Uncovering Collusive Spammers in Chinese Review Websites

Chang Xu[†], Jie Zhang[†], Kuiyu Chang[†], Chong Long[‡]
[†]School of Computer Engineering, Nanyang Technological University, Singapore
[‡]A9.com at Amazon, Beijing, China
[†]xuch0007@e.ntu.edu.sg, zhangj@ntu.edu.sg, kuiyu.chang@pmail.ntu.edu.sg
[‡]longc@amazon.com

ABSTRACT

As the rapid development of China's e-commerce in recent years and the underlying evolution of adversarial spamming tactics, more sophisticated spamming activities may carry out in Chinese review websites. Empirical analysis, on recently crawled product reviews from a popular Chinese e-commerce website, reveals the failure of many state-of-the-art spam indicators on detecting collusive spammers. Two novel methods are then proposed: 1) a KNN-based method that considers the pairwise similarity of two reviewers based on their group-level relational information and selects k most similar reviewers for voting; 2) a more general graph-based classification method that jointly classifies a set of reviewers based on their pairwise transaction correlations. Experimental results show that both our methods promisingly outperform the indicator-only classifiers in various settings.

Categories and Subject Descriptors

H.3.3 [**Information Search and Retrieval**]: Information Filtering

Keywords

Opinion Spam; Collusive Spammer; Spam Review Detection

1. INTRODUCTION

With the growing availability of review services at online stores (e.g. amazon.com) and opinion sharing websites (e.g. epinions.com), consumer-generated reviews become an indispensable part of online shopping; nowadays online shoppers will not purchase a product without reading the reviews. Unfortunately, many of the reviews they read may not be that genuine as expected. It has been found that some paid professionals fabricate reviews without even using the products [2] or consuming the services [1], with the sole goal of promoting the reputation of their employers or demoting the competitors.

CIKM'13, Oct. 27–Nov. 1, 2013, San Francisco, CA, USA.

http://dx.doi.org/10.1145/2505515.2505700.

Existing review spam tackling approaches focus on detecting spam reviews, spammers and spammer groups in popular review websites. The general way is to extract engineered strong indicators from review contents or reviewer behaviors which are then used for modeling and learning. Jindal and Liu [2] use supervised learning to detect three types of spam reviews: untruthful reviews, reviews on brands only, and non-reviews. Li *et al.* [3] identify spam reviews via semi-supervised learning from the views of reviews and reviewers. Ott *et al.* [8] detect deceptive reviews based on review text categorization by combining linguistic features of reviews and features borrowed from studies in psychology. Lim *et al.* [4] detect spammers via a scoring model, ranking reviewers based on their behavioral patterns. Mukherjee *et al.* [6] study spammer groups. Reviewer-group based features are proposed to capture aggregated behavioral patterns of spammer groups. However, little research has been done to solve this problem in Chinese review websites. The situation is critical in China, given the great rich-poor divide and the massive Internet population. China's Internet users have hit 564 million by the end of 2012 according to China Internet Network Information Center (CNNIC). 37.1% of the entire users are students and unemployed, who are the ideal source for low-cost information diffusion on the Internet. People join together (aka. the "Internet Water Army") to post spam reviews for only 0.10 to 0.50 RMB per posting. We argue that more sophisticated spam campaigns are expected to be spotted due to the sheer size and the organizational prowess of the "Internet Water Army" in China. In fact, we have found in a popular Chinese review website that spammers have evolved to operate in small scales with well-coordinated attacks, where small groups of spammers collude to generate a desired result without showing up on the radar of site moderators.

In this paper, we detect collusive spammers (colluders) in Chinese review websites. We focus on colluders because 1) paid spammers in China are more likely to be well organized and collaborate on tasks coordinated by a shady organizer, and 2) compared to "individual" versions, colluders may exert full control over the opinions of the compromised targets, thereby undermining the trustworthiness of the review websites. Particularly, 1) We empirically analyze 25 state-of-the-art spam indicators based on our recently crawled Chinese review dataset. Anomalies are spotted not only in the languages the spammers use but also in the behaviors they act, causing the ineffectiveness of many spam indicators when used in classification; 2) As we have observed that reviewers within "similar" reviewer groups [6] are more likely

to have similar class labels, a KNN-based method is proposed to detect colluders more effectively, in which the similarity of two reviewers is computed based on the similarities of their corresponding groups, then k most similar reviewers are selected for the final voting; 3) Finally, a more general graph-based classification method is proposed by capturing the *transaction correlations* among reviewers where the correlation is formed once a pair of reviewers have both reviewed at least one product within a predefined time interval. Compared to spam indicators, transaction correlations are much harder to fake once the transactions are made. A novel colluder graph model based on *pairwise Markov network* is introduced to capture the transaction correlations, and an approximate inference algorithm based on the iterative classification algorithm (ICA) [7] is designed where a reviewer's class label can be collectively determined not only by his own intrinsic attributes but also by the class labels of the neighborhood. To the best of our knowledge, it is the first time the graph model based method has been used to detect colluders in online review websites. The experimental results show that both our methods promisingly outperform the indicator-only classifiers in various settings.

2. DATA ACQUISITION

We created a colluder dataset by crawling consumer reviews from Amazon.cn (the Chinese counterpart of Amazon.com): a snapshot of manufacturing product reviews till August 20, 2012. It contains 1,205,125 reviews written by 645,072 reviewers on 136,785 products (e.g., electronics, housewares). Each review has 6 attributes: ReviewerID, ProductID, Product Brand, Rating, Date and Review Text. We selected Amazon.cn because: 1) Popularity: it is one of the most popular e-commerce websites in China; 2) Abundance and Variety: the huge number of consumer reviews cover a wide range of products; 3) Comparability: many existing studies such as [4, 1, 12] used datasets from Amazon.com for evaluation. Thus we believe that the data from Amazon.cn, which shares similar scheme with Amazon.com, should be comparable to previous studies.

2.1 Colluder Sampling

To create a dataset that holds sufficient colluders for evaluation, the first task is to search for the places where colluders would probably be found. A good way to achieve this is to use frequent itemset mining (FIM), similar with [6]. In such context, reviewer IDs are regarded as *items*, each transaction is the set of reviewer IDs who have reviewed a particular product. Through FIM, groups of reviewers who have reviewed multiple common products can be found. Here we use maximal frequent itemset mining (MFIM) to discover groups with maximal size since we focus on the worst spamming activities in our dataset. Following the same parameter settings as [6], 8,915 groups are found, each represented as a mapping from a set of members (≥ 2) to a set of commonly reviewed products (≥ 3). Finally, we merge all the members of each group into a collection V which consists of 5,055 reviewers in all.

2.2 Annotation

Each reviewer in V should be annotated as either colluder or non-colluder. It is acknowledged that spammers who express fake opinions in review websites are hard to identify due to the lack of ground truth. Fortunately we noticed that Amazon.cn is practically clearing up the displayed reviews periodically. We also confirmed that typically a displayed review will be deleted if it has been regarded as spam or if its content has been confirmed to be irrelevant to the specific product, by the website moderators. Thus it would save great annotation efforts by locating those deleted reviews in our dataset. We then re-crawl all the reviews of each reviewer in V on March 25th, 2013, finding that 1,822 out of 5,055 reviewers have at least one of their posted reviews being deleted by Amazon.cn, during the near seven-month period[1]. However, not all of the spam reviews have been removed from the system, which forces us to manually examine the remaining reviewers. With the help of the practical spam review identification guidelines from previous studies [12, 6], online discussions and our own observations, as a result, a total of 3,118 non-colluders and 1,937 colluders are identified. We argue that nowadays for many popular review websites like Yelp and Amazon, it is common to hire anti-spam groups to control spam over the sites. They typically hold information that would be more useful for spam review detection (e.g., IP Information of reviewers) which is not publicly available. Nevertheless, for those sites that do not consistently clean up spam reviews, the practical spam signals presented in aforementioned sources may facilitate the annotation process to a great extent.

