ICIC Express Letters
Volume **10**, Number **9**, September **2016**

 ISSN 1881-803X
pp. **2269–2275**

DETECTING DECEPTIVE REVIEW SPAMMERS BASED ON PSEUDO-SUPERVISED CLASSIFICATION

Zhuo Wang, Xiangnan Zhao and Tingting Hou

School of Information Science and Engineering
Shenyang Ligong University
No. 6, Nanping Center Road, Hunnan District, Shenyang 110159, P. R. China
syluwz@foxmail.com

Received February 2016; accepted May 2016

Abstract. *Online product reviews have become an important resource for making purchase decisions. Driven by financial incentives, some online product reviewers deliberately post fake reviews to promote their products or defame their competitors' products. Lacking gold-standard labeled datasets for model training or testing, traditional supervised learning algorithms often fail to detect such deceptive reviewers. In this paper, we propose a pseudo-supervised learning approach to identifying review spammers based on improved review graph computing. Our method is completely unsupervised which does not require any manual labeling of data. Experimental study shows that our method outperforms previous ranking-based method in both precision and recall.*
Keywords: Review spammer, Product review graph, Classification, Opinion mining

1. **Introduction.** Nowadays people are getting used to buying products or services under the guidance of online product reviews. For the sake of profits, there exist a large number of review spammers who deliberately post fake reviews to promote or demote products. Review spammers are becoming a real threat to online rating systems. To address this problem, researchers proposed different kinds of approaches to detect spam reviews or review spammers [1, 2, 4, 5]. Unlike Email spam or Web page spam, review spam detection is even more difficult due to the lack of gold-standard labeled datasets for model building or evaluation. Jindal and Liu [1] exploited machine learning approaches to detect fake reviews or reviewers. They used duplicate or near duplicate reviews as spam reviews, and extracted spamming behavior features to train classifiers. In [2], a gold-standard dataset was created by hiring Amazon Mechanical Turks to write deceptive reviews for chosen hotels, while the truthful reviews were mined from real online hotels. With this dataset, detect deceptive review tasks are issued by human judges, genre identification, psycholinguistic method and text categorization, respectively. Shojaee et al. [3] exploited lexical and syntactic features extracted from review content to train classifiers so that deceptive reviews can be detected. Such methods are very limited in that 1) the precision is relatively low due to the quality of labeled datasets, and 2) it is laborious and error-prone in labeling spam reviews and/or review spammers. Therefore, it is well established that traditional supervised learning is inferior to unsupervised approaches in review spam detection area.

Ranking approaches are widely adopted in review spammer detection. Lim et al. [4] proposed scoring methods to measure the degree of spamicity of a reviewer based on the reviewer's rating behaviors. Wang et al. [6] first proposed a heterogeneous review graph model to reveal the intricate relationships among reviews, reviewers and online stores, which can detect subtle fake store reviewers. Wang et al. [7] proposed a similar product review graph structure which deals with the store-less shopping environment. Since the number of products is significantly larger than the number of stores, [7] proposed algorithm

ICE, which can shorten the convergence time by eliminating a small portion of trustable reviewers and their reviews during each iteration. More features of spamming activities extracted from reviews, reviewers and products are incorporated into the computation of node reputation, which can significantly improve the detection precision.

However, the method in [7] suffers the following problems. 1) As the method is based on ranking, the detection precision decreases as the number of top reviewers increases. 2) The method in [7] computes the reviewer trustiness and review honesty simultaneously. According to our observation, there exist a large number of reviews whose honesty scores are very low, while the authors of these reviews are not in the front of the suspicious reviewer list. Meanwhile, there are also some top suspicious reviewers who almost have no reviews in the top suspicious review list. This phenomenon indicates that the detection precision needs to be further improved and there are also many more suspicious reviewers that need to be further recalled.

