Spotting Fake Reviewer Groups in Consumer Reviews

Arjun Mukherjee
Department of Computer Science
University of Illinois at Chicago
851 S. Morgan, Chicago, IL 60607
arjun4787@gmail.com

Bing Liu
Department of Computer Science
University of Illinois at Chicago
851 S. Morgan, Chicago, IL 60607
liub@cs.uic.edu

Natalie Glance
Google Inc.
4720 Forbes Ave, Lower Level
Pittsburgh, PA 15213
nglance@google.com

ABSTRACT

Opinionated social media such as product reviews are now widely used by individuals and organizations for their decision making. However, due to the reason of profit or fame, people try to game the system by opinion spamming (e.g., writing fake reviews) to promote or demote some target products. For reviews to reflect genuine user experiences and opinions, such spam reviews should be detected. Prior works on opinion spam focused on detecting fake reviews and individual fake reviewers. However, a fake reviewer group (a group of reviewers who work collaboratively to write fake reviews) is even more damaging as they can take total control of the sentiment on the target product due to its size. This paper studies spam detection in the collaborative setting, i.e., to discover fake reviewer groups. The proposed method first uses a frequent itemset mining method to find a set of candidate groups. It then uses several behavioral models derived from the collusion phenomenon among fake reviewers and relation models based on the relationships among groups, individual reviewers, and products they reviewed to detect fake reviewer groups. Additionally, we also built a labeled dataset of fake reviewer groups. Although labeling individual fake reviews and reviewers is very hard, to our surprise labeling fake reviewer groups is much easier. We also note that the proposed technique departs from the traditional supervised learning approach for spam detection because of the inherent nature of our problem which makes the classic supervised learning approach less effective. Experimental results show that the proposed method outperforms multiple strong baselines including the state-of-the-art supervised classification, regression, and learning to rank algorithms.

Categories and Subject Descriptors

H.1.2 [**Information Systems**]: Human Factors; J.4 [**Computer Applications**]: Social and Behavioral Sciences

Keywords

Opinion Spam, Group Opinion Spam, Fake Review Detection

1. INTRODUCTION

Nowadays, if one wants to buy a product, most probably, one will first read reviews of the product. If he/she finds that most reviews are positive, he/she is very likely to buy it. However, if most reviews are negative, he/she will almost certainly choose another product. Positive opinions can result in significant financial gains and fames for organizations and individuals. This, unfortunately, gives strong incentives for *opinion spamming*, which refers to human activities (e.g., writing fake reviews) that try to deliberately mislead readers by giving unfair reviews to some entities (e.g. products) in order to promote them or to damage their reputations. As more and more individuals and organizations are using reviews for their decision making, detecting such fake reviews becomes a pressing issue. The problem has been widely reported in the news[1].

There are prior works [14, 15, 23, 24, 31, 32, 34] on detecting fake reviews and individual fake reviewers or spammers. However, limited research has been done to detect fake reviewer (or spammer) groups, which we also call *spammer groups*. Group spamming refers to a group of reviewers writing fake reviews together to promote or to demote some target products. A spammer group can be highly damaging as it can take total control of the sentiment on a product because a group has many people to write fake reviews. Our experiments show that it is hard to detect spammer groups using review content features [31] or even indicators for detecting abnormal behaviors of individual spammers [24] because a group has more manpower to post reviews and thus, each member may no longer appear to behave abnormally. Note that by a group of reviewers, we mean a set of reviewer-ids. The actual reviewers behind the ids could be a single person with multiple ids (sockpuppet), multiple persons, or a combination of both. We do not distinguish them in this work.

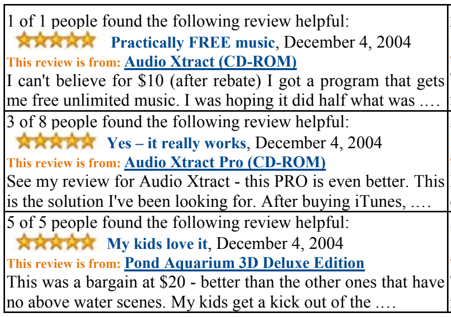
Before proceeding further, let us see a spammer group found by our algorithm. Figures 1, 2, and 3 show the reviews of a group of three reviewers[2]. The following suspicious patterns can be noted about this group: (i) the group members all reviewed the same three products giving all 5 star ratings; (ii) they posted reviews within a small time window of 4 days (two of them posted in the same day); (iii) each of them only reviewed the three products (when our Amazon review data [14] was crawled); (iv) they were among the early reviewers for the products (to make a big impact). All these patterns occurring together strongly suggest suspicious activities. Notice also, none of the reviews themselves are similar to each other (i.e., not duplicates) or appear deceptive. If we only look at the three reviewers individually, they all appear genuine. In fact, 5 out of 9 reviews received 100% helpfulness votes by Amazon users indicating that the reviews are useful. Clearly, these three reviewers have taken total control of the sentiment on the set of reviewed products. In fact, there is a fourth reviewer in the group. Due to space limitations, we omit it here.

If a group of reviewers work together only once to promote or to demote a product, it is hard to detect them based on their collective behavior. They may be detected using the content of their reviews, e.g., copying each other. Then, the methods in [14, 23, 24, 31, 32, 34] are applicable. However, over the years, opinion spamming has become a business. People get paid to write fake reviews. Such people cannot just write a single review

WWW 2012, April 16–20, 2012, Lyon, France.
ACM 978-1-4503-1229-5/12/04.

[1] http://www.nytimes.com/2012/01/27/technology/for-2-a-star-a-retailer-gets-5-star-reviews.html
http://www.nytimes.com/2011/08/20/technology/finding-fake-reviews-online.html

[2] http://www.amazon.com/gp/pdp/profile/A3URRTIZEE8R7W
http://www.amazon.com/gp/pdp/profile/A254LYRIZUYXZG
http://www.amazon.com/gp/cdp/member-reviews/A1O70AIHNS4EIY

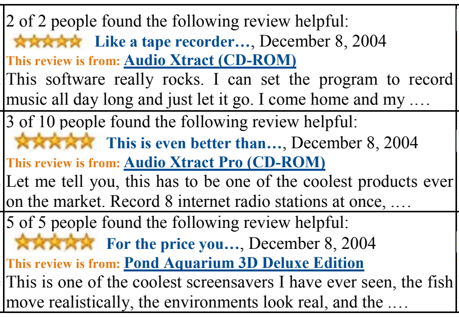


1 of 1 people found the following review helpful:
★★★★★ **Practically FREE music**, December 4, 2004
This review is from: Audio Xtract (CD-ROM)
I can't believe for $10 (after rebate) I got a program that gets me free unlimited music. I was hoping it did half what was ….

3 of 8 people found the following review helpful:
★★★★★ **Yes – it really works**, December 4, 2004
This review is from: Audio Xtract Pro (CD-ROM)
See my review for Audio Xtract - this PRO is even better. This is the solution I've been looking for. After buying iTunes, ….

5 of 5 people found the following review helpful:
★★★★★ **My kids love it**, December 4, 2004
This review is from: Pond Aquarium 3D Deluxe Edition
This was a bargain at $20 - better than the other ones that have no above water scenes. My kids get a kick out of the ….

Figure 1: Big John's Profile

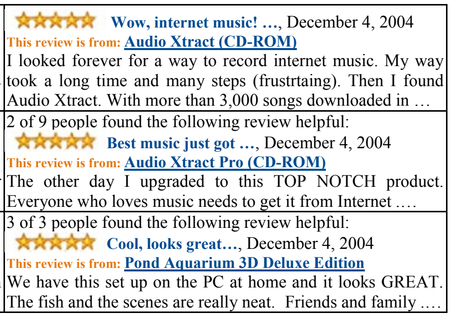


2 of 2 people found the following review helpful:
★★★★★ **Like a tape recorder…**, December 8, 2004
This review is from: Audio Xtract (CD-ROM)
This software really rocks. I can set the program to record music all day long and just let it go. I come home and my ….

3 of 10 people found the following review helpful:
★★★★★ **This is even better than…**, December 8, 2004
This review is from: Audio Xtract Pro (CD-ROM)
Let me tell you, this has to be one of the coolest products ever on the market. Record 8 internet radio stations at once, ….

5 of 5 people found the following review helpful:
★★★★★ **For the price you…**, December 8, 2004
This review is from: Pond Aquarium 3D Deluxe Edition
This is one of the coolest screensavers I have ever seen, the fish move realistically, the environments look real, and the ….

Figure 2: Cletus' Profile



★★★★★ **Wow, internet music! …**, December 4, 2004
This review is from: Audio Xtract (CD-ROM)
I looked forever for a way to record internet music. My way took a long time and many steps (frustrtaing). Then I found Audio Xtract. With more than 3,000 songs downloaded in …

2 of 9 people found the following review helpful:
★★★★★ **Best music just got …**, December 4, 2004
This review is from: Audio Xtract Pro (CD-ROM)
The other day I upgraded to this TOP NOTCH product. Everyone who loves music needs to get it from Internet ….