3. SPOTTING ANOMALIES: EMPIRICAL ANALYSIS OF SPAM INDICATORS

Effective spam indicators are crucial to the performance of detecting colluders through modeling or learning. Prior to our work, existing studies have proposed different spam indicators in different perspectives. In this section, we perform empirical analysis of these indicators on our Chinese review dataset from a *colluder-centric* view, i.e., the spam indicators are categorized into three dimensions - linguistic indicators, individual behavioral indicators and collusive behavioral indicators - according to a colluder's potential reviewing activities on the review sites. Spotted anomalies are presented together with the evaluations of each indicator. Cumulative histograms (CHs) are used to show the distributions of the indicator scores over colluders and non-colluders respectively, which provides a visual intuition for how well each indicator discriminates colluders. The larger the gap between the curves of two distributions exhibits, the better the discrimination capability will be achieved. Note that although in practice indicators may not be used individually, our individual evaluation can give a more fine-grained perspective of how the feasibility of *each* indicator would be affected by the changes of the spamming patterns in real world data, guided by which the potential correlations among the indicators can be further assessed in practice. All the indicator scores are normalized to [0,1].

3.1 Linguistic Indicators

Spammers are considered to express opinions in a different way with regular reviewers. We evaluate 10 linguistic indicators (LIs) used in [8, 2, 3]: for the text of each review 1) Unigrams (after Chinese word segmentation); 2) POS (part-of-speech) tags frequencies; 3) Positive sentiment word frequency (PWF); 4) Negative sentiment word frequency (NWF); 5) Subjective word frequency (SWF); 6)

[1]In Amazon.cn, reviewers themselves cannot delete reviews.

Cosine similarity of the review and product descriptions (SRD); 7) Brand name mentioned frequency (BNF); 8) First person word frequency (FPF); 9) Second person word frequency (SPF); 10) Squared average length (SAL). Reviews in Chinese characters are different from those in English words, where the text is written without any spaces between words. *Word segmentation* thus is used to split Chinese sentences into a sequence of meaningful words. We also remove stopwords and punctuations from the review text, and obtain 100 unigrams by using feature selection metric χ_2. POS tagging is performed based on the unigrams. Positive/Negative sentiment words and subjective words are identified by HowNet. For each reviewer we aggregate all his reviews into one collection and represent it with these linguistic indicators. The CHs of each indicator (exclude Unigrams and POS) are shown in Fig. 1.

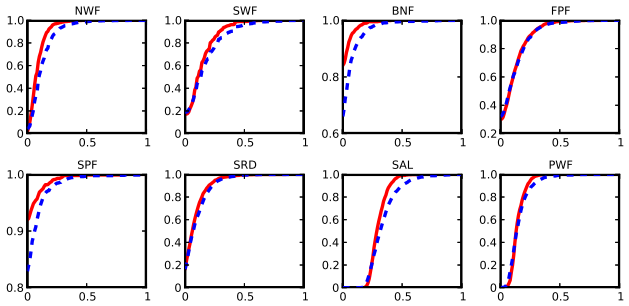

Figure 1: Distributions of colluders (solid) and non-colluders (dashed) vs. LI scores.

As shown, none of the cumulative histograms of the examined linguistic indicators show very clear gaps between the curves. This gives us a hint that nowadays in Chinese review websites, colluders generally express in a similar way with ordinary reviewers, or we can say that they behave *normally* in the linguistic level. In fact, this is not that surprising because writing reviews differently (e.g., more passionate) would not bring any benefit: 1) website moderators may find these reviews suspicious and then do some cleanups; 2) readers may find a review untrustworthy if it contains too many bombastic words, then choose to ignore it.

It is worth noting that in BNF, the curve of non-colluders lies closer to the left axis, suggesting that non-colluders mention brand names more often than colluders. [2] considers the reviews that *only* comment on the brands of specific products as spam because they believe this kind of reviews is often biased. While in our case, we find it very likely that regular reviewers often mention brand names just for reference. Not a few reviewers buy products greatly relying on the reputation of specific brands; it is thus natural to comment on brands or just use brands as pronouns. In contract, colluders write reviews in a more general fashion, they even do not mention product names in their reviews in practice.

In SAL we can see that non-colluders generally post longer reviews than colluders. Many of non-colluders are identified, through our further investigation, as active reviewers who work hard on their comments. The average squared length of the text of the reviews written by colluders is 4.03, lower than the first quartile of that over all the reviews in our dataset, which is 4.12. Colluders often write reviews with moderate length probably because they are in a hurry to move on to the next task for pursuing more profits.

3.2 Individual Behavioral Indicators

Individual behavioral indicators (IBIs) were used effectively in [4, 3]. Only the ones applicable to our dataset are selected: 1) Review Count (RC); 2) Brand Deviation Score (BDS); 3) Rating Deviation Score (RDS). 4) Targeting Products (TP); 5) Targeting Product Groups (TPG); 6) General Deviation (GD); 7) Early Deviation (ED). For each reviewer, BDS measures the deviation in his review counts over different brands, while RDS measures the variance of his ratings over different brands. TP evaluates for a particular product how similar all his reviews are in terms of ratings and contents. TPG measures the pattern of ratings towards a set of products sharing common attributes (e.g., brand) within a short time interval. GD measures the average difference between the reviewer rating on one product versus the product's average rating, and ED measures how early the ratings of a product deviate from the average rating of that product. TP is meaningless in our dataset since only a small portion of reviewers write multiple reviews for the same products. Thus the CHs of the remaining six IBIs are shown in Fig. 2.

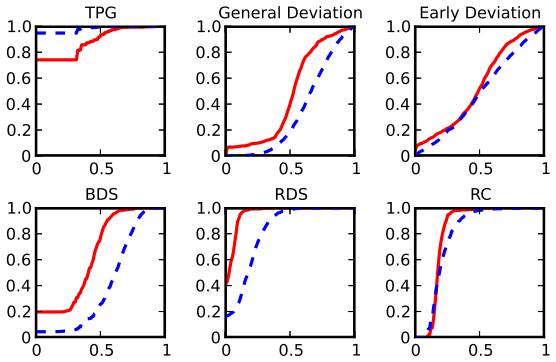

Figure 2: Distributions of colluders (solid) and non-colluders (dashed) vs. IBI scores.

