Learning to Identify Review Spam

Fangtao Li, Minlie Huang, Yi Yang and Xiaoyan Zhu
State Key Laboratory of Intelligent Technology and Systems
Tsinghua National Laboratory for Information Science and Technology
Dept. of Computer Science and Technology, Tsinghua University, Beijing, China
{fangtao06, yangyiycc}@gmail.com; {aihuang, zxy-dcs}@tsinghua.edu.cn

Abstract

In the past few years, sentiment analysis and opinion mining becomes a popular and important task. These studies all assume that their opinion resources are real and trustful. However, they may encounter the faked opinion or opinion spam problem. In this paper, we study this issue in the context of our product review mining system. On product review site, people may write faked reviews, called review spam, to promote their products, or defame their competitors' products. It is important to identify and filter out the review spam. Previous work only focuses on some heuristic rules, such as helpfulness voting, or rating deviation, which limits the performance of this task.

In this paper, we exploit machine learning methods to identify review spam. Toward the end, we manually build a spam collection from our crawled reviews. We first analyze the effect of various features in spam identification. We also observe that the review spammer consistently writes spam. This provides us another view to identify review spam: we can identify if the author of the review is spammer. Based on this observation, we provide a two-view semi-supervised method, co-training, to exploit the large amount of unlabeled data. The experiment results show that our proposed method is effective. Our designed machine learning methods achieve significant improvements in comparison to the heuristic baselines.

1 Introduction

With the development of the Internet, people are more likely to express their views and opinions on the Web. They can write reviews or other opinions on E-Commerce sites, forums, and blogs. These opinion information is important for individual users. Currently, it is a common and typical behavior to read the reviews or comments before purchasing some products or services. The opinion information also benefits the business organizations. They can monitor the consumers' expressions, and effectively adjust their production and marketing strategies. Hence, in recent years, sentiment analysis and opinion mining has become a popular topic for the researchers of artificial intelligence (AI).

The researchers from AI community have developed various sentiment analysis tasks, including sentiment classification [Li *et al.*, 2010; Wu and Huberman, 2010], opinion retrieval [Furuse *et al.*, 2007], opinion extraction [Qiu *et al.*, 2009], to serve users' need. All of the above studies have the same assumption: their opinion resources are real and trustful. However, in practice, this opinion information may be faked. Since the opinion information can guide the people's purchasing behavior, and on the web, any people can write any opinion text, this can let the people give undeserving positive opinions to some target objects in order to promote the objects, and/or give unjust or malicious negative opinions to some other objects in order to damage their reputations. These faked opinion information is called opinion spam [Jindal and Liu, 2008]

The opinion spam identification task has great impacts on industrial and academia communities. For sentiment analysis companies, if the opinion provided services contain large number of spams, they will affect the users' experience. Furthermore, if the user is cheated by the provided opinion, he will never use the system again. For academic researchers, they have conducted various research studies on sentiment analysis tasks. If their acquired opinion resources contain many opinion spams, it is meaningless to provide any sentiment analysis results. Therefore, it is an essential task to identify and filter out the opinion spam.

In this paper, we study the opinion spam task in the context of our product review mining system. In our crawled review site, some people, called spammer, may write faked reviews, called review spam, to promote their products, and/or defame their competitors' products. We need to identify and filter out the review spam to provide the real and trustful review services. Previous methods only employ some heuristic rules: some one use the helpfulness voting, which is from other people's helpfulness evaluation on the posted review; others use rating deviation rules, which means that if the review rating varies much from the average product rating, this review can be considered as spam.

In this paper, we exploit machine learning methods to identify review spam. We first describe the influence of different features in the supervised learning framework. Since the annotation is tedious, the number of the annotated reviews

is small. We also design a semi-supervised method to utilize the large number of unlabeled reviews. We observe that the review spammer consistently writes spam. This provide us another view to identify review spam: we can identify if the author of the review is spammer. Based on this observation, the two-view method, co-training algorithm, is used as our semi-supervised method. The experiment results show that our proposed method is effective. Our designed machine learning methods achieve significant improvements than the heuristic baselines.