3 of 3 people found the following review helpful:
★★★★★ **Cool, looks great…**, December 4, 2004
This review is from: Pond Aquarium 3D Deluxe Edition
We have this set up on the PC at home and it looks GREAT. The fish and the scenes are really neat. Friends and family ….

Figure 3: Jake's Profile

as they would not make enough money that way. Instead, they write many reviews about many products. Such collective behaviors of a group working together on a number of products can give them away. This paper focuses on detecting such groups.

Since reviewers in the group write reviews on multiple products, the data mining technique *frequent itemset mining* (FIM) [1] can be used to find them. However, so discovered groups are only group spam candidates because many groups may be coincidental as some reviewers happen to review the same set of products due to similar tastes and popularity of the products (e.g., many people review all 3 Apple products, iPod, iPhone, and iPad). Thus, our focus is to identify true spammer groups from the candidate set.

One key difficulty for opinion spam detection is that it is very hard to manually label fake reviews or reviewers for model building because it is almost impossible to recognize spam by just reading each individual review [14]. In this work, multiple experts were employed to create a labeled group opinion spammer dataset. This research makes the following main contributions:

1. It produces a labeled group spam dataset. To the best of our knowledge, this is the first such dataset. What was surprising and also encouraging to us was that unlike judging individual fake reviews or reviewers, judging fake reviewer groups were considerably easier due to the group context and their collective behaviors. We will discuss this in Sec. 4.
2. It proposes a novel relation-based approach to detecting spammer groups. With the labeled dataset, the traditional approach of supervised learning can be applied [14, 23, 31]. However, we show that this approach can be inferior due to the inherent nature of our particular problem:
   (i) Traditional learning assumes that individual instances are independent of one another. However, in our case, groups are clearly not independent of one another as different groups may share members. One consequence of this is that if a group *i* is found to be a spammer group, then the other groups that share members with group *i* are likely to be spammer groups too. The reverse may also hold.
   (ii) It is hard for features used to represent each group in learning to consider each individual member's behavior on each individual product, i.e., a group can conceal a lot of internal details. This results in severe information loss, and consequently low accuracy.

   We discuss these and other issues in greater detail in Sec. 7. To exploit the relationships of groups, individual members, and products they reviewed, a novel relation-based approach is proposed, which we call GSRank (Group Spam Rank), to rank candidate groups based on their likelihoods for being spam.
3. A comprehensive evaluation has been conducted to evaluate GSRank. Experimental results show that it outperforms many strong baselines including the state-of-the-art learning to rank, supervised classification and regression algorithms.

2. RELATED WORK

The problem of detecting review or opinion spam was introduced in [14], which used supervised learning to detect individual fake reviews. Duplicate and near duplicate reviews which are almost certainly fake reviews were used as positive training data. While [24] found different types of behavior abnormalities of reviewers, [15] proposed a method based on unexpected class association rules and [31] employed standard word and part-of-speech (POS) n-gram features for supervised learning. [23] also used supervised learning with additional features. [32] used a graph-based method to find fake store reviewers. A distortion based method was proposed in [34]. None of them deal with group spam. In [29], we proposed an initial group spam detection method, but it is much less effective than the proposed method in this paper.

In a wide field, the most investigated spam activities have been in the domains of Web [4, 5, 28, 30, 33, 35] and Email [6]. Web spam has two main types: *content spam* and *link spam*. Link spam is spam on hyperlinks, which does not exist in reviews as there is usually no link in them. Content spam adds irrelevant words in pages to fool search engines. Reviewers do not add irrelevant words in their reviews. Email spam usually refers to unsolicited commercial ads. Although exists, ads in reviews are rare.

Recent studies on spam also extended to blogs [20], online tagging [21], and social networks [2]. However, their dynamics are different from those of product reviews. They also do not study group spam. Other literature related to group activities include mining groups in WLAN [13]; mobile users [8] using network logs, and community discovery based on interests [36].

Sybil Attacks [7] in security create pseudo identities to subvert a reputation system. In the online context, pseudo identities in Sybil attacks are known as sockpuppets. Indeed, sockpuppets are possible in reviews and our method can deal with them.

Lastly, [18, 25, 37] studied the usefulness or quality of reviews. However, opinion spam is a different concept as a low quality review may not be a spam or fake review.

3. BUILDING A REFERENCE DATASET

As mentioned earlier, there was no labeled dataset for group opinion spam before this project. To evaluate our method, we built a labeled dataset using expert human judges.

Opinion spam and labeling viability: [5] argues that classifying the concept "spam" is difficult. Research on Web [35], email [6], blogs [20], and even social spam [27] all rely on manually labeled data for detection. Due to this inherent nature of the problems, the closest that one can get to gold standards is by creating a manually labeled dataset using human experts [5, 21, 23, 27, 28]. We too built a group opinion spam dataset using human experts.

Amazon dataset: In this research, we used product reviews from Amazon [14], which have also been used in [15, 24]. The original crawl was done in 2006. Updates were made in early 2010. For our study, we only used reviews of manufactured products, which are comprised of 53,469 reviewers, 109,518 reviews and 39,392 products. Each review consisted of a title, content, star rating, posting date, and number of helpful feedbacks.

Mining candidate spammer groups: We use frequent itemset mining (FIM) here. In our context, a set of items, I is the set of all reviewer ids in our database. Each transaction t_i ($t_i \subseteq I$) is the set of reviewer ids who have reviewed a particular product. Thus, each product generates a transaction of reviewer ids. By mining frequent itemsets, we find groups of reviewers who have reviewed multiple products together. We found 7052 candidate groups with *minsup_c* (minimum support count) = 3 and at least 2 items (reviewer ids) per itemset (group), i.e., each group must have worked together on at least 3 products. Itemsets (groups) with support lower than this (*minsup_c* =1, 2) are very likely to be due to random chance rather than true correlation, and very low support also causes combinatorial explosion because the number of frequent itemsets grows exponentially for FIM [1]. FIM working on reviewer ids can also find *sockpuppeted* ids forming groups whenever the ids are used *minsup_c* times to post reviews.

Opinion spam signals: We reviewed prior research on opinion spam and guidelines on consumer sites such as consumerist.com, lifehacker.com and consumersearch.com[3], and collected from these sources a list of spamming indicators or signals, e.g., (i) having zero caveats, (ii) full of empty adjectives, (iii) purely glowing praises with no downsides, (iv) being left within a short period of time of each other, etc. These signals were given to our judges. We believe that these signals (and the additional information described below) enhance their judging rather than bias them because judging spam reviews and reviewers is very challenging. It is hard for anyone to know a large number of possible signals without substantial prior experiences. These signals on the Web and research papers have been compiled by experts with extensive experiences and domain knowledge. We also reminded our judges that these signals should be used at their discretion and encouraged them to use their own signals.

To reduce the judges' workload further, for each group we also provided 4 additional pieces of information as they are required by some of the above signals: reviews with posting dates of each individual group member, list of products reviewed by each member, reviews of products given by non-group members, and whether group reviews were tagged with AVP (Amazon Verified Purchase). Amazon tags each review with AVP if the reviewer actually bought the product. Judges were also given access to our database for querying based on their needs.

Labeling: We employed 8 expert judges: employees of Rediff Shopping (4) and eBay.in (4) for labeling our candidate groups. The judges had domain expertise in feedbacks and reviews of products due to their nature of work in online shopping. Since there were too many patterns (or candidate groups), our judges could only manage to label 2431 of them as being "spam", "non-spam", or "borderline". The judges were made to work in isolation to prevent any bias. The labeling took around 8 weeks. We did not use Amazon Mechanical Turk (MTurk) for this labeling task because MTurk is normally used to perform simple tasks which require human judgments. However, our task is highly challenging, time consuming, and also required the access to our database. Also, we needed judges with good knowledge of the review domain. Thus, we believe that MTurk was not suitable.

4. LABELING RESULTS

We now report the labeling results and analyze the agreements among the judges.

Spamicity: We calculated the "spamicity" (degree of spam) of each group by assigning 1 point for each *spam* judgment, 0.5 point for each *borderline* judgment and 0 point for each *non-spam* judgment a group received and took the average of all 8 labelers. We call this average the *spamicity* score of the group. Based on the spamicities, the groups can be ranked. In our evaluation, we will evaluate the proposed method to see whether it can rank similarly. In practice, one can also use a spamicity threshold to divide the candidate group set into two classes: *spam* and *non-spam* groups. Then supervised classification is applicable. We will discuss these in detail in the experiment section.

Agreement study: Previous studies have showed that labeling individual fake reviews and reviewers is hard [14]. To study the feasibility of labeling groups and also the judging quality, we used Fleiss' multi-rater kappa [10] to measure the judges' agreements. We obtained $\kappa = 0.79$ which indicates close to perfect agreement based on the scale[4] in [22]. This was very encouraging and also surprising, considering that judging opinion spam in general is hard [14]. It tells us that labeling groups seems to be much easier than labeling individual fake reviews or reviewers. We believe the reason is that unlike a single reviewer or review, a group gives a good context for judging and comparison, and similar behaviors among members often reveal strong signals. This was confirmed by our judges who had domain expertise in reviews.

5. SPAMMING BEHAVIOR INDICATORS

For modeling or learning, a set of effective spam indicators or features is needed. This section proposes two sets of such indicators or behaviors which may indicate spamming activities.