In the figure, TPG, BDS and RDS exhibit larger gaps than other three. Specifically, colluders have a much higher TPG score than non-colluders at any cumulative percentage, indicating that colluders tend to review products of the same brand within a short time period (e.g., 1 day). Also, the colluder curves for BDS and RDS lie closer to the left axis, meaning that colluders exhibit less variations in the number of reviewed brands and ratings given to each brand. Colluders seem to target a limited number of brands, and rate all the products under the same brand similarly (e.g., consistently high ratings of 4 or 5 stars). Despite the fairly clear separation in GD, the GD scores of colluders are generally lower, indicating that their ratings are more consistent with the product average ratings, which contradicts the expectation that colluders are the ones often give outlier ratings. This is possible if the compromised products have been overwhelmed by colluders who all give similar ratings. We also find that most of the compromised products are the less popular ones, thus ratings from a small portion of genuine reviewers would have little impact on the average ratings. In ED we can hardly separate the curves. In fact, as confirmed by GETF in Section 3.3, colluders typically post reviews early. ED performs poorly because the rating deviation of colluders is marginal (with low GD scores). Thus the low rating deviation scores would effectively neutralize the higher weights attached to the early reviews of colluders, re-

sulting in similar ED scores with non-colluders. In terms of RC, colluders seem not to post too many reviews, probably because they balance their multiple accounts by dispersing all the reviews.

3.3 Collusive Behavioral Indicators

Since colluders collaboratively post spam reviews on many products, their collusive behaviors may sell them out, thus implying the potency of the collusive behavioral indicators. We select all the 8 collusive behavioral indicators (CBIs) from [6]: 1) Group Time Window (GTW); 2) Group Deviation (GD); 3) Group Content Similarity (GCS); 4) Group Member Content Similarity (GMCS); 5) Group Early Time Frame (GETF); 6) Group Size Ratio (GSR); 7) Group Size (GS); 8) Group Support Count (GSUP). GTW evaluates how close, temporally, the members of a group write reviews for a particular product. GD flags a group as suspicious if its ratings diverge significantly from other reviewers for a particular product. GCS and GMCS calculate the maximum average cosine similarities among inter-member reviews and among individual member's reviews, respectively. GETF reveals how early a group post reviews for common products. GSR measures the averaged ratio of the group size to the actual reviewer number of each common product. GSUP records the number of the common products reviewed by a group. However, these indicators are originally group-based while our task is reviewer-based. We thus first generate the group-based scores for each reviewer group (see Section 2.1), then each reviewer will inherit the group-based score from the corresponding group. For those involved in multiple groups the highest score will be chosen, as we intend to capture the worst spamming behaviors of colluders. Fig. 3 shows the CHs of the CBIs.

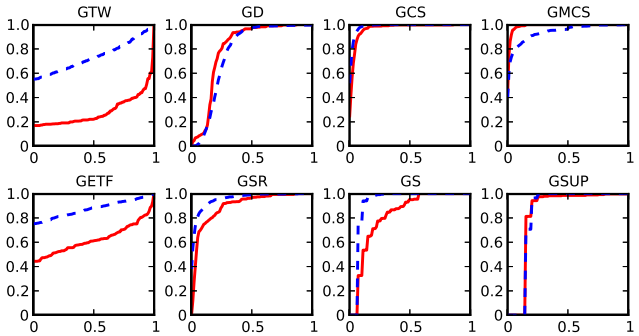

Figure 3: Distributions of colluders (solid) and non-colluders (dashed) vs. CBI scores.

As shown, GTW, GETF and GSR also perform well in our case. GS shows a clear gap when cumulative percentage ≥ 0.3. It is not surprising that the sizes of colluder groups are typically larger. However, we also notice the overlap appearing at the beginning of both curves in GS, indicating that quite a few colluder groups have similarly small sizes with benign ones which may form at random. GCS and GMCS show interesting results about the review contents. In GCS, neither the result of colluders nor non-colluders is more statistically significant than each other. In GMCS even colluders achieve lower scores than non-colluders, implying that nowadays colluders tend not to copy from neither their own prior reviews nor those of the accomplices. In fact we find that many specifications of online published spamming tasks explicitly state that copying is not allowed during the whole campaigns, so we speculate that spammers may not get paid if they simply cut-and-paste prior reviews. GD fails for the same reason as General Deviation (Section 3.2). GSUP performs the worst, implying that colluders are not likely to work together on too many common products in Chinese review websites.

3.4 Indicators Comparison and Discussion

We obtain AUC scores for each indicator (Fig. 4) using a Logistic Regression classifier. The AUC ranking is roughly consistent with the situation illustrated in the cumulative histograms. Linguistic indicators generally underperform in this simple detection setting, which has also been observed in [6]. In terms of the combinations, IBI+CBI performs the best which is in fact the combination of all behavioral indicators. It seems that for a spammer it is easier to comment just as a regular reviewer, however, it is much harder to fake or even change the spamming actions which in most cases are driven by the specifications or tactics of the spam campaigns. We further investigate why those eight behavioral spam indicators (General Deviation, ED, RC, Group Deviation, GCS, GMCS, GS, GSUP) perform poorly.

In fact, when computing CBI scores for each group we observed that 63.4% colluders belong to more than one group while only 25.5% non-colluders meet this criteria. For illustration, we randomly pick a colluder who is the member of 13 groups and make the following observations: (o1) *Low GSUP*: 12 out of the 13 (92.3%) have only 3 common products while the remaining one has 4; (o2) *Low GS*: 5 have size 2, 3 have size 3, 4 have size 4, and 1 has size 5; (o3) *Low brand variations*: all 13 groups target products of one particular brand; (o4) *Similar ratings*: among all 124 product-rating pairs, 21 (16.9%) are 4 stars, and 103 (83.1%) are 5 stars. There are no ratings lesser than 4 stars; (o5) *Overlapping colluders*: among all 78 group-group pairs, 69 (88.5%) have one common member, 8 have two, and one pair even has three common members; (o6) *Overlapping products*: among all 78 group-group pairs, 57 (73.1%) have at least one common product. Note that these groups have very low GSUP.

To better understand how these tiny groups cooperate with each other, we draw a bipartite graph to represent the relationship among groups, colluders and products. Fig. 5 shows an example, in which all 4 groups of colluders share a common member "Z4". C_1 and C_2 both review products "1C" and "88" while C_3 and C_4 both review products "7Y" and "1C". Such an arrangement suffices to evade the capture of those defective behavioral indicators:

- Groups C_1 and C_2 (likewise C_3 and C_4) would have been merged into one bigger group if both groups write reviews for at least three common products (see Section 2.1). Now that they are single groups, GS will fail to distinguish them from normal ones (randomly formed groups);
- The GSUP of these groups are also in the same scale as normal ones which may coincidentally review a few common products;
- From (o4) we know that groups that target products of the same brand would give consistently high/low ratings. If a less popular product is overwhelmed by many such groups, the majority rule followed by ED, General Deviation and Group Deviation would become invalid since the ratings given by colluders will be close to the average ratings of the compromised products.

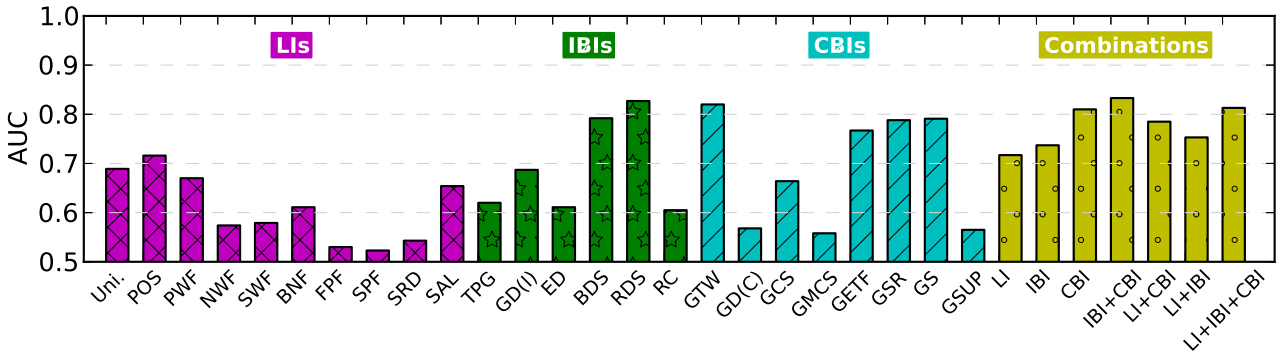

Figure 4: Areas under ROC curves (AUC) for all the 24 spam indicators (exclude TP) and their combinations. Larger AUC denotes better detection trade-off between true and false positives based on the corresponding indicator. GD(I) as General Deviation, GD(C) as Group Deviation.

