu, D., eds., *Proceedings of EMNLP 2004*, 412–418. Barcelona, Spain: Association for Computational Linguistics.

Nigam, K., and Hurst, M. 2004. Towards a robust metric of opinion. In *Proceedings of the AAAI Spring Symposium on Exploring Attitude and Affect in Text: Theories and Applications*.

Pang, B., and Lee, L. 2004. A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In *Proceedings of ACL-04, 42nd Meeting of the Association for Computational Linguistics*, 271–278. Barcelona, ES: Association for Computational Linguistics.

Pang, B., and Lee, L. 2005. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In *Proceedings of ACL-05, 43nd Meeting of the Association for Computational Linguistics*. Ann Harbor, US: Association for Computational Linguistics.

Pang, B.; Lee, L.; and Vaithyanathan, S. 2002. Thumbs up? sentiment classification using machine learning techniques. In *Proceedings of EMNLP-02, the Conference on Empirical Methods in Natural Language Processing*, 79–86. Philadelphia, US: Association for Computational Linguistics.

Popescu, A.-M., and Etzioni, O. 2005. Extracting product features and opinions from reviews. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, 339–346. Vancouver, British Columbia, Canada: Association for Computational Linguistics.

Shalev-Shwartz, S.; Crammer, K.; Dekel, O.; and Singer, Y. 2004. Online passive-aggressive algorithms. In Thrun, S.; Saul, L.; and Schölkopf, B., eds., *Advances in Neural Information Processing Systems 16*. Cambridge, MA: MIT Press.

Turney, P. D., and Littman, M. L. 2003. Measuring praise and criticism: Inference of semantic orientation from association. *ACM Trans. Inf. Syst.* 21(4):315–346.

Wilson, T.; Wiebe, J.; and Hoffmann, P. 2005. Recognizing contextual polarity in phrase-level sentiment analysis. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, 347–354. Vancouver, British Columbia, Canada: Association for Computational Linguistics.

Yang, Y., and Pedersen, J. O. 1997. A comparative study on feature selection in text categorization. In *Proc. 14th International Conference on Machine Learning*, 412–420.