5.1 Group Spam Behavior Indicators

Here we discuss group behaviors that may indicate spam.

1. ***Group Time Window*** (***GTW***): Members in a spam group are likely to have worked together in posting reviews for the target products during a short time interval. We model the degree of active involvement of a group as its *group time window* (*GTW*):

$$GTW(g) = \max_{p \in P_g}(GTW_P(g,p)),$$
$$GTW_P(g,p) = \begin{cases} 0 & \text{if } L(g,p) - F(g,p) > \tau \\ 1 - \dfrac{L(g,p) - F(g,p)}{\tau} & \textit{otherwise} \end{cases}, \quad (1)$$

where $L(g, p)$ and $F(g, p)$ are the latest and earliest dates of reviews posted for product $p \in P_g$ by reviewers of group g respectively. P_g is the set of all products reviewed by group g. Thus, $GTW_P(g, p)$ gives the time window information of group g on a single product p. This definition says that a group g of reviewers posting reviews on a product p within a short burst of time is more prone to be spamming (attaining a value close to 1). Groups working over a longer time interval than τ, get a value of 0 as they are unlikely to have worked together. τ is a parameter, which we will estimate later. The group *time window* $GTW(g)$ considers all products reviewed by the group taking *max* over p ($\in P_g$) so as to capture the worst behavior of the group. For subsequent behaviors, *max* is taken for the same reason.

2. ***Group Deviation*** (***GD***): A highly damaging group spam occurs when the ratings of the group members deviate a great deal from

[3] http://consumerist.com/2010/04/how-you-spot-fake-online-reviews.html
http://lifehacker.com/5511726/hone-your-eye-for-fake-online-reviews
http://www.consumersearch.com/blog/how-to-spot-fake-user-reviews

[4] No agreement (κ<0), slight agreement ($0 < \kappa \le 0.2$), fair agreement ($0.2 < \kappa \le 0.4$), moderate agreement ($0.4 < \kappa \le 0.6$), substantial agreement ($0.6 < \kappa \le 0.8$), and almost perfect agreement for $0.8 < \kappa \le 1.0$.

those of other (genuine) reviewers so as to change the sentiment on a product. The larger the deviation, the worse the group is. This behavior is modeled by *group deviation* (*GD*) on a 5-star rating scale (with 4 being the maximum possible deviation):

$$GD(g) = \max_{p\in P_g}(\, D(g,p)), \qquad D(g,p) = \frac{|\, r_{p,g} - \bar{r}_{p,g}\,|}{4}, \tag{2}$$

where $r_{p,g}$ and $\bar{r}_{p,g}$ are the average ratings for product *p* given by members of group *g* and by other reviewers not in *g* respectively. *D*(*g*, *p*) is the deviation of the group on a single product *p*. If there are no other reviewers who have reviewed the product *p*, $\bar{r}_{p,g} = 0$.

3. ***Group Content Similarity*** (***GCS***): Group connivance is also exhibited by content similarity (duplicate or near duplicate reviews) when spammers copy reviews among themselves. So, the victimized products have many reviews with similar content. *Group content similarity* (*GCS*) models this behavior:

$$GCS(g) = \max_{p\in P_g}(CS_G(g,p)), \qquad CS_G(g,p) = \underset{m_i, m_j \in g, i<j}{avg}\left(cosine(c(m_i,p), c(m_j,p))\right), \tag{3}$$

where *c*(*m*$_i$, *p*) is the content of the review written by group member $m_i \in g$ for product *p*. $CS_G(g, p)$ captures the average pairwise similarity of review contents among group members for a product *p* by computing the cosine similarity.

4. ***Group Member Content Similarity*** (***GMCS***): Another flavor of content similarity is exhibited when the members of a group *g* do not know one another (and are contacted by a contracting agency). Since writing a new review every time is taxing, a group member may copy or modify his/her own previous reviews for similar products. If multiple members of the group do this, the group is more likely to be spamming. This behavior can be expressed by *group member content similarity* (*GMCS*) as follows:

$$GMCS(g) = \frac{\sum_{m\in g} CS_M(g,m)}{|g|}, \qquad CS_M(g,m) = \underset{p_i, p_j \in P_g, i<j}{avg}\left(cosine(c(m,p_i), c(m,p_j))\right) \tag{4}$$

The group attains a value ≈1 (indicating spam) on *GMCS* when all its members entirely copied their own reviews across different products in P_g. $CS_M(g, m)$ models the average pairwise content similarity of member $m \in g$ over all products in P_g.

5. **Group *Early Time Frame*** (***GETF***): [24] reports spammers usually review early to make the biggest impact. Similarly, when group members are among the very first people to review a product, they can totally hijack the sentiments on the products. The *group early time frame* (*GETF*) models this behavior:

$$GETF(g) = \max_{p\in P_g}(GTF(g,p)), \qquad GTF(g,p) = \begin{cases} 0 & \text{if } L(g,p) - A(p) > \beta \\ 1 - \dfrac{L(g,p) - A(p)}{\beta} & \text{otherwise} \end{cases}, \tag{5}$$

where *GTF*(*g*, *p*) captures the time frame as how early a group *g* reviews a product *p*. *L*(*g*, *p*) and *A*(*p*) are the latest date of review posted for product $p \in P_g$ by group *g* and the date when *p* was made available for reviewing respectively. β is a threshold (say 6 months, later estimated) which means that after β months, *GTF* attains a value of 0 as reviews posted then are not considered to be early any more. Since our experimental dataset [14] does not have the exact date when each product was launched, we use the date of the first review of the product as the value for *A*(*p*).

6. ***Group Size Ratio*** (***GSR***): The ratio of group size to the total number of reviewers for a product can also indicate spamming. At one extreme (worst case), the group members are the only reviewers of the product completely controlling the sentiment on the product. On the other hand, if the total number of reviewers of the product is very large, then the impact of the group is small.

$$GSR(g) = \underset{p\in P_g}{avg}(GSR_P(g,p)), \qquad GSR_P(g,p) = \frac{|g|}{|M_p|}, \tag{6}$$

where $GSR_P(g, p)$ is the ratio of group size to M_p (the set of all reviewers of product *p*) for product *p*.

7. ***Group Size*** (***GS***): Group collusion is also exhibited by its size. For large groups, the probability of members happening to be together by chance is small. Furthermore, the larger the group, the more damaging it is. *GS* is easy to model. We normalize it to [0, 1]. max($|g_i|$) is the largest group size of all discovered groups.

$$GS(g) = \frac{|g|}{\max(|g_i|)} \tag{7}$$

8. ***Group Support Count*** (***GSUP***): Support count of a group is the total number of products towards which the group has worked together. Groups with high support counts are more likely to be spam groups as the probability of a group of random people happen to have reviewed many products together is small. *GSUP* is modeled as follows. We normalize it to [0, 1], with max($|P_{g_i}|$) being the largest support count of all discovered groups:

$$GSUP(g) = \frac{|P_g|}{\max(|P_{g_i}|)} \tag{8}$$

These eight group behaviors can be seen as group spamming features for learning. From here on, we refer the 8 group behaviors as $f_1 \ldots f_8$ when used in the context of features.

It is important to note that by no means do we say that whenever a group attains a feature $f > 0$ or a threshold value, it is a spam group. It is possible that a group of reviewers due to similar tastes coincidently review some similar products (and form a coincidental group) in some short time frame, or may generate some deviation of ratings from the rest, or may even have modified some of the contents of their previous reviews to update their reviews producing similar reviews. The features just indicate the extent those group behaviors were exhibited. The final prediction of groups is done based on the learned models. As we will see in Sec. 6.2, all features $f_1 \ldots f_8$ are strongly correlated with spam groups and feature values attained by spam groups exceed those attained by other non-spam groups by a large margin.

5.2 Individual Spam Behavior Indicators

Although group behaviors are important, they hide a lot of details about its members. Clearly, individual members' behaviors also give signals for group spamming. We now present the behaviors for individual members used in this work.

1. ***Individual Rating Deviation*** (***IRD***): Like group deviation, we can model *IRD* as

$$IRD(m,p) = \frac{|\, r_{p,m} - \bar{r}_{p,m}\,|}{4}, \tag{9}$$

where $r_{p,m}$ and $\bar{r}_{p,m}$ are the rating for product *p* given by reviewer *m* and the average rating for *p* given by other reviewers respectively.

2. ***Individual Content Similarity*** (***ICS***): Individual spammers may review a product multiple times posting duplicate or near duplicate reviews to increase the product popularity [24]. Similar to *GMCS*, we model *ICS* of a reviewer *m* across all its reviews towards a product *p* as follows:

$$ICS(m,p) = avg\,(cosine\,(c(m,p)) \tag{10}$$

The average is taken over all reviews on *p* posted by *m*.