2 System Description

In this section, we introduce our product review mining system, which aims to help the consumers to easily mine the needed information from large amount of reviews. The overall framework is shown in Figure 1. We first crawl the review pages from the review site, then we parse these html pages, with several regular expressions, to extract the review-relevant text parts. Before the review analyzer module, we need to identify and filter out the faked reviews, called review spam, to provide the consumer real and trustful reviews. The review analyzer mainly predicts the overall sentiment and extract the topic and opinion words for each review. We then index all the analyzed reviews into the indexer. We provide several applications. First, the consumer can target their product with product search. With consumer's constraint, the system also can recommend products based on the product reviews. After targeting the product, the system directs the user into the product pages. This page provides the review summarization for this product based on the review topic and opinion extraction. Since the number of reviews can be very large, especially for the popular products, we also provide the review search module. The consumer can search the reviews with sentiment queries, such as "positive opinions on battery" for a camera. We also provide a product comparison module. The consumer can compare the overall products, and product attributes based on the posted reviews.

Review spam identification is an important component in our system. We need to provide real and trustful review mining results. If our review collection contains many faked reviews, the user may be cheated, and never trust the services. Previous methods only employ some heuristic methods to identify review spam. In this paper, we introduce our review spam identification component based on machine learning algorithms.

3 Review Spam Corpus Construction

To identify review spam, we manually build a review spam corpus. We use a part of our crawled product reviews, which are obtained from Epinions. The data set consists of about 60k reviews. On the review sites, after the review is posted, other users can evaluate the posted review. They can provide a score to denote if this review is helpful, or write comments for the reviews. There is a large amount of helpfulness evaluations and comments in the review sites. Some reviews can draw dozens, or hundreds of helpfulness evaluations and comments. We will annotate the review spam by taking advantages of these evaluations.

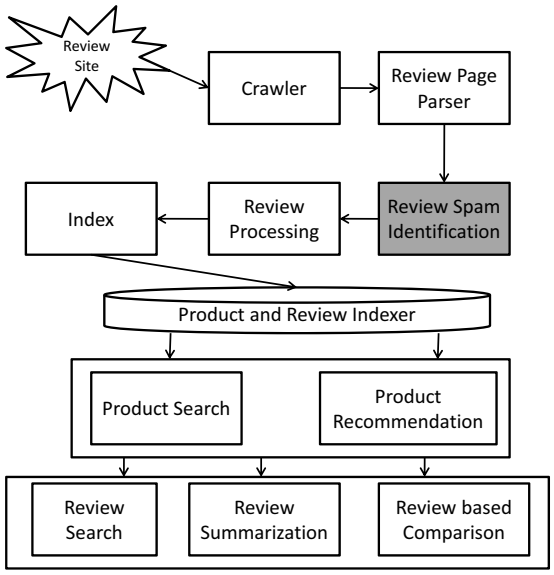

Figure 1: The Framework of Product Review Mining System

It is impossible to annotate all the reviews. However, simply randomly sampling the reviews could lead us to a small number of spam, which will compromise the creation of the effective training and testing data sets for our analysis. We assume the relation between review helpfulness and spam is as follows: the review spam will not help the people know the product. Therefore, The low-helpful reviews contains more review spam. We need to manually annotate more low-helpful reviews.

Based on this assumption, we create our spam data set as follows:

Data Pre-processing:

1. The duplicate products are removed based on full name match. The reviews with anonymous reviewers are also removed. After that, we select the reviews with enough social evaluations, whose number of helpfulness and comment evaluation is above 5.

2. We rank all the left reviews (about 30k) , based on their overall helpfulness, and divide them into three sets: top-helpful set, middle-helpful set, low-helpful set. We randomly select 1000, 1000, and 4000 reviews from the three sets separately.

3. We extract various contexts for the review to be annotated, including all the comments and helpfulness evaluations, the target product description, and the reviewer's profile and history reviews.