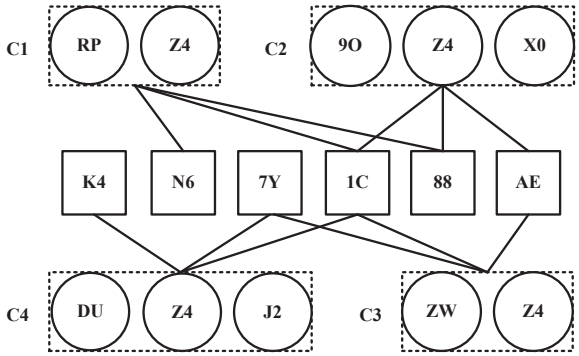

Figure 5: 4 groups (dashed boxes) of colluders (circles) collaboratively review six products (squares).

4. DETECTING COLLUDERS

Through the above analysis, we are not quite confident about the stability of the spam indicators when detecting colluders in Chinese review websites. Colluders may learn and change their tactics and eventually adapt to evade existing anti-spam techniques that equip with those indicators. In this section, we attempt to solve this problem from another perspective that may bypass the potential defects of the spam indicators.

4.1 The KNN-based Method

According to the observations in Section 3.4 suspicious groups are found to be highly similar with each other in terms of overlapped members and reviewed products, and similar ratings. This is because given a dense reviewer-product bipartite graph, by using the FIM algorithm with fixed parameter settings, the whole graph would be decomposed into many small pieces of fully connected sub-graphs (groups) that may have many overlapped nodes (members and products). Such small sub-graphs may dilute the effectiveness of some indicators as discussed in Section 3.4. However, being used properly, these tiny groups may be favorable to detect colluders in a novel way. In this section we propose a KNN-based method to detect colluders by utilizing the *similarities* between such groups. Let $\{c_j\}_{j=1}^{m}$ be a set of groups and $\{v_i\}_{i=1}^{n}$ be a set of reviewers with each associated with an vector $\mathbf{a}_i$ of attributes (instantiated as specific spam indicators in our case). Note that each reviewer may belong to multiple groups. By modeling the colluder detection problem as a binary classification problem, our goal is to assign each reviewer v_i with a class label $l_i \in \{pos, neg\}$[2]. The idea is that given a set of groups, the reviewers who belong to "similar" groups may be more likely to have the same class labels. Thus the class label of a reviewer v_i can be determined commonly (e.g., via voting) by a set of k reviewers who belong to groups most "similar" to the groups v_i belongs to. Now, we begin by measuring the pairwise similarity of two groups which consists of three components as follows:

Common Member Ratio measures the *Jaccard similarity* of the sets of members of two groups:

$$s_{\text{cmr}} = \frac{\#(M_i \cap M_j)}{\#(M_i \cup M_j)} \tag{1}$$

where M_i and M_j are the member sets of groups c_i and c_j.

Common Product Ratio is computed as the sum of the number of products of the same brands reviewed by each group, divided by the sum of the number of products reviewed by each group:

$$s_{\text{cpr}} = \max_{b \in B} \frac{\#(P_{b,i}) + \#(P_{b,j})}{\#(P_i) + \#(P_j)} \tag{2}$$

where B is the set of common brands reviewed by both groups c_i and c_j. $P_{b,i}$ ($P_{b,j}$) is the set of the products with brand b reviewed by group c_i (c_j), and P_i (P_j) is the set of the products reviewed by group c_i (c_j).

Common Brand Rating Deviation computes the deviation between the average ratings given to the products of common brands reviewed by two groups, by using the normalized inverted RMSD (Root Mean Square Deviation):

$$s_{\text{cbrd}} = \frac{1}{1 + \sqrt{\frac{1}{|B|}\sum_{b \in B}(\overline{r}_{b,i} - \overline{r}_{b,j})^2}} \tag{3}$$

where $\overline{r}_{b,i}$ ($\overline{r}_{b,j}$) is the average rating given to the products with brand b by group c_i (c_j).

Thus, the pairwise similarity of two groups is defined as the weighted average of the above measurements:

$$sc_{i,j} = \frac{\sum w_k s_k}{w_k s_k} \tag{4}$$

[2]Reviewers labeled as colluders are positive instances.

Algorithm 1: The KNN-based Method

Input : Training data: D; Test data: T; k
Output: Spam label for each reviewer v

1 **foreach** $v_i \in T$ **do**
2 | compute $sv_{i,j}$ with each instance $v_j = (\mathbf{a}_j, l_j) \in D$;
3 | choose k instance $v'_j \in D$ with highest nonzero sv;
4 | add to D_k;
5 | **if** $|D_k| < k$ **then**
6 | | add $KNN_{\mathbf{a}}(D, \mathbf{a}_j, k - |D_k|)$ to D_k;
7 | $l_i \leftarrow \arg\max_{l \in \{pos,neg\}} \sum_{v'_j \in D_k} I(l = l'_j)$;
8 **return** $\{l_i\}$;

where $s_k \in \{s_{cmr}, s_{cpr}, s_{cbrd}\}$ and w_k is the non negative weight for s_k such that $\sum w_k = 1$.

Having defined the pairwise similarity of two groups, the pairwise similarity of two reviewers is computed by taking the average over the pairwise similarity of each pair of their respective groups:

$$sv_{i,j} = \frac{\sum_{k \in C_i} \sum_{l \in C_j} sc_{k,l}}{|C_i| \cdot |C_j|} \quad (5)$$

where C_i and C_j are the set of groups that have reviewer v_i and v_j respectively. We thus present the design of our KNN-based method for colluder detection in Alg. 1. The k nearest neighbors of reviewer v_i are selected according to the pairwise similarity score computed with each reviewer v_j in training data D (lines 1-4). However, if there are not enough reviewers ($\leq k$) in D to achieve $sv_{i,j} > 0$, the vacancies will be filled by performing the traditional $KNN_{\mathbf{a}}(\mathbf{.})$ algorithm that computes the distance between two reviewers based on their own attribute vectors $\mathbf{a}$ (lines 5-6). Finally, the class label of reviewer v_i is assigned as the one that covers most of the reviewers in D_k, wherein $I(\mathbf{.})$ is the identity function that takes value 1 if $l = l'_j$ and 0 otherwise (line 7).