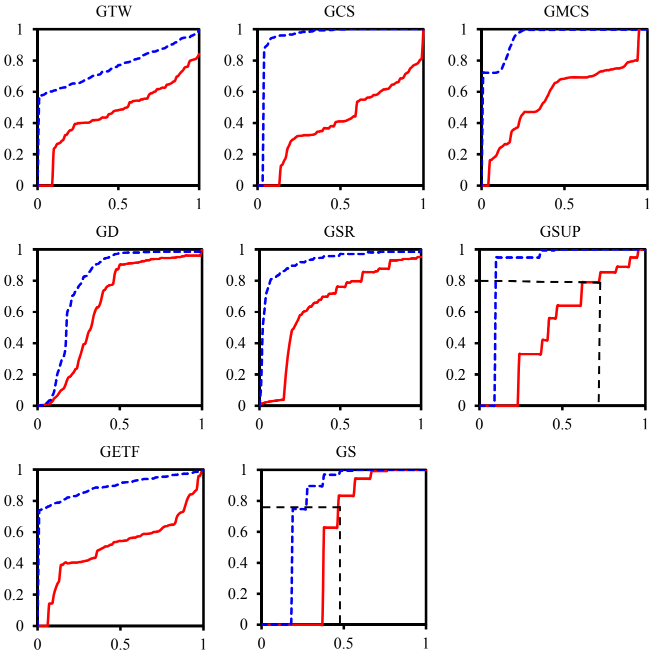


Figure 4: Behavioral Distribution. Cumulative % of spam (solid) and non-spam (dashed) groups vs. feature value

3. ***Individual Early Time Frame*** (***IETF***): Like *GETF*, we define *IETF* of a group member *m* as:

$$IETF(m,p)=\begin{cases}0 & \text{if } L(m,p)-A(p)>\beta\\ 1-\dfrac{L(m,p)-A(p)}{\beta} & \text{otherwise}\end{cases}, \tag{11}$$

where $L(m, p)$ denotes the latest date of review posted for a product *p* by member *m*.

4. ***Individual Member Coupling in a group*** (***IMC***): This behavior measures how closely a member works with the other members of the group. If a member *m* almost posts at the same date as other group members, then *m* is said to be tightly coupled with the group. However, if *m* posts at a date that is far away from the posting dates of the other members, then *m* is not tightly coupled with the group. We find the difference between the posting date of member *m* for product *p* and the average posting date of other members of the group for *p*. To compute time, we use the time when the first review was posted by the group for product *p* as the baseline. Individual member coupling (*IMC*) is thus modeled as:

$$IMC(g,m)=\underset{p\in P_g}{avg}\left(\frac{|(T(m,p)-F(g,p))-avg(g,m)|}{L(g,p)-F(g,p)}\right), \tag{12}$$

$$avg(g,m)=\frac{\sum_{m_i\in G-\{m\}}(T(m_i,p)-F(g,p))}{|g|-1},$$

where $L(g, p)$ and $F(g, p)$ are the latest and earliest dates of reviews posted for product $p \in P_g$ by group *g* respectively, and $T(m, p)$ is the actual posting date of reviewer *m* on product *p*.

Note that IP addresses of reviewers may also be of use for group spam detection. However, IP information is privately held by proprietary firms and not publicly available. We believe if IP addresses are also available, additional features may be added, which will make our proposed approach even more accurate.

6. EMPIRICAL ANALYSIS

To ensure that the proposed behavioral features are good indicators of group spamming, this section analyzes them by statistically validating their correlation with group spam. For this study, we used the classification setting for spam detection. A spamicity threshold of 0.5 was employed to divide all candidate groups into two categories, i.e., those with spamicity greater than 0.5 as *spam* groups and others as *non-spam* groups. Using this scheme, we get 62% non-spam groups and 38% spam groups. In Sec. 9, we will see that these features work well in general (rather than just for this particular threshold). Note that the individual spam indicators in Sec. 5.2 are not analyzed as there is no suitable labeled data for that. However, these indicators are similar to their group counterparts and are thus indirectly validated through the group indicators. They also helped GSRank well (Sec. 9).

6.1 Statistical Validation

For a given feature *f*, its effectiveness ($Eff(\cdot)$) is defined with:

$$Eff(f)\equiv P(f>0\,|\,Spam)-P(f>0\,|\,Non-spam), \tag{13}$$

where $f > 0$ is the event that the corresponding behavior is exhibited to some extent. Let the *null hypothesis* be: both spam and normal groups are equally likely to exhibit *f*, and the *alternate hypothesis*: spam groups are more likely to exhibit *f* than non-spam groups and are correlated with *f*. Thus, demonstrating that *f* is observed among spam groups and is correlated is reduced to show that $Eff(f) > 0$. We estimate the probabilities as follows:

$$P(f>0\,|\,Spam)=\frac{|\{g\,|\,f(g)>0\wedge g\in Spam\}|}{|\{g\,|\,g\in Spam\}|} \tag{14}$$

$$P(f>0\,|\,Non-spam)=\frac{|\{g\,|\,f(g)>0\wedge g\in Non-spam\}|}{|\{g\,|\,g\in Non-spam\}|} \tag{15}$$

We use Fisher's exact test to test the hypothesis. The test rejects the *null hypothesis* with p<0.0001 for each of the modeled behaviors. This shows that spam groups are indeed characterized by the modeled behaviors. Furthermore, since the modeled behaviors are all *anomalous*, and Fisher's exact test verifies strong correlation of those behaviors with groups labeled as "spam", it also indirectly gives us a strong confidence that the majority of the class labels in the reference dataset are trustworthy.

6.2 Behavioral Distribution

We now analyze the underlying distribution of spam and non-spam groups across each behavioral feature dimension. Figure 4 shows the cumulative behavioral distribution (CBD). Against each value *x* attained by a feature *f* ($0 \le x \le 1$ as $f \in [0, 1]\ \forall f$), we plot the cumulative percentage of spam/non-spam groups having values of $f \le x$. We note the following insights from the plots:

Position: CBD curves of non-spam groups lean more towards the left boundary of the graph than those for spam groups across all features. This implies that for a given cumulative percentage *cp*, the corresponding feature value x_n for non-spam groups is less than x_s for spam groups. For example, in CBD of the *GS* feature, if cp = 0.75, then 75% of the non-spam groups are bounded by x_n = 0.18 (i.e. 0.18×11≈2 members)[5] while 75% of the spam groups are bounded by x_s = 0.46 (i.e. 0.46×11≈5 members). As another example, we take CBD of *GSUP* with cp = 0.8. We see that 80% of the non-spam groups are bounded by x_n = 0.15 (i.e. 0.15×13 ≈ 2 products)[6] while 80% of spam groups are bounded by x_s = 0.76 (i.e. 0.76×13 ≈ 10 products). This shows that spamming groups usually work with more members and review more products. As non-spam groups are mostly coincidental, we find that their feature values remain low for most groups indicating benign behaviors. Also, we emphasize the term "bounded by" in the above description. By no means do we claim that every spam

[5] Dataset in Sec. 3 of all candidate groups with *minsup_c* =3, yielded max{$|g_i|$} = 11 and max support = 13. We multiply feature values of *GS* and *GSUP* by these numbers to get the actual counts. See equations (7) and (8) for details.

group in our database reviewed at least 10 products and is comprised of at least 5 members. Lastly, since $x_n < x_s\ \forall\ cp$ (due to the leftward positioning of the CBD curve for non-spam groups), spam groups obtain higher feature values than non-spam groups for each modeled behavior *f*.

Steep initial jumps: These indicate that very few groups obtain significant feature values before jump abscissa. For example, we find that there are very few spam groups with $GSUP < 0.25$, and for *GCS*, we find a majority (≈ 90%) of non-spam groups in our database have minuscule content similarity[6] among their reviews.

Gaps: CBD curves of non-spam groups are higher than those of spam groups and the gap (separation margin) refers to the relative discriminative potency. *GCS* has the maximum gap and next in order are *GSUP* and *GETF*. This result is not surprising as a group of people having a lot of content similarity in their reviews is highly suspicious of being spamming and hence *GCS* has good discriminative strength.

Lastly, we note again that the above statistics are inferred from the 2431 labeled groups by domain experts based on the data of [14] crawled in 2006. By no means do we claim that the results can be generalized across any group of random people who happen to review similar products together owing to similar interests.

7. MODELING RELATIONS

With the 8 group behavioral features separating spam and non-spam groups by a large margin and the labeled data from Sec. 4, the classic approach to detect spammer groups is to employ a supervised classification, regression, or learning to rank algorithm to classify or rank candidate groups. All these existing methods are based on a set of features to represent each instance (group in our case). However, as we indicated in the introduction section, this feature-based approach has some shortcomings for our task:

- They assume that training and testing instances are drawn independently and identically (iid) from some distribution. However, in our case, different groups (instances) can share members and may review some common products. Thus, our data does not follow the iid assumption because many instances are related, i.e., apart from group features, the spamicity of a group is also affected by the other groups sharing its members, the spamicity of the shared members, the extent to which the reviewed products are spammed, etc.
- Group features ($f_1 \ldots f_8$) only summarize (e.g., by max/avg) group behaviors. This clearly leads to loss of information because spamicity contributions from members are not considered at each individual member level, but are summarized (max/avg) to represent the whole group behavior. Due to different group sizes and complex relationships, it is not easy to design and include each individual member related features explicitly without some kind of summary.
- It is also difficult to design features which can consider the extent to which each product is spammed by groups. Although our focus is on detecting spammer groups, the underlying products being reviewed are clearly related.