In this paper, we propose a heuristic pseudo-supervised learning approach to detecting deceptive suspicious review spammers by coupling the ranking approach and supervised learning approaches. We give a refined product review graph model which can dramatically improve the precision in ranking review spammers. First, we use *RICE* (refined *ICE*) to find top N review spammers as spam reviewers, and then deliberately choose N' genuine reviewers as non-spam reviewers, so that traditional supervised learning methods can be applied to classifying other suspicious reviewers. Human evaluations show that our method not only can improve the detection precision, but also can improve the recall in comparison to the method in [7].

2. Refined Product Review Graph Model.

2.1. Product review graph.

To further improve the detection precision of the ranking method in [7], we give a refined data model of product review graph. Our refined product review graph model also comprises three kinds of nodes: the reviewer node, the review node, and the product node. Each review node is connected to a reviewer node and a product node if and only if the reviewer posts at least one review toward the product. The product review graph reveals the intricate relationship among reviewers, reviews, and products. Like [7], each node in the graph is assigned to a reputation score of being spamming. That is, the honesty of a review v, denoted by $H(v)$, is a score representing how honest the review is; the trustiness of reviewer r, denoted by $T(r)$, is a score of how much we can trust the reviewer; the reliability of product p, denoted by $R(p)$, is a score indicating the quality of the product. Each kind of nodes is attached to a set of features indicating the spamicity of the node, denoted as $f_1, f_2, \ldots, f_n$. Each node's reputation score is affected by other kinds of nodes related to that node. With these assumptions, the reputation scores of nodes in the graph are iteratively updated until all the reputation scores become stationary.

Unlike [7], which limits all reputation scores and feature values to $[-1, 1]$, we limit all these values to $[0, 1]$ to avoid the counteraction of scores from different nodes. The smaller the reputation score or the feature values of a node, the more spamming the node becomes. In accordance with these variations, we also modify the scoring functions for each kind of nodes, and use different methods to compute the feature values of each kind of nodes.

2.2. Node features.

For review-related features, we used review content similarity (RCS), and rating deviation (RD). For product-related features, we used average rating (AR), and total number of reviews (TNR). These features are the same as those used in [7]. For reviewer-related features, we used reviewer's review content similarity (RRCS), rating score variance (RSV), early review (ER), multiple rated products (MRP), reviewer active

duration (RAD), and review date entropy (RDE). The RRCS, RAD and RDE are calculated in the same way as in [7], while we supplemented three new features: ER, RSV and MRP, which are defined below. The ARS (average rating score) and RFR (ratio of first reviews) features used in [7] are removed according to our feature analysis result.

1. Early review (ER)
   Spammers intend to post reviews early to influence the upcoming reviewers. Given a reviewer r, we define this feature as:
$$ER(r) = \max_{p \in P_r} ERP(r,p) \tag{1}$$
$$ERP(r,p) = \begin{cases} (L(r,p) - A(p))/\beta, & L(r,p) - A(p) \leq \beta \\ 1, & L(r,p) - A(p) > \beta \end{cases} \tag{2}$$
   where P_r is the product set reviewed by reviewer r, $L(r,p)$ is the latest date when r reviews product p, $A(p)$ is the date of the first review of p, and β is a time threshold which was set to 180 days in our experiments.
2. Rating score variance (RSV)
   Spammers intend to rate the similar scores. Let S_r^2 denote the rating variance of reviewer r, and we use logistic function to restrict RSV to $[0,1)$:
$$RSV(r) = \frac{2}{1 + e^{-S_r^2}} - 1 \tag{3}$$
3. Multiple rated products (MRP): Spammers often post many reviews for the same product to promote or demote that product [4]. Let V_r^p denote the review set reviewed by reviewer r toward product p, and we define:
$$MRP(r) = 1 - \frac{|\{p | p \in P_r, |V_r^p| \geq 2\}|}{|P_r|} \tag{4}$$

2.3. **Scoring functions.** How to compute the reputation scores of all the nodes in the review graph is crucial to the final ranking of reviewers. Here we give the formulae to compute the trustiness of reviewers, the honesty of reviews, and the reliability of products.