Annotator Training: In this part, we present a principled way to conduct annotation. In public blogs and forums, there is of great interest in identifying review spam. A recent study[1] introduces 30 ways to identify the review spam. We can find more other related articles and discussions in the Internet. We employ 10 college students to annotate the review

[1]http://consumerist.com/2010/04/how-you-spot-fake-online-reviews.html

spam data set. They are first asked to read all the above related articles and discussions to know what the review spam looks like. They then independently label the review data. When they determine if the review is spam or not, they are also asked to carefully read the contexts provided in the data pre-processing step. Each review is labeled by two people. The conflict is resolved by the third one.

Finally, in the 6000 reviews, we get 1398 spam reviews. The result also verifies our hypothesis: most of the spam reviews are from the low-helpful review set. We identify 1256, 112, 30 spam reviews from the low, middle and top helpful set separately.

4 Review Spam Identification

4.1 Methods

Supervised Methods

With our labeled review spam data set, we can design fully supervised method to identify review spam. We test several supervised methods, including SVM, logistic regression, Naive Bayes, based on the public machine learning software Weka [Hall *et al.*, 2009]. We find that Naive Bayes achieves best results in our experiments. In this section, we only briefly introduce the Naive Bayes Classifier. Naive Bayes assumes the features are conditionally independent given the review's category.

$$P_{NB}(c|d) = \frac{P(c)\prod_{i=0}^{m} P(f_i|c)}{(P(d))} \quad (1)$$

Despite its simplicity and the fact that its conditional independence assumption doesn't hold in real-world situations, Naive Bayes-based categorization still tends to perform surprisingly well [Lewis, 1998].

Semi-Supervised Method

Since it is a labor intensive task to manually label the review spam, we only annotate a small set of review data. There are still a large number of unlabeled data, which may boost the performance. In this section, we want to use the semi-supervised method to utilize the unlabeled reviews.

Before designing our semi-supervised method, we observe that the spammers consistently write review spam. To verify this observation, we randomly select 40 spammers from our labeled data set by removing the reviewers with low number (below 3) of reviews. For each spammer, we randomly extract 10 his reviews. We manually label these reviews, which aims to check if the spammer consistently writes review spam. Among 40 spammers, 25 always write spams, 3 write about 80% spams, 6 write about 70% spams, 4 write about 40% spams, 2 write about 30% spams. On average, the spammers have about 85% possibility to write spam. This can bring us two views to identify the review spams: the first view is to directly detect if the review is review spam; the other view is to detect if the author of the review is spammer. If the author of the review is a spammer, this review have very high probability to be a review spam.

Based on the above observations, we design a two-view semi-supervised method for review spam detection. We employ the framework of the co-training algorithm. The co-training algorithm [Blum and Mitchell, 1998] is a typical bootstrapping method, which starts with a set of labeled data, and increases the amount of annotated data by adding unlabeled data incrementally. One important aspect of co-training algorithm is property of two views. The separation of two views proves to be more effective than the single view in practice and theory [Blum and Mitchell, 1998; Wan, 2009]. In the context of review spam identification, each review has two views of features: features about review itself and features about corresponding reviewers. The overall framework is shown in Algorithm 1.

In practice, there are always noises in the data. The assumptions of co-training, such as conditional independent views, may not hold in practice. Following previous work [Collins and Singer, 1999], which used "agreement" strategy between the two view classifiers, we also design a variant of co-training. We only select the p positive instances and n negative instances, when the two view classifiers agree most: $T \cup T^{'}$ in Step 8 is changed to $T \cap T^{'}$.

Algorithm 1 Co-Training Algorithm

Require: two views of feature sets for each review: review features F_r and reviewer features F_u; a small set of labeled reviews L; a large set of unlabeled reviews U.