Below, we propose a more effective model which can consider the inter-relationship among products, groups, and group members in computing group spamicity. Specifically, we model three binary relations: Group Spam–Products, Member Spam–Products, and Group Spam–Member Spam. The overall idea is as follows:

We first model the three binary relations to account for how each entity affects the other. We then draw inference of one entity from the other entity based on the corresponding binary relation. For example, using the Group Spam–Member Spam relation, we infer the spamicity of a group based on the spamicity of its individual members and vice-versa. Our ranking method called GSRank (Group Spam Rank) is then presented to tie all these inferences, which solves an eigenvalue problem by aligning the group vector to the dominant eigenvector. Before going to the details, we first define some notations used in the following sub-sections.

Let $P = \{p_i\}$, $G = \{g_j\}$, and $M = \{m_k\}$ be the set of all products, groups and members. Let $s(g_j)$ and $s(m_k)$ be the "spamicity" of g_j and m_k graded over [0, 1] respectively, and let $s(p_i)$ be the "extent to which p_i is spammed" also graded over [0, 1]. Values close to 1 signify high spamicity for groups and members and greater extent to which products are spammed. Additionally, let $V_P = [s(p_i)]_{|P|}$, $V_G = [s(g_j)]_{|G|}$, and $V_M = [s(m_k)]_{|M|}$ be the corresponding product, group and member score vectors.

7.1 Group Spam–Products Model

This model captures the relation among groups and products they target. The extent a product *p* is spammed by various groups is related to: (i) spam contribution to *p* by each group reviewing *p* and (ii) "spamicity" of each such group. Also, spam contribution by a group with high spamicity counts more. Similarly, the spamicity of a group is associated with (i) its own spam contribution to various products and (ii) the extent those products were spammed. To express this relation, we first compute the spam contribution to a product p_i by a group g_j. From Sec. 5, we have GTW_P (time window of group *g*'s activity over a product *p*), *D* (*g*'s deviation of ratings for *p*), *GTF* (early time frame of *g*'s spam infliction towards *p*), CS_G (*g*'s content similarity of reviews on *p*) and GSR_P (ratio of group's size for *p*). We note that these behaviors are "symmetric" in the sense that higher values indicate that *g*'s behavior on *p* is suspicious, and also indicate that spam contribution to *p* by *g* is high. Thus, the spam contribution by g_j to p_i can be expressed by the following function:

$$w_1(p_i,g_j) = \frac{1}{5}[GTW_P(g_j,p_i) + D(g_j,p_i) + GTF(g_j,p_i) + CS_G(g_j,p_i) + GSR_P(g_j,p_i)],$$

$$W_{PG} = [w_1(p_i, g_j)]_{|P|\times|G|} \tag{16}$$

$w_1(p_i, g_j) = 0$ when g_j did not review p_i. The sum captures the spam inflicted across various spamming dimensions and is normalized by 5 so that $w_1 \in [0, 1]$. For subsequent contribution functions too, summation and normalization are used for the same reason. W_{PG} denotes the corresponding contribution matrix.

Using (16), (17) computes the extent p_i is spammed by various groups. It sums the spam contribution by each group, $w_1(p_i, g_j)$, and weights it by the spamicity of that group, $s(g_j)$. Similarly, (18) updates the group's spamicity by summing its spam contribution on all products weighted by the extent those products were spammed. The relations can also be expressed as matrix equations.

$$s(p_i) = \sum_{j=1}^{|G|} w_1(p_i,g_j)s(g_j);\ \ V_P = W_{PG}V_G, \tag{17}$$

$$s(g_j) = \sum_{i=1}^{|P|} w_1(p_i,g_j)s(p_i);\ \ V_G = W_{PG}^T V_P \tag{18}$$

Since $s(g_j) \propto$ "spamicity of g_j", $s(p_i) \propto$ "extent to which p_i was spammed", and $w_1 \propto$ "degree of spam inflicted by g_j towards p_i", (17) and (18) employ a summation to compute $s(g_j)$ and $s(p_i)$. Further, as spam contribution by a group with higher "spamicity" is more damaging, the degree of spam contribution by a group is weighted by its spamicity in (17). Similarly, for (18), spam contribution, w_1 is weighted by $s(p_i)$ to account for the effective spamicity of the group. For subsequent models too, weighted summation is used for similar reasons. The matrix equation (17) also shows how the product vector can be inferred from the group vector using matrix W_{PG} and vice-versa using (18).

[6] Computed using LingPipe Java API available at http://alias-i.com/lingpipe

7.2 Member Spam–Product Model

Spam by a group on a product is basically spam by individuals in the group. A group feature can only summarize spam of members in the group over the set of products they reviewed. Here we consider spam contributions of all group members exclusively. Like w_1, we employ $w_2 \in [0, 1]$ to compute the spam contribution by a member m_k towards product p_i. We model w_2 as follows:

$$w_2(m_k, p_i) = \frac{1}{3}[IRD(m_k, p_i) + ICS(m_k, p_i) + IETF(m_k, p_i)],$$

$$W_{MP} = [w_2(m_k, p_i)]_{|M| \times |P|} \tag{19}$$

$w_2(m_k, p_i) = 0$ if m_k did not review p_i. Similar to (16), w_2 captures individual member spam contribution over the spam dimensions: *IRD* (individual rating deviation of m towards p), *ICS* (individual content similarity of reviews on p by m), and *IETF* (individual early time frame of spam infliction by m towards p). Similar to (18), we compute the spamicity of m_k by summing its spam contributions towards various products, w_2 weighted by $s(p_i)$ (20). And like (17), we update p_i to reflect the extent it was spammed by members. We sum the individual contribution of each member w_2, weighted by its spamicity (21).

$$s(m_k) = \sum_{i=1}^{|P|} w_2(m_k, p_i) s(p_i); \;\; V_M = W_{MP} V_P \tag{20}$$

$$s(p_i) = \sum_{k=1}^{|M|} w_2(m_k, p_i) s(m_k); \;\; V_P = W_{MP}^T V_M \tag{21}$$

7.3 Group Spam–Member Spam Model

Clearly, the spamicity of a group is related to the spamicity of its members and vice-versa. If a group consists of members with high spamicities, then the group's spam infliction is likely to be high. Similarly, a member involved in spam groups of high spamicity affects its own spamicity. We first compute the contribution of a member m ($\in g$) towards a group g. From Sec. 5, we see that the contribution is captured by *IMC* (degree of m's coupling in g), *GS* (size of g with which m worked), and *GSUP* (number of products towards which m worked with g). We model it as follows:

$$w_3(g_j, m_k) = \frac{1}{3}[IMC(g_j, m_k) + (1 - GS(g_j)) + GSUP(g_j)],$$

$$W_{GM} = [w_3(g_j, m_k)]_{|G| \times |M|} \tag{22}$$

$w_3(g_j, m_k) = 0$ when $m_k \notin g_j$. As *GS* is normalized over [0, 1], for large groups, the individual contribution of a member diminishes. Hence we use $1\text{-}GS(g_j)$ to compute w_3.

Using (22), (23) computes the spamicity of a group by summing up the spamicities of all its members, $s(m_k)$; each weighted by his contribution to the group, $w_3(g_j, m_k)$. Since groups can share members, (24) updates the spamicity of a member by summing up the spamicities of all groups it worked with, each weighted by its own contribution to that group.

$$s(g_j) = \sum_{k=1}^{|M|} w_3(g_j, m_k) s(m_k); \;\; V_G = W_{GM} V_M, \tag{23}$$

$$s(m_k) = \sum_{j=1}^{|G|} w_3(g_j, m_k) s(g_j); \;\; V_M = W_{GM}^T V_G. \tag{24}$$

8. GSRank: Ranking Group Spam

Using the relation models, each entity is inferred twice, once from each other entity. As the two inferences for each entity are conditioned on other two entities, they are thus complementary. For example, V_G is inferred once from V_P (18) and then from V_M (23). Both of these inferences complement each other because group spamicity is related to both its collective spam activities on products and also the spamicities of its members. This complementary connection is further explicitly shown in Lemma 1. Since the relations are circularly defined, to effectively rank the groups, GSRank uses the iterative algorithm below.

```
Algorithm: GSRank
Input: Weight matrices $W_{PG}$, $W_{MP}$, and $W_{GM}$
Output: Ranked list of candidate spam groups
1.  Initialize $V_G^0 \leftarrow [0.5]_{|G|}$; $t \leftarrow 1$;
2.  Iterate:
      i.   $V_P \leftarrow W_{PG} V_G^{(t-1)}$ ; $V_M \leftarrow W_{MP} V_P$ ;
      ii.  $V_G \leftarrow W_{GM} V_M$ ; $V_M \leftarrow W_{GM}^T V_G$ ;
      iii. $V_P \leftarrow W_{MP}^T V_M$ ; $V_G^{(t)} \leftarrow W_{PG}^T V_P$ ;
      iv.  $V_G^{(t)} \leftarrow V_G^{(t)} / \| V_G^{(t)} \|_1$ ;
    until $\| V_G^{(t)} - V_G^{(t-1)} \|_\infty < \delta$
3.  Output the ranked list of groups in descending order of $V_G^*$
```

In line 1, we first initialize all groups with spamicity of 0.5 over the spamicity scale [0, 1]. Next, we infer V_P from the current value of V_G; and then infer V_M from the so updated V_P (line 2-i). This completes the initial bootstrapping of vectors V_G, V_P, and V_M for the current iteration. Line 2-ii then draws inferences based on the Group Spam–Member Spam model. It first infers V_G from V_M because V_M contains the recent update from line 2-i and then infers V_M from so updated V_G. This ordering is used to guide the inference procedure across the iterations. Line 2-iii then updates V_P based on the Member Spam–Product model first, and defers the inference of V_G from so updated V_P based on the Group Spam–Product model until the last update so that V_G gets the most updated value for the next iteration. Finally, line 2-iv performs normalization to maintain an invariant state (discussed later). Thus, as the iterations progress, the fact that each entity affects the other is taken into account as the score vectors V_G, V_M, and V_P are updated via the inferences drawn from each relation model. The iterations progress until V_G converges to the stable V_G*. Since V_G contains the spamicity scores of all groups, line 3 outputs the final ordering of spam groups according to the spamicity scores of the stable vector V_G*. We now show the convergence of GSRank.