4.2 The Graph-based Classification Method

Although shown to be heuristic, the KNN-based method has made the first attempt to exploit the relational information of reviewer groups to conduct detection. Unlike the review contents or reviewing behaviors, the group structures of colluders are harder to fake because they have to review the assigned products to make profits. Once a set of colluders have reviewed multiple common products together, they will be merged into a group. Moreover, the KNN-based method explores the correlations among colluders by measuring their similarities. This is intuitive because colluders do not work alone. They are well-organized, thus they must be correlated. However, the KNN-based method may rely too much on groups. The major limitation is that in practice the parameters are hard to set when splitting reviewers into groups. If being tightly set, false negatives would increase otherwise false positives would grow up. In addition, the group-level relational information may become quite sparse in some datasets. We have shown that (Section 5.1) if there are not sufficient neighbors with non-zero sv for voting, the KNN-based method could degenerate into a traditional KNN classifier that only considers individuals' intrinsic attributes which are volatile in practice.

More general and flexible approaches are expected to detect colluders while carrying on the explorations made by the KNN-based method. We thus present our second method to detect colluders based on the observation that there is another type of information which may also be hard to deceive practically once the spam campaigns have taken place, which is the *transaction correlations* among reviewers. Given two reviewers and their transaction histories, their transaction correlation forms once they have both reviewed at least one product within a predefined time window. We observed that colluders are more likely to review the same products with other colluders within the period of the spam campaigns. Thus if we artificially link two reviewers with an edge once they are found to be *transactionally correlated* with each other bounding by a time window Δt, it turns out that colluders will be more likely to appear as the neighbors of other colluders than non-colluders. Mapping to the classification setting, we say that interlinked reviewers are more likely to have the same class labels than remote pairs. As such, the determination of a reviewer's class label can be influenced not only by his own attributes, but also by the class labels of the neighborhood. Thus the class labels of the interconnected reviewers can be inferred collectively during the classification, which is the very idea of *collective classification* [7, 10] that attempts to jointly classify a set of unlabeled instances which can be implicitly or explicitly interrelated. To this end, in order to conduct the classification collectively, we first need a graph model to represent the interconnections among reviewers and then a classification framework for the inference.

4.2.1 Colluder Graph Model

Our colluder graph model is based on the *pairwise Markov network* [11] as the assumption is made that the class labels of reviewers can only be inferred from the attributes of the corresponding reviewers and the class labels of the direct neighbors[3]. Accordingly, we define a colluder graph $\mathbf{CG} \triangleq \{\mathcal{L} \cup \mathcal{A}, \mathcal{E}, \Delta t\}$. The nodes set $\mathcal{L} = \{L_i\}_{i=1}^m$ is the set of the class labels to be assigned to each reviewer v_i, whose values $\{l_i | l_i \in \{pos, neg\}\}_{i=1}^m$ are unobserved and need to be determined. The nodes set $\mathcal{A} = \{A_j\}_{j=1}^n$ is the set of observed attributes associated with each reviewer which can be spam indicators as discussed in Section 3. $\mathcal{E}$ is the set of edges where $(L_i, L_j) \in \mathcal{E}$ if v_i and v_j have reviewed $\kappa (\kappa \geq 1)$ common product(s) within the time window Δt, and $(L_i, A_j) \in \mathcal{E}$ if A_j is one of the attributes associated with reviewer v_i whose class label is L_i. For brevity of notation, we denote by $\mathbf{A}_i = (A_{i1}, ..., A_{ik})$ the attribute vector associated with reviewer v_i, thus $(L_i, A_{ij}) \in \mathcal{E}$, $j \in [1, k]$, and by $\mathbf{a}_i = (a_{i1}, ..., a_{ik})$ the value of $\mathbf{A}_i$. Our goal is to collectively assign each reviewer v_i with an appropriate class label $l_i \in \{pos, neg\}$. Thus the colluder graph model $\mathbf{CG}$ is associated with the global probability distribution:

$$\log(Pr(\boldsymbol{l}|CG)) = \sum_{L_i \in \mathcal{L}} \log(\phi_i(l_i)) + \sum_{(L_i, L_j) \in \mathcal{E}} \log(\psi_{ij}(l_i, l_j)) - \log Z \quad (6)$$

[3]This is reasonable because based on the rules of the formation of the edges between reviewers' labels shown later in the definition of our colluder graph model, the interrelations between one reviewer and the neighbors of his direct neighbors make little sense provided that they may not have necessarily reviewed common products at all.

$$\phi_i(l_i) = \psi_i(l_i) \prod_{(L_i, A_j) \in \mathcal{E}} \psi_{ij}(l_i) \quad (7)$$

where $\psi_i(l_i)$, $\psi_{ij}(l_i)$, $\psi_{ij}(l_i, l_j)$ correspond to the potential functions over three types of cliques $L_i \in \mathcal{L}$, $(L_i, A_j) \in \mathcal{E}$ and $(L_i, L_j) \in \mathcal{E}$ respectively. Z is the regularization factor. In Eq.(6), ϕ_i can be obtained through the computation of the distribution over l_i given the attribute $\mathbf{a}_i$ associated with reviewer v_i; while ψ_{ij} should involve relational information of reviewers v_i and v_j to allow the adjacent class labels to affect the classification result. The definition of the potential functions will be presented in the subsequent section, together with the presentation of the inference algorithm. Given Eq.(6), our goal is to find the appropriate configuration of the class labels $\hat{\boldsymbol{l}}$ for all the reviewers that maximize the following objection function:

$$\hat{\boldsymbol{l}} = \arg\max_{\boldsymbol{l}} \log(Pr(\boldsymbol{l}|CG)) \quad (8)$$

4.2.2 Collective Inference Algorithm

It is feasible to perform exact inference for a given Markov network if it has special structures such as trees. However, in our case, a CG typically consists of thousands of nodes and loops; thus it becomes impossible to apply exact inference to the optimization function (Eq.(8)). Hence approximate inference algorithms are needed to tackle this issue. Previously, [9] and [5] have evaluated different approximate inference algorithms on synthetic data and real world data respectively and found Iterative Classification Algorithm (ICA) to be more reliable. As such, we will base our design of the collective inference algorithm on ICA [7] to infer the probable class labels for each reviewer.

We first define the potential functions used in Eq.(6) as:

$$\phi_i(l_i) = Pr(l_i|\mathbf{a}_i) \quad (9)$$

$$\psi_{ij}(l_i, l_j) = cr_{i,j} \cdot cd_{i,j} \cdot I(l_i = l_j) \quad (10)$$

where $\phi_i(l_i)$ is computed as the probability of assigning v_i with l_i given the attribute $\mathbf{a}_i$. $I(\bullet)$ is the identity function (see Section 4.1). $cr_{i,j}$ and $cd_{i,j}$ are *collusion scoring functions* where $cr_{i,j}$ calculates the *collusive rate* of v_i and v_j which captures the collaboration frequency of v_i and v_j during the period Δt, and $cd_{i,j}$ calculates the *collusive degree* of v_i and v_j which captures the collaboration intensity of v_i and v_j during the period Δt. They are formalized as:

$$cr_{i,j} = \frac{\#(g_i \cap g_j)}{\#(g_i \cup g_j)} \quad (11)$$

$$cd_{i,j} = \frac{1}{1 + \sqrt{\frac{1}{n}\sum_{k=1}^{n}(r_{ik} - r_{jk})^2}} \quad (12)$$

where in Eq.(11) g_i (g_j) denotes the products reviewed by v_i (v_j) within Δt. $\#(g_i \cap g_j)$ denotes the number of products commonly reviewed by v_i and v_j within Δt, and $\#(g_i \cup g_j)$ denotes the number of products reviewed by either v_i or v_j within Δt. $cr_{i,j}$ equals to 1 if both of them have exactly reviewed the same set of products. The intuition is that the more common products two reviewers have reviewed within a certain period, the more likely they may collude with each other. Note that although this notion attaches little significance to non-colluders, the classification process will not be affected. In Eq.(12), r_{ik} (r_{jk}) is the rating given to product k by v_i (v_j). Thus $cd_{i,j}$ is the normalized inverted RMSD over all pairs of ratings given to the n common products reviewed by both v_i and v_j. $cd_{i,j}$ equals to 1 if both v_i and v_j give exact the same ratings to their common products. Note that Eq.(12) does not consider the rating scale of each reviewer because spammers always choose the same scale that is consistent with the perceptions of the masses (e.g., reviews with 4 or 5 stars are regarded as positive reviews while 1 or 2 stars as negative ones).