2.3.1. *Trustiness of reviewers.* Intuitively, a reviewer r's trustiness is determined by the honesty of all the reviews he posts. In [7], the summation of all the honesty scores of reviews by reviewer r, denoted as H_r is calculated, i.e., $H_r = \sum_{v \in V_r} H(v)$, where $H(v)$ is the *Honesty* of review v which will be defined later, and V_r is the review set by reviewer r. Then the logistic function is exploited to smooth H_r to range $(-1,1)$:
$$T'(r) = \frac{2}{1 + e^{-H_r}} - 1 \tag{5}$$

Unlike [7], since our review honesty scores are within $[0,1]$, we just use the average value of all the review honesty scores of reviewer r to represent the trustiness of reviewer r. That is, we use:
$$T'(r) = avg_{v \in V_r} H(v) \tag{6}$$

To further take into account the reviewer's features, we also supplement a second part of the scoring function as a penalty item using the feature values of that reviewer:
$$T(r) = \alpha T'(r) + (1 - \alpha) \sum_{i=1}^{m} k_i f_i \tag{7}$$

where f_i is the i-th feature (value) of reviewer r, k_i is the co-efficient of f_i, $\sum_{i=1}^{m} k_i = 1$, and α is a weight to control the importance of each part of the equation. k_i and α are empirical parameters specified by user. In our experiments, we use the entropy weight [8] of feature f_i as k_i, and set $\alpha = 0.8$ (the same below).

2.3.2. *Honesty of reviews.* A review's honesty is related to the reviewer who writes it. Moreover, the honesty of a review can also be inferred by its *surrounding* reviews. In [7], the surrounding reviews of review v is divided into the *similar* review set (S_v) and *dissimilar* review set (DS_v) in a predefined time window Δt.

Let R_s be the reviewer set of all the reviewers who write the reviews in S_v, and R_{ds} be the reviewer set of all the reviewers who write the reviews in DS_v, define $A(v, \Delta t)$ as follows:

$$A(v, \Delta t) = \sum_{r \in R_s, r \neq r_v} T(r) - \sum_{r' \in R_{ds}, r' \neq r_v} T(r') \tag{8}$$

where r_v is the author of review v. This reveals that the honesty of the review becomes larger if the summation of the trustiness scores of the reviewers in R_s gets larger, or if the summation of the trustiness scores of the reviewers in R_{ds} gets smaller. The final scoring function of the honesty of review v:

$$H(v) = \alpha T(r_v) R(p_v) \frac{1}{1 + e^{-A(v, \Delta t)}} + (1 - \alpha) \sum_{i=1}^{n} k_i f_i \tag{9}$$

where p_v is the product of review v, f_i is the i-th feature of review v, $\sum_{i=1}^{n} k_i = 1$. Equation (9) implies that the honesty of a review is dependant on the trustiness of its author, and the reliability of the product being reviewed.

2.3.3. *Reliability of products.* The reliability of a product relies on the reviews issued by trustworthy reviewers rather than those by untrustworthy reviewers. We model the summation of reliability of product p as:

$$R_p = \sum_{v \in V_p, T(r_v) > 0.5} T(r_v)(\Psi_v - \mu)$$

where Ψ_v is the rating score of review v, and μ is the median of the rating scores, which can be set to 3 in a 5-star rating system. We use logistic function to smooth R_p:

$$R'(p) = \frac{1}{1 + e^{-R_p}} \tag{10}$$

Equation (10) implies that higher rating scores by trustworthy reviewers lead to more reliable products. Again, we take into account the features of product p and get the final scoring function as:

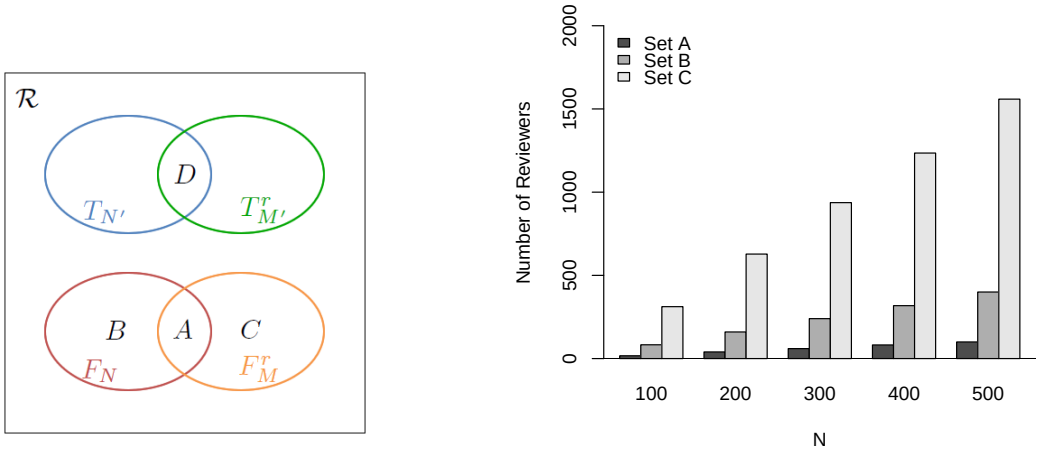
$$R(p) = \alpha R'(p) + (1 - \alpha) \sum_{i=1}^{s} k_i f_i \tag{11}$$

where k_i is the i-th feature of product p, and k_i is its coefficient, $\sum_{i=1}^{s} k_i = 1$.

2.4. **The *RICE* algorithm.** Algorithm *ICE* in [7], which computes the node reputation scores, can also be applied to our refined product review graph model. We refer to the refined algorithm as *RICE* (refined iterative computation with elimination).

3. Pseudo-Supervised Classification.

3.1. **Motivation.** As *ICE* or *RICE* computes the honesty of reviews and the trustiness of reviewers simultaneously, one can conjecture that the honesty scores of the reviews written by the reviewers with lower trustiness scores would also be lower with a high probability. However, we observed a large number of reviews in the top fake reviews (reviews with low honesty scores) whose corresponding reviewers are not in the top fake reviewer list. On the other hand, there are also many top spammers who have no reviews in the top fake review list. Let F_N denote the top N fake reviewer set, F_M denote the top



(a) Relationships among Set A, B, C, D

(b) Number of reviewers in Set A, B, C w.r.t N

FIGURE 1. Reviewer set construction

M fake review set, $T_{N'}$ denote the top N' true reviewer set, and $T_{M'}$ denote the top M' true review set, we can get four typical reviewer sets:

$$A = \{r | r \in F_N; \exists v \in V_r, v \in F_M\}$$
$$B = \{r | r \in F_N; \forall v \in V_r, v \notin F_M\}$$
$$C = \{r | v \in F_M, r_v \notin F_N\}$$
$$D = \{r | r \in T_{N'}; \exists v \in V_r, v \in T_{M'}; \forall v \in V_r, v \notin F_M\}$$

Let $F^r_M = \{r | v \in F_M, r = r_v\}$, $T^r_{M'} = \{r | v \in T_{M'}, r = r_v\}$. Figure 1(a) illustrates the relationships among the four sets. Obviously, we have:

$$A = F_N \cap F^r_M$$
$$B = F_N - A$$
$$C = F^r_M - A$$
$$D = T_{N'} \cap T^r_{M'}$$

where A is a set of reviewers with the highest probability to be spammers, D is a set of reviewers with the highest probability to be genuine reviewers, B and C are suspicious fake reviewer sets. We plot the number of reviewers in Set A, B, C with respect to N, as shown in Figure 1(b) (M is set to the total number of reviews of reviewers in F_N automatically). We can see that C is very large, which indicates that there are a large number of reviewers whose reviews are in F_M, but they are not in F_N. Meanwhile, there are also many reviewers in F_N but none of their reviews are in F_M (Set B). This reveals that reviewers in B and C are abnormal and we should further investigate their spamicity.