Lemma 1: *GSRank seeks to align V_G towards the dominant eigenvector and is an instance of an eigenvalue problem.*

Proof: From line 2 of GSRank, we have:

$$V_G = W_{GM} W_{MP} W_{PG} V_G^{(t-1)} \tag{25}$$

$$V_G^{(t)} = W_{PG}^T W_{MP}^T W_{GM}^T V_G \tag{26}$$

Substituting (25) in (26) and letting $Z = W_{GM} W_{MP} W_{PG}$ we get:

$$V_G^{(t)} = (Z^T Z) V_G^{(t-1)} \tag{27}$$

Clearly, this is an instance of *power iteration* for the *eigenvalue problem* of computing the group vector V_G as the eigenvector of $Z^T Z$ corresponding to the *dominant* eigenvalue [12]. ■

From (25), (26), and (27), we can see how the two inferences for each entity are linked. For example, spamicity of groups V_G based on the spamicity of its members, V_M (line 2-ii) and based on the collective spam behavior on products, V_P (line 2-iii) are both linked and accounted for by the product of matrices W_{GM} and W_{PG}^T in (27). Similar connections exist for V_M and V_P when inferred from other entities. Thus, all model inferences are combined and encoded in the final iterative inference of V_G in (27).

Theorem 1: *GSRank converges*

Proof: As GSRank seeks to align V_G towards the dominant eigenvector, to show convergence, it is sufficient to show that the stable vector V_G* is aligned to the dominant eigenvector after a certain number of iterations. Let $A = Z^T Z$. From (27) and line 2-iv of GSRank, we get:

$V_G^{(t)} = \dfrac{AV_G^{(t-1)}}{\| AV_G^{(t-1)} \|_1}$, which when applied recursively gives:

$$V_G^{(t)} = \frac{A^t V_G^{(0)}}{\prod_{j=0}^{t-1} \| AV_G^{(j)} \|_1} \tag{28}$$

We note that A is a square matrix of order $|G|$. Assuming A to be *diagonalizable* [7], $V_G^{(0)}$ can be expressed as a convex combination of the $|G|$ eigenvectors of A [12].

$$V_G^{(0)} = \sum_{i=1}^{|G|} \alpha_i v_i \tag{29}$$

Also let λ_i denote the eigenvalue corresponding to the eigenvector v_i with λ_1, v_1 being the dominant eigenvalue-vector pair of A. Then, using (28) and (29) we obtain:

$$V_G^{(t)} = \frac{1}{\prod_{j=0}^{t-1} \| AV_G^{(j)} \|_1} \sum_{i=1}^{|G|} \lambda_i^t \alpha_i v_i = \frac{1}{\prod_{j=0}^{t-1} \| AV_G^{(j)} \|_1} \lambda_1^t \left[\alpha_1 v_1 + \sum_{i=2}^{|G|} \left(\frac{\lambda_i}{\lambda_1} \right)^t \alpha_i v_i \right]$$

since λ_1 is dominant, $|\lambda_i / \lambda_1| < 1$, $\forall\ i > 1$. Thus, for large t, $\sum_{i=2}^{|G|} \left(\frac{\lambda_i}{\lambda_1} \right)^t \alpha_i v_i \approx 0$ and the stable vector $V_G^* \propto v_1$. ■

Normalization: Before each iteration, V_G is normalized (line 2-iv). L_1 normalization maintains an invariant state between two consecutive iterations so that convergence is observed as a very small change in the value of V_G [19]. We employ L_∞ (as it is a max) norm of the difference of V_G over consecutive iterations to be less than δ = 0.001 as our terminating condition[8]. Normalization also prevents any overflow that might occur due to the geometric growth of components during each iteration [12].

Complexity: At each iteration, GSRank requires the multiplication of A with V_G, so it takes $O(t|E|)$, where $|E|$ is the number of non-zero elements in A and t is the total number of iterations. In terms of P, G, and M, it takes $O(t(|G|(|M|+|P|) + |M||P|))$ which is linear in the number of candidate groups discovered by FIM. The actual computation, however, is quite fast since the matrices W_{PG}, W_{MP}, W_{GM} are quite sparse due to the power law distribution followed by reviews [14]. Furthermore, GSRank being an instance of power iteration, it can efficiently deal with large and sparse matrices as it does not compute a matrix decomposition [12].

9. EXPERIMENTAL EVALUATION

We now evaluate the proposed GSRank method. We use the 2431 groups described in Sec. 3. We first split 2431 groups into the development set, D with 431 groups (randomly sampled) for parameter estimation and the validation set, V with 2000 groups for evaluation. All evaluation metrics are averaged over 10-fold cross validation (CV) on V. Below we first describe parameter estimation and then ranking and classification experiments.

9.1 Parameter Estimation

Our proposed behavioral model has two parameters τ and β, which have to be estimated. τ is the parameter of *GTW*, i.e., the time interval beyond which members in a group are not likely to be working in collusion. β is the parameter of *GETF* which denotes the time interval beyond which reviews posted are not considered to be "early" anymore (Sec. 5.1). For this estimation, we again use the classification setting. The estimated parameters actually work well in general as we will see in the next two subsections.

Let $\boldsymbol{\theta}$ denote the two parameters. We learn $\boldsymbol{\theta}$ using a greedy hill climbing search to maximize the log likelihood of the set D:

$$\vec{\theta}_{estimated} = \arg\max_{\vec{\theta}} \sum_j \log P(g_j = spam \mid \vec{\theta}) \tag{30}$$

where $g_j \in D$. To compute $P(g_j = spam)$ we treat $g_j = [X_1 \ldots X_8]$ as a vector where each X_i takes the values attained by the i^{th} feature f_i. As each feature models a different behavior, we can assume the features to be independent and express $P(g_k{=}spam) = \Pi P(X_i^{g_k} = spam)$. To compute $P(X_i^{g_k} {=}spam)$, i.e., $P(X_i{=}spam)$ for g_k, we discretize the range of values obtained by f_i into a set of k intervals $\{{}^1f_i, \ldots, {}^kf_i\}$ such that $\cup\, {}^kf_i = [0, 1]$. $P(X_i = spam)$ then reduces to $P({}^kf_i = spam)$ whenever X_i lies in the interval ${}^kf_i$. And $P({}^kf_i = spam)$ is simply the fraction of spam groups whose value of f_i lies in interval ${}^kf_i$. We used the popular discretization algorithm in [9] to divide the value range of each feature into intervals. To bootstrap the hill climbing search, we used initial seeds τ_0 = 2 months and β_0 = 6 months. The final estimated values were: $\tau = 2.87$, $\beta = 8.86$.

9.2 Ranking Experiments

To compare group spam ranking of GSRank, we use regression and learning to rank [26] as our baselines. Regression is a natural choice because the spamicity score of each group from the judges is a real value over [0, 1] (Sec. 4). The problem of ranking spammer groups can be seen as optimizing the spamicity of each group as a regression target. That is, the learned function predicts the spamicity score of each test group. The test groups can then be ranked based on the values. For this set of experiments, we use the support vector regression (SVR) system in *SVM^light^* [16].

Learning to rank is our second baseline approach. Given the training samples $x_1 \ldots x_n$, a learning to rank system takes as input k different rankings $y_1 \ldots y_k$ of the samples generated by k queries $q_1 \ldots q_k$. Each ranking y_i is a permutation of $x_1 \ldots x_n$ based on q_i. The learning algorithm learns a ranking model h which is then used to rank the test samples $u_1 \ldots u_m$ based on a query q. In our case of ranking spam groups, the desired information need q denotes the question: *Are these group spam*? To prepare training rankings, we treat each feature f as a ranking function (i.e. the groups are ranked in descending order of values attained by each $f_1 \ldots f_8$). This generates 8 training ranks. A learning algorithm then learns the optimal ranking function. Given no other knowledge, this is a reasonable approach since $f_1 \ldots f_8$ are strongly correlated with spam groups (Sec. 6). The rank produced by each feature is thus based on a certain spamicity dimension. None of the training ranks may be optimal. A learning to rank method basically learns an optimal ranking function using the combination of $f_1 \ldots f_8$. Each group is vectorized with (represented with a vector of) the 8 group spam features. We ran two widely used learning to rank algorithms [26]: SVMRank [17] and RankBoost [11]. For SVMRank, we used the system in [17]. RankBoost was from RankLib[9]. For both systems, their default parameters were applied. We also experimented with RankNet [3] in RankLib, but its results are significantly poorer on our data. Thus, its results are not included.