Algorithm 2: Collective Inference Algorithm

Input : Colluder Graph **CG**
Output: Spam label for each reviewer v

```
// Bootstrapping
1  foreach L_i ∈ L do
2      p_i ← max_l φ_i(l);   l_i ← arg max_l φ_i(l);
   // Iteration Classification
3  for q = 1 to M do
4      foreach L_i ∈ L do
5          p_i(l_i|CG, l) ← α exp{Σ_{L_j∈N(L_i)} cr_{i,j}·cd_{i,j}·p_j(l_i)·φ_i(l_i)}
           such that Σ_{l∈{pos,neg}} p_i(l|CG, l) = 1;
6          p_i ← max_l p_i(l|CG, l);
7          l_i ← arg max_l p_i(l|CG, l);
8      k ← (q/M) × |L|;
9      Update l_i with top-k p_i;
10 return {l_i};
```

The collective inference algorithm is presented in Alg. 2. l_i is the current most likely assignment of the class label for reviewer v_i, and p_i is the corresponding probability. α is a temporal normalized factor. For each reviewer, the unobserved class labels and the corresponding distributions are initialized based on the attribute-label clique potential functions (Eq.(9)) in the bootstrapping (lines 1-2). In the iteration classification, each iteration recomputes the distribution of the class label of each reviewer conditioned on the current class label distributions of the neighborhood by using the potential function $\psi_{ij}(l_i, l_j)$ (lines 4-7). At the end of each iteration, top-k confident class labels are updated where k linearly increases with the iteration times (lines 8-9).

5. EXPERIMENTAL ANALYSIS

5.1 Evaluation of the KNN-based Method

As shown in Alg. 1, the core of our KNN-based method is to compute the pairwise similarity of two reviewers and choose k most similar ones for voting. The performance will heavily rely on the hypothesis that the pairwise similarity of two reviewers within the same class should be higher than that between reviewers having opposite class labels. Fig. 6 shows the distributions of the pairwise similarity scores over three types of reviewer pairs: neg-neg pairs, neg-pos pairs and pos-pos pairs. Recall that colluders are considered as positive instances.

Fig. 6 shows clear separations between the three curves. The pos-pos pairs generally have much higher sv (the pairwise similarity of two reviewers) than the other two. In Section 3.4 we have revealed that colluders are often found in multiple similar groups in terms of overlapped members,

Table 1: Statistics of colluder graphs with different time window settings. The statistics of the colluder graphs corresponding to $\Delta t = 14, 28, 42, 56, 70, 84, 98, 112$ are not shown due to the space constraints.

	Size	#(pos)	#(neg)	#(neg-neg)	#(neg-pos)	#(pos-pos)	%(neg-neg)	%(neg-pos)	%(pos-pos)	Density
$\Delta t = 7$	4496	1894	2602	6416	1473	22420	**0.212**	**0.049**	**0.740**	0.0030
$\Delta t = 21$	4869	1924	2945	15618	2989	40298	0.265	0.051	0.684	0.0050
$\Delta t = 35$	4931	1928	3003	24066	4348	53815	0.293	0.053	0.654	0.0068
$\Delta t = 49$	4963	1932	3031	31848	5537	64190	0.313	0.055	0.632	0.0082
$\Delta t = 63$	4978	1932	3046	39179	6674	73462	0.328	0.056	0.616	0.0096
$\Delta t = 77$	4997	1935	3062	46165	7595	82032	0.339	0.056	0.604	0.0108
$\Delta t = 91$	5010	1935	3075	52587	8571	89575	0.349	0.057	0.594	0.0120
$\Delta t = 105$	5020	1935	3085	58951	9412	96107	0.358	0.057	0.584	0.0131
$\Delta t = 119$	5023	1935	3088	65195	10176	101848	**0.368**	**0.057**	**0.575**	0.0141

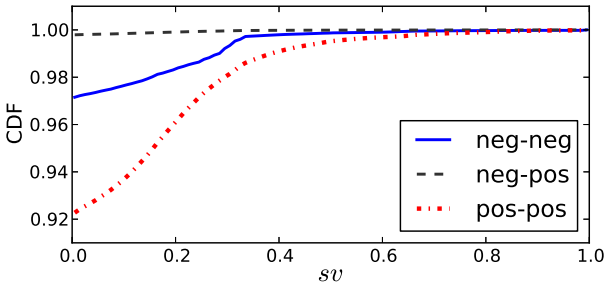

Figure 6: Distributions of the pairwise similarity scores over three different type of reviewer pairs.

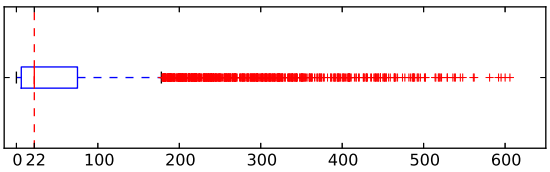

Figure 7: Distribution of the number of the neighbors with non-zero sv for each reviewer.

commonly reviewed products and similar ratings. These features are all captured by the three similarity measures defined in Section 4.1. When choosing the nearest neighbors with the top-k highest sv for voting a colluder's class label, as much more pos-pos pairs achieve higher sv than the neg-pos ones, most chosen nearest neighbors will be positive instances, which is the very case we desire. Similar situation applies to non-colluder instances given the apparent gaps between the curves of neg-pos pairs and neg-neg pairs. It is also worth noting that the distribution of sv is quite sparse, over 97.5% of the pairwise similarity scores between any of the reviewer pairs in our dataset are zero. This would be problematic when many instances in the dataset do not have enough neighbors with nonzero sv for voting, our KNN-based method would then degenerate into a traditional KNN classifier that only considers individuals' attributes for decision making. In our case, half of the reviewers have no more than 22 neighbors with non-zero sv (Fig. 7). Thus our method is in essence an extension of the classic classification approaches by exploiting the reviewer group-level relational information to improve the final performance.

Finally, we evaluate the classification performance of the KNN-based method using the standard metrics - precision, recall and f1-score where precision and recall are the ratio of the predicted true colluders to the predicted reviewers and true colluders, respectively. Besides, as the classes in our dataset are of quite different sizes, MCC (Matthews Correlation Coefficient) is also included which is a more balanced measure for skewed binary classifications. 10-fold cross-validation is used to create dataset splits for training and testing based on the dataset presented in Section 2. The combination of all the spam indicators LI+IBI+CBI is used as the attributes $\mathbf{a}$ for the traditional $KNN_{\mathbf{a}}(\cdot)$ algorithm. For simplicity, the weights in Eq.(4) are equally set. The results are shown in Fig. 8. We can see that our KNN-based method for colluder detection can achieve encouraging results when the k is not too large. Specifically, when $k = 5$ the F1-score attains 0.914, which drops accordingly as k becomes larger. This again verifies the fact that as more neighbors are set to be chosen from the training set, the ones in the test set may have greater portion of neighbors with whom the pairwise similarity scores attain zero. The traditional KNN algorithm thus steps in to fill up the vacancies. As such, subsequent experiments will choose small k to reduce the chance of degeneration.