Ensure: Loop for I iterations

1: Learn the first view classifier C_r from L based on review features F_r;
2: Use C_r to label reviews from U based on F_r;
3: Choose p positive and n negative most confidently predicted reviews T_{review} from U.
4: Learn the second view classifier C_u from L based on reviewer features F_u;
5: Use C_u to label reviews from U based on reviewer features F_u;
6: Choose p positive and n negative most confidently predicted reviews $T^{'}_{reviewer}$ from U.
7: Extract the reviews $T^{'}_{review}$ authored by $T^{'}_{reviewer}$
8: Move Reviews $T_{review} \cup T^{'}_{review}$ from U to L with their predicted labels.

4.2 Features

The feature engineering is a key task for review spam identification task. We have acquired various observations to identify review spam by analyzing our data set and reading discussions from public blogs and forums. But how to transfer these observations to the features is still a challenging task. In this section, we introduce our extracted features for review spam identification. We mainly divide the features into two groups. One is related with review, the other is related with reviewer.

Review Related Features

This type of features contains four groups: content features, sentiment features, product features and meta data features.

Content Features

unigram and bigram.

We use feature selection metric χ^2 to select the text classification features: the top 100 unigrams and top 100 bigrams.

Square of normalized length

A length of the review, normalized by the maximum length, is also extracted as a real number feature.

First Person vs. Second Person

We find that in the faked review, it sometimes says "you" should do something, rather than how "I" experienced. We use the ratio of the first personal pronouns, such as "I", "my", "we", and the second personal pronouns, such as "you", "your", as a real number feature.

High Similarity Score

The spammer may just change the product name in the review, or post the same review on more than one products. We represent each review as a word vector, and select the highest cosine similarity score with other reviews as a real number feature.

we also extract other content features as follows: ratio of the question and exclamation sentences, where these sentences are identified simply by regular expressions; ratio of the capital letters.

Sentiment Features

Subjective vs. Objective

If the review consists of much objective information, it may just describe the products' attributes or off topic advertisements. We compute the ratio of subjective and objective at the word and sentence level. The subjective word is identified by subjective lexicons, SentiWordNet and HowNet. If the sentence contains at least one subjective word, it is considered as subjective.

Positive v.s. Negative

If the review only express positive sentiment or negative sentiment on the product, it tends to be spam. Because the real reviews will express both sides of sentiments. We compute the ratio of positive and negative text at the word and sentence level. The positive and negative sentiment are also identified by sentiment lexicons.

Product Features

Product Centric Features

We employ the number of reviews under this product to denote the popularity of the product. We also use the average rating of the product as a feature.

Product Description Features

It is a good indicator that how the product is described in the review. If the product name is not mentioned, this review may be an off-topic advertisement. If the brand name or product name is mentioned many times, this review may be an advocator for this product. We compute the percent of brand and product name in all words as a real number feature.

Meta-data Features

The meta-data features include the rating of the reviews. We also compute the difference between the review rating and the average rating of the target product. The post time is also considered. We use a binary feature to denote if the review is the first product review.

Reviewer related Features

All the reviewer related features are divided into two groups: Profile Features and Behavior Features.

Profile Features

The profile features are all extracted from the reviewer profile page. It contains the reviewer id, the number of written reviews, whether contains real name, homepage, and self-descriptions, the rank of popularity in the whole site and the specific category.

Behavior Features

Authority Score

On Epinions site, one reviewer can "trust" another reviewer, if the former thinks the reviews written by the latter is trustful. This is similar to the web page links. A directed reviewer graph is first constructed based on the "trust" relation. We compute the reviewer's authority score based on the link analysis algorithm $PageRank$:

$$PR(u_i) = \frac{1-d}{N} + d \sum_{u_j \in M(u_i)} \frac{PR(u_j)}{L(u_j)} \quad (2)$$

where $u_1, \ldots, u_N$ are the reviewers in the collection, N is the total number of reviewers, $M(u_i)$ is the set of reviewers that "trust" reviewer u_i, $L(u_i)$ is the number of reviewers that reviewer u_i "trust", d is a damping factor, which is set as 0.85. The $PageRank$ score can be computed iteratively with random initial values. We use $PageRank$ score as the authority score. We also try to employ the number of in-degree "trust" and out-degree "trust" as authority score features.