In addition, we also experimented with the following baselines:

- Group Spam Feature Sum (GSFSum): As each group feature $f_1 \ldots f_8$ measures spam behavior on a specific spam dimension, an obvious baseline (although naïve) is to rank the groups in descending order of the sum of all feature values.
- Helpfulness Score (HS): In many review sites, readers can provide helpfulness feedback to each review. It is reasonable to assume that spam reviews should get less helpfulness feedback. HS uses the mean helpfulness score (percentage of people who found a review helpful) of reviews of each group to rank groups in ascending order of the scores.

[7] A matrix $A_{n \times n}$ over the field F is diagonalizable *iff* the sum of the dimensions of its eigenspaces is equal to n. This can be shown to be equivalent to A being of full rank with n linearly independent eigenvectors. The proof remains equally valid when A is *defective* (not diagonalizable), i.e., it has $k < |G|$ linearly independent eigenvectors and hence the summation in (29) goes up to k. Convergence of GSRank is still guaranteed because of the following argument:

$$\lim_{t\to\infty} \sum_{i=2}^{k} \left(\frac{\lambda_i}{\lambda_1} \right)^t \alpha_i v_i = 0 \text{ ,whenever } \lim_{t\to\infty} \sum_{i=2}^{|G|} \left(\frac{\lambda_i}{\lambda_1} \right)^t \alpha_i v_i = 0 \text{ as } k < |G|.$$

[8] Using this threshold our implementation converges in 96 iterations.

[9] http://www.cs.umass.edu/~vdang/ranklib.html

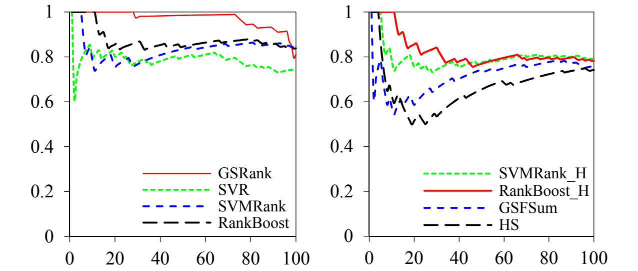
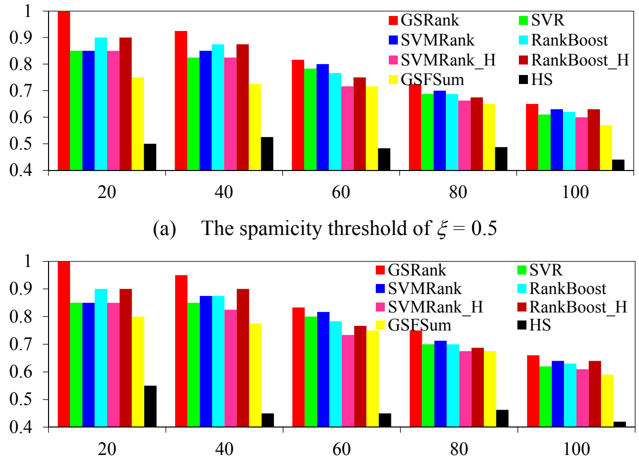


Figure 5: NDCG@*k* comparisons (NDCG for top 100 rank positions)



(a) The spamicity threshold of $\xi = 0.5$

(b) The spamicity threshold of $\xi = 0.7$

Figure 6: Precision @ *n* = 20, 40, 60, 80, 100 rank positions.

All the improvements of GSRank over other methods are statistically significant at the confidence level of 95% based on paired *t*-test.

- Heuristic training rankings (H): In our preliminary study [29], three heuristic rankings using feature mixtures were proposed to generate the training ranks for learning to rank methods. We list them briefly here. For details, please see [29].

  $h_1(g) : G \rightarrow \mathbf{R}^+,\ h_1(g) = GCS(g) + GMCS(g)$
  $h_2(g) : G \rightarrow \mathbf{R}^+,\ h_2(g) = GS(g) + GSUP(g) + GTW(g)$
  $h_3(g) : G \rightarrow \mathbf{R}^+,\ h_3(g) = GSR(g) + GETF(g) + GD(g)$

  Using these three functions to generate the training ranks, we ran the learning to rank methods. We denote these methods and their results with RankBoost_H, and SVMRank_H.

To compare rankings we first use Normalized Discounted Cumulative Gain (NDCG) as our evaluation metric. NDCG is commonly used to evaluate retrieval algorithms with respect to an ideal ranking based on relevance. It rewards rankings with the most relevant results at the top positions [26], which is also our objective, i.e., to rank those groups with the highest spamicities at the top. The spamicity score for each group computed from judges (Sec. 4) thus can be regarded as the relevance score to generate the "ideal" spam ranking. Let $R(m)$ be the relevance score of the m^{th} ranked item. NDCG @ k is defined as:

$$NDCG@k = \frac{DCG@k}{Z_k}; \qquad DCG@k = \sum_{m=1}^{k} \frac{2^{R(m)} - 1}{\log_2(1+m)}, \tag{31}$$

where Z_k is the discounted cumulative gain (DCG) of the ideal ranking of the top k results. We report NDCG scores at various top positions up to 100 in Figure 5. In our case, $R(m)$ refers to the score(g_m) computed by each ranking algorithm (normalization was applied if needed), where g is the group ranked at position m. To compute Z_k = DCG@k for the ideal ranking, we use the spamicity(g_m) from our expert judges.

From Figure 5, we observe that GSRank performs the best at all top rank positions except at the bottom, which are unimportant because they are most likely to be non-spam (since in each fold of cross validation, the test set has only 200 groups and out of the 200 there are at most 38% spam groups; see Table 1 below). Paired *t*-tests for rank positions k = 20, 40, 60, 80 show that all the improvements of GSRank over other methods are significant at the confidence level of 95%. Although regression is suitable for the task, it did not perform as well as RankBoost and SVMRank. RankBoost_H and SVMRank_H behave similarly to RankBoost and SVMRank, but performed slightly poorer. GSFSum fared mediocrely as ranking based on summing all feature values is unable to balance the weights of features because not all features are equal in discriminative strength. HS performs poorly, which reveals that while many genuine reviews may not be helpful, spam reviews can be quite helpful (deceptive). Thus, helpfulness scores are not good for differentiating spam and non-spam groups.

Since in many applications, the user wants to investigate a certain number of highly likely spam groups and NDCG does not give any guidance on how many are very likely to be spam, we thus also use *precision @ n* to evaluate the rankings. In this case, we need to know which test groups are spam and non-spam. We can use a threshold ξ on the spamicity to decide that, which can reflect the user's strictness for spam. Since in different applications the user may want to use different thresholds, we use two thresholds in our experiments, $\xi = 0.5$ and $\xi = 0.7$. That is, if the spamicity value is $\geq \xi$, the group is regarded as *spam*, otherwise *non-spam*. These thresholds give us the following data distributions:

	$\xi = 0.5$	$\xi = 0.7$
Spam	38%	29%
Non-spam	62%	71%

Table 1: Data distributions for the two spamicity thresholds ξ

Figure 6 (a) and (b) show the precisions @ 20, 40, 60, 80, and 100 top rank positions for $\xi = 0.5$ and $\xi = 0.7$ respectively. We can see that GSRank consistently outperforms all existing methods. RankBoost is the strongest among the existing methods.

9.3 Classification Experiments

If a spamicity threshold is applied to decide spam and non-spam groups, supervised classification can also be applied. Using the thresholds of $\xi = 0.5$ and 0.7, we have the labeled data in Table 1. We use SVM in SVM^{light} [16] (with linear kernel) and Logistic Regression (LR) in WEKA (www.cs.waikato.ac.nz/ml/weka) as the learning algorithms. The commonly used measure AUC (Area Under the ROC Curve) is employed for classification evaluation.

Next we discuss the features that we consider in learning:

- Group Spam Features (GSF) $f_1 \ldots f_8$: These are the proposed eight (8) group features presented in Sec. 5.1.
- Individual Spammer Features (ISF): A set of features for detecting individual spammers was reported in [24]. Using these features, we represented each group with their average values of all the members of each group. We want to see whether such individual spammer features are also effective for groups. Note that these features cover those in Sec. 5.2.
- Linguistic Features of reviews (LF): In [31], word and POS (part-of-speech) n-gram features were shown to be effective for detecting individual fake reviews. Here, we want to see whether such features are also effective for spam groups. For each group, we merged its reviews into one document and represented it with these linguistic features.