To this end, we heuristically establish a dataset by taking Set A as labeled spammers, and Set D as labeled non-spammers. Using this dataset as training dataset, we can categorize the reviewers in Set B and C into spammers and non-spammers using machine learning algorithms such as K-means, SVM, KNN, and Naive Bayes. We call such kind of machine learning method as *pseudo-supervised learning* since the *labeled data* are obtained automatically rather than by human judges.

3.2. **Classifying fake reviewer candidates.**

3.2.1. *K-Means clustering.* Note that K-Means is an unsupervised classification which does not rely on labeled data. However, it makes no sense if we directly do K-Means over Set B and C to cluster them into two clusters: the spammers and the non-spammers.

This is because reviewers in B and C are all suspicious spammers and thus essentially belong to the same cluster. To categorize Set B and C, we use Set A and D as the spammer representatives and non-spammer representatives, respectively. Then we do K-Means clustering on $A \cup B \cup C \cup D$, so that reviewers in B and C are classified into spammers and non-spammers based on their similarities to A and D, respectively.

3.2.2. *Other classifiers.* For other supervised learning-based classifiers, such as SVM, KNN, and Naive Bayes, we use $A \cup D$ as training set, and do classification over $B \cup C$, so that members in $B \cup C$ are predicted as spammers or non-spammers accordingly.

4. **Experimental Study.**

4.1. **Datasets.** We use Amazon review dataset crawled in 2006, which was also used in [1, 4, 9, 10] for review spammer detection. As the whole dataset is extremely large, we only extracted the book review data from the dataset. Reviewers who have less than 3 reviews were removed because such reviewers do not have enough clues to calculate their trustiness scores. Such kind of reviews was tackled in [11]. The final book dataset contains 148,566 reviewers, 1,440,576 reviews, and 478,684 products.

4.2. **The effectiveness of *RICE*.** To evaluate the effectiveness of *RICE*, we compared *RICE* with *ICE* by human evaluation. Our human evaluators are three postgraduate students who are very familiar with E-commerce environments and product review characteristics. Due to the immense work for human evaluation, we only fetch the top 200 suspicious reviewers detected by *RICE* and *ICE*, respectively. The 400 reviewers are randomly shuffled so that the evaluators do not know from which algorithm the reviewer comes. We regard a reviewer as a spammer if at least 2 evaluators mark the reviewer as a spammer. Table 1 shows the evaluation result. From the table we can see that the precision of top N spammers returned by *RICE* is extremely high, which is 100% when $N = 100$ and 99.5% when $N = 200$, far better than the precisions by *ICE*, which are 91% and 90.5%, respectively. This indicates that our refined computing model is more effective than the model in [7], which guarantees that Set A can be used as the positive instances (spammers) for classifier training.

Table 1. Human evaluation result

N	*ICE*		*RICE*	
	Number of spammers	Precision	Number of spammers	Precision
100	91	91%	100	100%
200	181	90.5%	199	99.5%

Table 2. Machine learning results on Set B and C

N	Classifier	Set B		Set C	
		spammer	non-spammer	spammer	non-spammer
200	bayes	159	1	104	524
	nnet	158	2	46	582
	logistic	159	1	90	538
	knn	156	4	43	585
	ksvm	159	1	72	556
	kmeans	160	0	30	598

4.3. **Evaluation of pseudo-supervised classification.** In Section 3, we discussed how to utilize machine learning methods to postprocess the result sets returned by *RICE*. We used R packages which implement K-Means (kmeans), SVM (ksvm), Naive Bayes (bayes), Neural networks (nnet), Logistic regression (logistic), and K-nearest neighbor (knn) to learn the classifiers. In each machine learning algorithm, we only use the reviewer features described in Subsection 2.2. Table 2 shows the learning results of all the classifiers doing pseudo-classification upon Set B and C for $N = 200$ (for $N = 500$, the results are similar thus omitted). We can see that all classifiers classify almost all the reviewers in Set B as spammers (which also demonstrates the effectiveness of *RICE*), while a large number of reviewers in Set C are classified as spammers, which can be taken as the potential spammers that need to be further investigated. Of all the 6 classifiers, K-Means intends to take all reviewers in Set B as spammers, and take less reviewers in Set C as spammers due to its clustering nature, showing quite different behaviors from other classifiers.