Brand Deviation Score

The spammer may focus on specific brands, we compute the distribution of the review numbers over different brands. We use entropy to denote this score:

$$H(X) = -\sum_{i=1}^{n} p(x_i) \log p(x_i) \quad (3)$$

where x_i is the ith brand, $p(x_i)$ is the probability with the number of the ith brand reviews divided by the total reviews.

Rating Deviation Score

The spammer may give different brands with different ratings. We compute the variance as a real number features:

$$Var(X) = \sum_{i=1}^{n} p(x_i)(s(x_i) - \mu)^2 \quad (4)$$

where $s(x_i)$ is the average rating for the ith brand, μ is the overall average rating on all the brands.

5 Experiments

5.1 Experiment Setup

The data has been described in Section 3. For our supervised methods, we need to divide the data set into training set and test set. We conduct 10-fold cross-validation: the data set is randomly split into ten folds, where nine folds are selected for training and the tenth fold is selected for test. We apply our co-training method on the same test data set as the supervised methods, for the convenience of comparison.

The evaluation metrics are $precision$, $recall$ and F-score: $precision = \frac{S_p \cap S_c}{S_p}$, $recall = \frac{S_p \cap S_c}{S_c}$, $F = \frac{2*precision*recall}{precision+recall}$, where S_c is the set of true review spams, S_p is the set of predicted review spams.

We also design several heuristic methods to identify the spam reviews. The simplest one is to select reviews randomly as spam, which we denote as RANDOM. The second is based

on rating deviation, where the review, which has high difference from average rating of the target product, is considered as spam. The third one is based on helpfulness evaluation, where the review, which has lower helpfulness evaluation, is considered as spam. We try several thresholds for rating difference and helpfulness evaluation, and select the best threshold based on the training data set.

5.2 Experiment Results

Supervised Method Results

Table 1 shows the experiment results. The machine learning method Naive Bayes (NB) achieves significant improvement compared with the heuristic methods. With all features, NB can achieve the best result 0.583 in F-Score. We also analyze the influence of different features. We exclude each feature from all features (A) to see the performance change. We can see that when we exclude the metadata feature (A-metadata) or the behavior feature (A-behavior), the performance drops most, which shows the importance of these two features. When we exclude all review related features (A-review), all reviewer related features can get 0.545 in F-Score. All review related features (A-reviewer) can get 0.550 in F-Score.

	Precision	Recall	F-Score
Random	0.233	0.233	0.233
Variation	0.347	0.371	0.359
Helpful	0.184	0.911	0.306
All Features(A)	0.517	0.669	**0.583**
A-content	0.502	0.662	0.571
A-sentiment	0.507	0.649	0.569
A-product	0.514	0.665	0.580
A-metadata	0.506	0.632	0.562
A-profile	0.516	0.658	0.578
A-behavior	0.541	0.587	0.563
A-review	0.593	0.504	0.545
A-reviewer	0.571	0.531	0.550

Table 1: Results with Different Features. "-" denotes to "exclude" the corresponding feature

Semi-supervised Method Results

From the previous section, we have analyzed the effect of various features. In this section, we exploit co-training method to utilize the large number of unlabeled data. Table 2 shows the experiment results. NB-Bootstrapping is a bootstrapping version of NB, which uses all features as a single view. Co-training is the original method in Algorithm 1, which separately predict unlabeled data with two views. Co-Training (Agreement) selects the unlabeled data with the most agreement for two view classifiers. We can see that co-training is suitable for this task. It achieves better result than NB-Bootstrapping. With the agreement strategy, the method can achieve the best results for review spam identification.