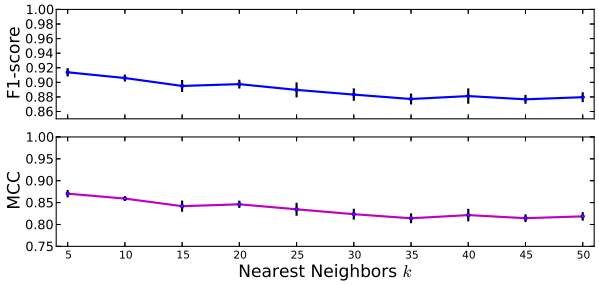

Figure 8: Performance (F1-score and MCC) of the KNN-based method vs. the number of nearest neighbors k. Error bars show the standard error of the mean.

5.2 Evaluation of the Graph-based Method

We first generate a collection of colluder graphs according to the definition described in Section 4.2.1 using different parameter settings. Recall that an edge forms between the class labels of two reviewers if they have reviewed $\kappa(\kappa \geq 1)$ common product(s) within a time window Δt. In our experiments, we set κ to 1 so as to capture the potential edges among reviewers as many as possible. The time window Δt is set from 7 to 119 days with an interval of one week long. For each parameter setting, the largest connected component of the resulting graph is taken as an instance of our colluder graphs. The statistics of the resulting colluder graphs are shown in Table 1. For data preparation, as Alg. 2 is fed into networked data, k-fold cross-validation like methods may not be suitable because splitting the dataset randomly

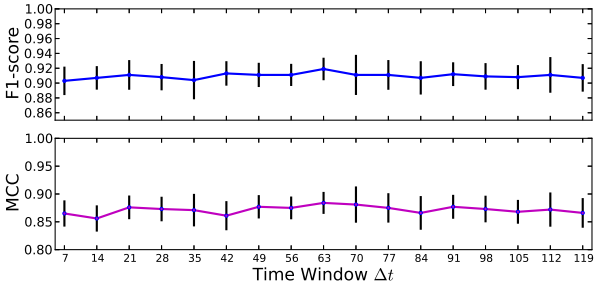

Figure 9: Performance (F1-score and MCC) of the graph-based classification method with different time windows. Error bars show the standard error of the mean.

into k subsets and using $k-1$ of them for training will lead to an expectation of $(k-1)/k$ of neighbors of a test node being labeled, which can bias the classification results. Instead, we use a breadth-first search (BFS) based strategy in which we construct splits for test data by randomly selecting the starting node and expanding around it till a predefined sample size has been reached[4]. The selected set of nodes Te are used for testing and the rest, denoted as Tr, are used for training. Note that, evaluation metrics are measured *only* on the subset of Te that has no neighbors in Tr. In our experiments, for a given colluder graph, we repeat this process multiple times and obtain 10 test-train pairs of splits where the class distribution of each test data is close to that of the entire dataset (near stratified). The resulting collection of splits for a given colluder graph with time window Δt is denoted as $BS_{\Delta t}$. SVM is used as the local classifier in the bootstrapping and the combination of all the spam indicators LI+IBI+CBI serves as the attribute set. The iteration number is set to 10. Fig. 9 shows the performance of our graph-based classification method (GC).

We observe that GC achieves promising results in a steady manner as the colluder graph expands incrementally. This is not surprising because at the beginning when $\Delta t = 7$ the colluder graph has already possessed very high homophily: $\frac{\#(pos-pos)}{\#(pos-pos)+\#(neg-pos)} = 93.8\%$ (from Table 1) of the colluder-induced edges connect colluders at both ends, meaning that a colluder will have a chance of 93.8% to choose another colluder as neighbors; similarly, a non-colluder will have a chance of 81.3% to choose another non-colluder as neighbors. This property greatly benefits the iteration phase of the collective inference algorithm where class labels are updated by seeking neighborhood for consulting.

As shown in Table 1, as the time window becomes larger, more and more neg-neg and neg-pos edges are added to the colluder graph. When the time window reaches 119 days, the pos-pos edges ratio drops to 57.5% while the neg-pos edges ratio goes up to 5.7%. One may worry that the added neg-pos edges could possibly affect the relabeling of colluders in each iteration because colluders are expected to have more non-colluder neighbors than before. However, the answer is no. Because when deciding the class labels of each reviewer, in addition to the neighbors' class labels being taken into account, the collusion scoring functions (cr and cd) also take effect as the weights of the edges. As shown in Fig. 10, as the time window becomes larger, the boxes of neg-pos edges and

[4]Empirically, we set the predefined sample size as half of the size of the corresponding colluder graph.

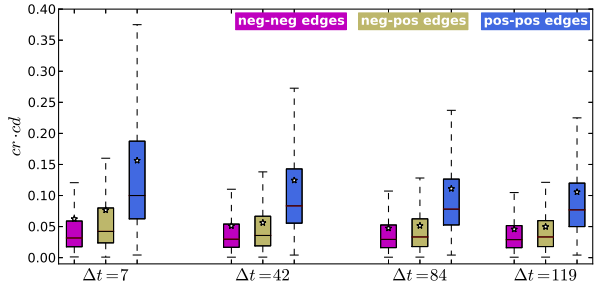

Figure 10: The distribution of the weights ($cr \cdot cd$) of the colluder graph edges vs. different time window settings with $\Delta t = 7, 42, 84, 119$.

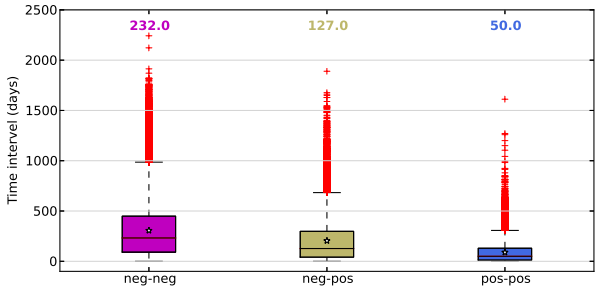

Figure 11: Distribution of the time intervals associated with different types of edges in the colluder graph with $\Delta t = +\infty$. The median values are displayed above the corresponding boxes.

pos-pos edges are shifting downwards due to many newly added edges having lower $cr \cdot cd$ (also note that the $cr \cdot cd$ of neg-pos edges are significantly lower than that of pos-pos edges all the time). As a result, these newly added neg-pos edges would exert much weaker effects on the determination of the class labels of colluders than the "older" pos-pos edges such that the overall performance will not get worse (Fig. 9).

How to set the time window properly to form a "good" colluder graph that is sufficient to catch most of the colluders in Chinese review websites? We argue that in practice the time window can be set based on the domain knowledge from anti-spam experts or the experiences of website mediators. In our case we plot the distributions of time intervals associated with different types of edges in the colluder graph with $\Delta t = +\infty$ (Fig. 11). The median value (50 days) of the time intervals associated with all the pos-pos edges seems to be a fair lower bound for the time window Δt. With this setting, most of the pos-pos edges are retained and not many neg-post are included so as to achieve a relatively high purity of the colluders' neighborhood, on the other hand, we can see from Table 1 that when $\Delta t = 49$ the colluder graph have already covered 1932/1937=99.7% colluders.