To evaluate the effectiveness of our proposed pseudo-supervised classification approach, we manually examined the detected suspicious spammers for $N = 200$. These suspicious reviewers include 40 reviewers in Set A, 160 reviewers in Set B, and 130 reviewers from Set C who are identified as spammers by at least 3 classifiers. As we have already evaluated in Subsection 4.2, reviewers in Set A are all taken as spammers, and in Set B, 159 reviewers are taken as spammers by our evaluators. From Table 2 we can see that the classifiers are all very accurate, which also indicates the effectiveness of our refined *RICE* algorithm. For Set C, 41 out of 130 reviewers are marked as spammers by evaluators. Although the precision for Set C is not very high, these spammers are unable to be detected by *RICE*.

5. **Conclusions and Future Work.** We propose a refined product review graph model, and give a completely unsupervised classification methodology for online product review spammer detection. Our proposed *RICE* algorithm outperforms *ICE* significantly in precision, so that it can generate high quality labeled spammers for classifier training. Our pseudo-supervised classification improves both the precision and the recall. Experiments and human evaluations verify the effectiveness of our methodology.

REFERENCES

[1] N. Jindal and B. Liu, Opinion spam and analysis, *Proc. of WSDM'08*, pp.219-230, 2008.

[2] M. Ott, Y. Choi, C. Cardie and J. T. Hancock, Finding deceptive opinion spam by any stretch of the imagination, *Proc. of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies – Volume 1*, Stroudsburg, PA, USA, pp.309-319, 2011.

[3] S. Shojaee, M. Murad, A. Azman, N. Sharef and S. Nadali, Detecting deceptive reviews using lexical and syntactic features, *Proc. of ISDA'13*, Salangor, Malaysia, pp.53-58, 2013.

[4] E.-P. Lim, V.-A. Nguyen, N. Jindal, B. Liu and H. W. Lauw, Detecting product review spammers using rating behaviors, *Proc. of CIKM'10*, New York, NY, USA, pp.939-948, 2010.

[5] F. Li, M. Huang, Y. Yang and X. Zhu, Learning to identify review spam, *Proc. of IJCAI*, pp.2488-2493, 2011.

[6] G. Wang, S. Xie, B. Liu and P. S. Yu, Review graph based online store review spammer detection, *Proc. of ICDM*, pp.1242-1247, 2011.

[7] Z. Wang, T. Hou, Z. Li and D. Song, Spotting fake reviewers using product review graph, *Journal of Computational Information Systems*, vol.11, no.16, pp.5759-5767, 2015.

[8] L. Jing, M. K. Ng and J. Z. Huang, An entropy weighting k-means algorithm for subspace clustering of high-dimensional sparse data, *IEEE Trans. Knowledge and Data Engineering*, vol.19, no.8, pp.1026-1041, 2007.

[9] N. Jindal, B. Liu and E.-P. Lim, Finding unusual review patterns using unexpected rules, *Proc. of CIKM'10*, New York, NY, USA, pp.1549-1552, 2010.

[10] A. Mukherjee, B. Liu and N. Glance, Spotting fake reviewer groups in consumer reviews, *Proc. of WWW'12*, New York, NY, USA, pp.191-200, 2012.

[11] S. Xie, G. Wang, S. Lin and P. S. Yu, Review spam detection via time series pattern discovery, *Proc. of WWW'12 Companion*, New York, NY, USA, pp.635-636, 2012.