Parameter Sensitivity

In this section, we exploit the parameter sensitivity. Figure 2 shows the results for different iteration numbers. We can find that when the iteration numbers are above 40, it can achieve good results. p and n are the numbers of newly added positive and negative samples in each iteration. Since the number of negative samples are larger than positive samples (1398 v.s. 4602, about 1:3), we add more negative samples in each iteration. From Figure 3, we can see that when $p = 15$, $n = 45$, it can achieve best results in our data set.

	Precision.	Recall.	F-Score
NB	0.517	0.669	0.583
NB-Bootstrapping	0.621	0.575	0.597
Co-Training	0.630	0.589	0.609
Co-Training(Agreement)	0.641	0.621	**0.631**

Table 2: Results on Semi-Supervised Methods.

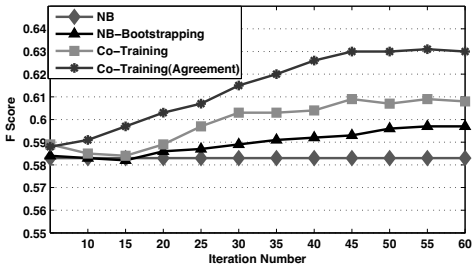

Figure 2: Evaluations on Different iteration numbers

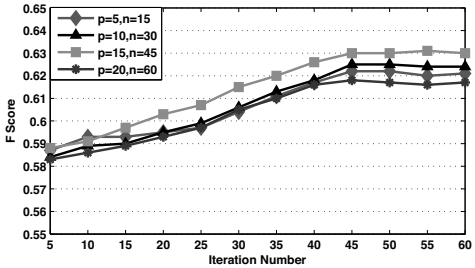

Figure 3: Evaluations on Different p, n.

6 Related Work

In the past few years, sentiment analysis and opinion mining becomes an important and popular task. Various research topics and applications are conducted in the research communities, see the surveys ([Pang and Lee, 2008; Liu, 2010]). Few of these studies are aware of the review spam problem. A preliminary research on Amazon reviews is reported in [Jindal and Liu, 2008]. They re-framed the review spam identification problem as duplicated reviews identification problem. However, their assumption that duplicated review is spam, is not appropriate [Pang and Lee, 2008]. First, repeated reviews constitute some sort of manipulation attempt. Because Amazon itself cross-posts reviews across "different" products, where "different" includes different instantiations or

subsequent editions of the same item in different categories. Specifically, in a sample of over 1 million Amazon book reviews, about one-third were duplicates, but these were all due to Amazon's cross-posting. Second, Human Operation errors (e.g., accidentally hitting the "submit" button twice) cause the repeated reviews. In our paper, we manually build a review spam corpus with the help of its contexts. Lim et al. [Lim *et al.*, 2010] propose to use the user behavior as features to predict spam user, without using any textual features. In this paper, besides the user related features, we also employ various review based features to identify review spams.

There are a lot of research papers on review quality prediction [Kim *et al.*, 2006; Liu *et al.*, 2007]. Review spam is different from the low quality review. Low quality review may be due to poor writing. But this low quality review is still real and trustful. While the review spam is faked in order to promote his products or defame his competitors' products. Review spam and low quality review have different characteristics. Our experiments also show that directly identifying review spam with helpfulness evaluation can't achieve satisfactory results. In this paper, we exploit machine learning algorithms with various extracted features for review spam identification.

7 Conclusions

In this paper, we study the review spam identification task in our product review mining system. We manually build a review spam collection based on our crawled reviews. We first employ supervised learning methods and analyze the effect of different features in review spam identification. We also observe that the spammer consistently writes spam. This provides us another view to identify review spam: we can identify if the author of the review is spammer. Based on the observation, we provide a two-view semi-supervised methods to exploit the large amount of unlabeled data. The experiment results show that the two-view co-training algorithms can achieve better results than the single-view algorithm. Our designed machine learning methods achieve significant improvements as compared with the heuristic baselines.

In future work, we plan to exploit the probabilistic two-view algorithm, such as Co-EM, to model the uncertainty in review spam identification task. We also plan to test our co-training algorithm in other opinion resources, such as blog, or twitter.