Table 2 (a) and (b) show the AUC values of the two classification algorithms for different feature settings using 10-fold cross

Feature Settings	SVM	LR	SVR	SVM Rank	Rank Boost	SVM Rank_H	Rank Boost_H	GS Rank
GSF	0.81	0.77	0.83	0.83	0.85	0.81	0.83	**0.93**
ISF	0.67	0.67	0.71	0.70	0.74	0.68	0.72	
LF	0.65	0.62	0.63	0.67	0.72	0.64	0.71	
GSF + ISF + LF	0.84	0.81	0.85	0.84	**0.86**	0.83	0.85	

(a) The spamicity threshold of $\xi = 0.5$

Feature Settings	SVM	LR	SVR	SVM Rank	Rank Boost	SVM Rank_H	Rank Boost_H	GS Rank
GSF	0.83	0.79	0.84	0.85	0.87	0.83	0.85	**0.95**
ISF	0.68	0.68	0.73	0.71	0.75	0.70	0.74	
LF	0.66	0.62	0.67	0.69	0.74	0.68	0.73	
GSF + ISF + LF	0.86	0.83	0.86	0.86	**0.88**	0.84	0.86	

(b) The spamicity threshold of $\xi = 0.7$

Table 2: AUC results of different algorithms and feature sets. All the improvements of GSRank over other methods are statistically significant at the confidence level of 95% based on paired *t*-test.

validation for $\xi = 0.5$ and $\xi = 0.7$ respectively. It also includes the ranking algorithms in Sec. 9.2 as we can also compute their AUC values given the spam labels in the test data. Note that the relation-based model of GSRank could not use other features than GSF features and the features in Sec. 5.2 (not shown in Table 2). Here, again we observe that GSRank is significantly better than all other algorithms (with the 95% confidence level using paired *t*-test). RankBoost again performed the best among the existing methods. Individual spam features (ISF) performed poorly. This is understandable because they cannot represent group behaviors well. Linguistic features (LF) fared poorly too. We believe it is because content-based features are more useful if all reviews are about the same type of products. The language used in fake and genuine reviews can have some subtle differences. However, reviewers in a group can review different types of products. Even if there are some linguistic differences among spam and non-spam reviews, the features become quite sparse and less effective due to a large number of product types and not so many groups. We also see that combining all features (Table 2, last row in each table) improves AUC slightly. RankBoost achieved AUC = 0.86 (ξ = 0.5) and 0.88 (ξ = 0.7), which are still significantly lower than AUC = 0.93 (ξ = 0.5) and 0.95 (ξ = 0.7) for GSRank respectively. Finally, we observe that the results for $\xi = 0.7$ are slightly better than those for $\xi = 0.5$. This is because with the threshold $\xi = 0.7$, the spam and non-spam groups are well separated (see Table 1).

In summary, we conclude that GSRank outperforms all baseline methods, including regression, learning to rank and classification. This is important considering that GSRank is an unsupervised method. This also shows that the relation-based model used in GSRank is indeed effective in detecting opinion spammer groups.

10. CONCLUSIONS

This paper proposed to detect group spammers in product reviews. The proposed method first used frequent itemset mining to find a set of candidate groups, from which a labeled set of spammer groups was produced. We found that although labeling individual fake reviews or reviewers is hard, labeling groups is considerably easier. We then proposed several behavior features derived from collusion among fake reviewers. A novel relation-based model, called GSRank, was presented which can consider relationships among groups, individual reviewers, and products they reviewed to detect spammer groups. This model is very different from the traditional supervised learning approach to spam detection. Experimental results showed that GSRank significantly outperformed the state-of-the-art supervised classification, regression, and learning to rank algorithms.

11. ACKNOWLEDGMENT

This work was partially supported by a Google Faculty Research Award.

12. REFERENCES

[1] Agrawal, R. and Srikant, R. Fast algorithms for mining association rules. *VLDB*. 1994.

[2] Benevenuto, F., Rodrigues, T., Almeida, V., Almeida, J, Gonvalves, M. A. Detecting spammers and content promoters in online video social networks. *SIGIR*. 2009.

[3] Burges , C., Shaked, T., Renshaw, E. Lazier, A. Deeds, M., Hamilton, N. Hullender., G. Learning to rank using gradient descent. *ICML*. 2005.

[4] Castillo, C., Davison, B. Adversarial Web Search Foundations and Trends in Information Retrieval, 5, 2010.

[5] Castillo, C., Donato, D., Becchetti, L., Boldi, P., Leonardi, S., Santini, M., and Vigna, S. 2006. A reference collection for web spam. *SIGIR Forum* 40, 2, 11–24, S. 2006.

[6] Chirita, P.A., Diederich, J., and Nejdl, W. MailRank: using ranking for spam detection. *CIKM*. 2005.

[7] Douceur, J. R. The sybil attack. *IPTPS Workshop*. 2002.

[8] Eagle, N. and Pentland, A. Reality Mining: Sensing Complex Social Systems. Personal and Ubiquitous Computing. 2005.

[9] Fayyad, U. M. and Irani, K. B. Multi-interval discretization of continuous-valued attributes for classification learning. *IJCAI*. 1993.

[10] Fleiss, J. L. Measuring nominal scale agreement among many raters. *Psychological Bulletin*, 76(5), pp. 378–382, 1971.

[11] Freund, Y., Iyer, R., Schapire, R. and Singer, Y. An efficient boosting algorithm for combining preference. *JMLR*. 2003.

[12] Heath, M. T., Scientific Computing: An Introductory Survey. McGrawHill, New York. Second edition. 2002.

[13] Hsu, W., Dutta, D., Helmy, A. Mining Behavioral Groups in Large Wireless LANs. *MobiCom*. 2007.

[14] Jindal, N. and Liu, B. Opinion spam and analysis. *WSDM*. 2008.

[15] Jindal, N., Liu, B. and Lim, E. P. Finding Unusual Review Patterns Using Unexpected Rules. *CIKM*. 2010.

[16] Joachims, T. Making large-scale support vector machine learning practical. *Advances in Kernel Methods*. *MIT Press*. 1999.

[17] Joachims, T. Optimizing Search Engines Using Clickthrough Data. *KDD*. 2002.

[18] Kim, S.M., Pantel, P., Chklovski, T. and Pennacchiotti, M. Automatically assessing review helpfulness. *EMNLP*. 2006.

[19] Kleinberg, J. M. Authoritative sources in a hyperlinked environment. ACM-SIAM *SODA*, 1998.

[20] Kolari, P., Java, A., Finin, T., Oates, T., Joshi, A. Detecting Spam Blogs: A Machine Learning Approach. *AAAI*. 2006.

[21] Koutrika, G., Effendi, F. A., Gyöngyi, Z., Heymann, P., and H. Garcia-Molina. Combating spam in tagging systems. *AIRWeb*. 2007.

[22] Landis, J. R. and Koch, G. G. The measurement of observer agreement for categorical data. *Biometrics*, 33, 159-174, 1977.

[23] Li, F., Huang, M., Yang, Y. and Zhu, X. Learning to identify review Spam. *IJCAI*. 2011.

[24] Lim, E. Nguyen, V. A., Jindal, N., Liu, B., and Lauw, H. Detecting Product Review Spammers Using Rating Behavior. *CIKM*. 2010.

[25] Liu, J., Cao, Y., Lin, C., Huang, Zhou, M. Low-quality product review detection in opinion summarization. *EMNLP*, 2007.

[26] Liu, T-Y. Learning to Rank for Information Retrieval. *Foundations and Trends in Information Retrieval* 3(3): 225-331. 2009.

[27] Markines, B., Cattuto, C., and Menczer, F. Social spam detection. *AIRWeb*. 2009.

[28] Martinez-Romo, J. and Araujo, A. Web Spam Identification Through Language Model Analysis. *AIRWeb*. 2009.

[29] Mukherjee, A., Liu, B., Wang, J., Glance, N., Jindal, N. Detecting Group Review Spam. *WWW*. 2011. (Poster paper)

[30] Ntoulas, A., Najork, M., Manasse M., Fetterly, D. Detecting Spam Web Pages through Content Analysis. *WWW'2006*.

[31] Ott, M., Choi, Y., Cardie, C. Hancock, J. Finding Deceptive Opinion Spam by Any Stretch of the Imagination. *ACL*. 2011.

[32] Wang, G., Xie, S., Liu, B., and Yu, P. S. Review Graph based Online Store Review Spammer Detection. *ICDM*. 2011.

[33] Wang, Y. Ma, M. Niu, Y. and Chen, H. Spam Double-Funnel: Connecting Web Spammers with Advertisers. *WWW*'2007.

[34] Wu, G., Greene, D., Smyth, B. and Cunningham, P. 2010. Distortion as a validation criterion in the identification of suspicious reviews. Technical report, UCD-CSI-2010-04, University College Dublin.

[35] Wu, B., Goel V. & Davison, B. D. Topical TrustRank: using topicality to combat Web spam. *WWW*. 2006.

[36] Yan, F., Jiang, J., Lu, Y., Luo, Q., Zhang, M. Community discovery based on social actors' interests & social relationships. SKG. 2008.

[37] Zhang, Z. and Varadarajan, B. Utility scoring of product reviews. *CIKM*. 2006.