5.3 Performance Comparison

Finally, we compare our two methods (KNN+[5] and GC) with two baseline indicator-only classifiers (KNN and SVM). For KNN and KNN+ we set $k = 5$. For GC, SVM is also used as the local classifier in the bootstrapping. We set $\Delta t = 50$. Thus BS_{50} is used as the evaluation dataset. Five spam indicator sets - LI, IBI, CBI, BI (IBI+CBI) and their combination (ALL) - are evaluated with each of the aforementioned classifiers. The results are shown in Table 2.

[5]We here denote our KNN-base method as KNN+.

Table 2: Performance comparison of the proposed methods (KNN+ and GC) and two baseline classifiers (KNN and SVM). Our proposed methods *respectively* improve over both of the baseline classifiers at the confident level of 95% based on two-tailed t-test.

	Precision	Recall	F1-score	MCC		Precision	Recall	F1-score	MCC
KNN(LI)	0.606±0.010	0.698±0.012	0.648±0.007	0.456±0.013	**SVM(LI)**	0.631±0.012	0.624±0.026	0.627±0.016	0.489±0.020
KNN(IBI)	0.658±0.012	0.688±0.017	0.672±0.012	0.424±0.019	**SVM(IBI)**	0.682±0.013	0.625±0.009	0.652±0.009	0.452±0.012
KNN(CBI)	0.795±0.015	0.749±0.017	0.771±0.015	0.663±0.020	**SVM(CBI)**	0.762±0.008	0.794±0.008	0.777±0.007	0.638±0.010
KNN(BI)	0.789±0.017	0.819±0.015	0.803±0.014	0.705±0.019	**SVM(BI)**	0.833±0.010	0.827±0.008	0.829±0.007	0.725±0.010
KNN(ALL)	0.805±0.015	0.858±0.010	0.830±0.009	0.742±0.014	**SVM(ALL)**	0.840±0.012	0.828±0.013	0.834±0.011	0.732±0.017
KNN+(LI)	0.869±0.019	0.936±0.007	0.900±0.013	0.852±0.018	**GC(LI)**	0.849±0.016	0.866±0.037	0.854±0.022	0.790±0.026
KNN+(IBI)	0.870±0.010	0.937±0.009	0.902±0.005	0.852±0.009	**GC(IBI)**	0.866±0.014	0.859±0.036	0.859±0.021	0.798±0.025
KNN+(CBI)	0.888±0.011	0.930±0.007	0.908±0.007	0.863±0.008	**GC(CBI)**	0.893±0.018	0.887±0.014	0.888±0.010	0.841±0.012
KNN+(BI)	0.877±0.011	0.933±0.011	0.903±0.005	0.855±0.008	**GC(BI)**	**0.939±0.011**	0.904±0.018	**0.919±0.009**	0.874±0.016
KNN+(ALL)	0.874±0.010	**0.945±0.004**	0.909±0.004	0.863±0.005	**GC(ALL)**	0.927±0.010	0.908±0.029	0.914±0.019	**0.879±0.025**

We observe that both our methods promisingly outperform the baseline classifiers on all indicator sets. KNN+ achieves stably well regardless of the indicator sets. The F1-score of KNN+ is improved over the baseline classifiers by 8.9% to 43.5%. This is not surprising because when k is small, most reviewers will succeed in finding the most similar neighbors who are very likely to have the same class labels, thus KNN+ will not degenerate into traditional KNN (using spam indicators only), the results will not be affected by specific indicator sets. GC also works well, whose F1-score is increased over the baseline classifiers by 9.6% to 36.2%. By comparing the two proposed methods we find that neither is statistically more significant than each other. Comparatively, GC slightly suffers from inferior indicators (LI and IBI), however, it will recover and even rush ahead once better indicators are utilized for bootstrapping. This is because once the local classifier of GC makes some errors locally in the bootstrapping, the errors would propagate to other parts of the network within just a few iterations. Finally, by using more powerful indicator sets like BI and ALL, GC achieves the best results in terms of both F1-score (GC(BI)) and MCC (GC(ALL)).

6. CONCLUSION AND FUTURE WORK

In this paper, we detect colluders in Chinese online reviews. Empirical analysis is conducted on recently crawled product reviews from a popular Chinese e-commerce website. Anomalies are spotted not only in the languages colluders use but also in the behaviors they act, causing the failures of many inspected state-of-the-art spam indicators. Two novel methods are then proposed. Both of the methods have made good use of the invariant concealed in the dynamics of spam campaigns and treat the reviewers' behavioral histories as relational data. Experimental results show that both of the methods promisingly outperform the baseline indicator-only classifiers. In the future, we will continue to verify our proposed approaches on more datasets.

7. ACKNOWLEDGMENTS

We thank the anonymous reviewers for their helpful comments. This work is supported by the MoE AcRF Tier 2 Grant M4020110.020.

8. REFERENCES

[1] S. Feng, L. Xing, A. Gogar, and Y. Choi. Distributional footprints of deceptive product reviews. In *Proceedings of the International AAAI Conference on WebBlogs and Social Media (ICWSM)*, 2012.

[2] N. Jindal and B. Liu. Opinion spam and analysis. In *Proceedings of the International Conference on Web Search and Web Data Mining (WSDM)*, 2008.

[3] F. Li, M. Huang, Y. Yang, and X. Zhu. Learning to identify review spam. In *Proceedings of the 22nd International Joint Conference on Artificial Intelligence (IJCAI)*, pages 2488–2493, 2011.

[4] E. Lim, V. Nguyen, N. Jindal, B. Liu, and H. Lauw. Detecting product review spammers using rating behaviors. In *Proceedings of the 19th ACM International Conference on Information and Knowledge Management (CIKM)*, 2010.

[5] L. K. McDowell, K. M. Gupta, and D. W. Aha. Cautious inference in collective classification. In *Proceedings of the National Conference on Artificial Intelligence (AAAI)*, 2007.

[6] A. Mukherjee, B. Liu, and N. Glance. Spotting fake reviewer groups in consumer reviews. In *Proceedings of the International Conference on World Wide Web (WWW)*, 2012.

[7] J. Neville and D. Jensen. Iterative classification in relational data. In *Proceedings of the AAAI Workshop on Learning Statistical Models from Relational Data*, 2000.

[8] M. Ott, Y. Choi, C. Cardie, and J. T. Hancock. Finding deceptive opinion spam by any stretch of the imagination. *arXiv preprint arXiv:1107.4557*, 2011.

[9] P. Sen and L. Getoor. Empirical comparison of approximate inference algorithms for networked data. In *Proceedings of ICML Workshop on Statistical Relational Learning (SRL)*, 2006.

[10] P. Sen, G. Namata, M. Bilgic, L. Getoor, B. Galligher, and T. Eliassi-Rad. Collective classification in network data. *AI magazine*, 29(3):93, 2008.

[11] B. Taskar, P. Abbeel, and D. Koller. Discriminative probabilistic models for relational data. In *Proceedings of the Eighteenth conference on Uncertainty in artificial intelligence*, 2002.

[12] G. Wang, S. Xie, B. Liu, and S. Philip. Identify online store review spammers via social review graph. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 3(4), 2012.