Acknowledgments

This work is supported by Canada's IDRC Research Chair in Information Technology program, Project Number: 104519-006, the Chinese Natural Science Foundation Grants No. 60973104 and 60803075, China 973 Project No. 2007CB311003 and China Core High-Tech Project No. 2011ZX01042-001-002. We thank Yunbo Cao and the anonymous reviewers for their helpful comments. The first author also thanks the support of Google PhD Fellowship.

References

[Blum and Mitchell, 1998] Avrim Blum and Tom Mitchell. Combining labeled and unlabeled data with co-training. In *COLT*, pages 92–100, New York, NY, USA, 1998. ACM.

[Collins and Singer, 1999] Michael Collins and Yoram Singer. Unsupervised models for named entity classification. In *EMNLP-VLC*, pages 100–110, 1999.

[Furuse *et al.*, 2007] Osamu Furuse, Nobuaki Hiroshima, Setsuo Yamada, and Ryoji Kataoka. Opinion sentence search engine on open-domain blog. In *IJCAI*, pages 2760–2765, CA, USA, 2007. Morgan Kaufmann Inc.

[Hall *et al.*, 2009] M. Hall, E. Frank, G. Holmes, B. Pfahringer, P. Reutemann, and I.H. Witten. The WEKA data mining software: an update. *SIGKDD Explorations*, 11(1):10–18, 2009.

[Jindal and Liu, 2008] Nitin Jindal and Bing Liu. Opinion spam and analysis. In *WSDM*, pages 219–230, New York, NY, USA, 2008. ACM.

[Kim *et al.*, 2006] Soo-Min Kim, Patrick Pantel, Tim Chklovski, and Marco Pennacchiotti. Automatically assessing review helpfulness. In *EMNLP*, pages 423–430, Morristown, NJ, USA, 2006. ACL.

[Lewis, 1998] David D. Lewis. Naive (Bayes) at forty: The independence assumption in information retrieval. In C. Nédellec and C. Rouveirol, editors, *ECML*, number 1398, pages 4–15, Chemnitz, DE, 1998. Springer Verlag.

[Li *et al.*, 2010] Fangtao Li, Minlie Huang, and Xiaoyan Zhu. Sentiment analysis with global topics and local dependency. In *AAAI*, 2010.

[Lim *et al.*, 2010] Ee-Peng Lim, Viet-An Nguyen, Nitin Jindal, Bing Liu, and Hady Wirawan Lauw. Detecting product review spammers using rating behaviors. In *CIKM*, pages 939–948, New York, NY, USA, 2010. ACM.

[Liu *et al.*, 2007] Jingjing Liu, Yunbo Cao, Chin-Yew Lin, Yalou Huang, and Ming Zhou. Low-quality product review detection in opinion summarization. In *Proceedings of EMNLP-CoNLL*, pages 334–342, 2007. Poster paper.

[Liu, 2010] Bing Liu. Sentiment analysis and subjectivity. In N. Indurkhya and F. Damerau, editors, *Handbook of Natural Language Processing, Second Edition*. Taylor and Francis Group, Boca Raton, FL, 2010.

[Pang and Lee, 2008] Bo Pang and Lillian Lee. Opinion mining and sentiment analysis. *Found. Trends Inf. Retr.*, 2:1–135, January 2008.

[Qiu *et al.*, 2009] Guang Qiu, Bing Liu, Jiajun Bu, and Chun Chen. Expanding domain sentiment lexicon through double propagation. In *IJCAI*, pages 1199–1204, CA, USA, 2009. Morgan Kaufmann Inc.

[Wan, 2009] Xiaojun Wan. Co-training for cross-lingual sentiment classification. In *ACL-IJCNLP*, ACL '09, pages 235–243, Stroudsburg, PA, USA, 2009. ACL.

[Wu and Huberman, 2010] Fang Wu and Bernardo A. Huberman. Opinion formation under costly expression. *ACM Trans. Intell. Syst. Technol.*, 1:5:1–5:13, October 2010.