Towards a Tricksy Group Shilling Attack Model against Recommender Systems

Youquan Wang[1], Zhiang Wu[2,⋆], Jie Cao[2,1], and Changjian Fang[2]

[1] College of Computer Science and Engineering,
Nanjing University of Science and Technology, Nanjing, China
[2] Jiangsu Provincial Key Laboratory of E-Business,
Nanjing University of Finance and Economics, Nanjing, China
{youq.wang,zawuster}@gmail.com,
{caojie690929,jselab1999}@163.com

Abstract. The robustness of recommender systems has drawn recently more and more attention of both industry and academia. Although a multitude of studies have been devoted to shilling attack modeling and detection, few of them focus on *group* shilling attack. The attackers in a shilling group work together to manipulate the output of the recommender system. Meanwhile, since the rating profiles in a shilling group are carefully designed, it is hard to detect them by traditional methods. This paper presents a generative model to create shilling group in which every pair of attackers has high diversity. In particular, both strict and loose versions of group shilling attack generation algorithm are proposed. Experimental results on MovieLens data set demonstrate that the shilling group generated by the our model can not only exert large negative effect to recommender systems, but also avoid the detection by the traditional methods.

Keywords: shilling attack, group shilling attack, recommender systems.

1 Introduction

The security and robustness of recommender systems has become a hot topic in recent years, in the context of *profile injection* or *shilling attacks* [1, 2]. As the prevalence of online shopping, shilling attackers have a natural profit incentive to promote their own products (or suppress their competitors' products) by creating biased online comments and/or extreme ratings. Therefore, shilling attacks are seriously threatening the sound development of e-commerce.

A large body of work has been devoted to discuss the shilling attack generative models [3, 4, 5], define classification metrics for genuine users shilling attackers [3, 6, 7, 8], and design the shilling attack detectors [6, 7, 9, 10, 11, 12]. The existing detectors are fairly effective to the shilling attackers working separately. However, as the development of the attack tricks, shilling attackers are not working separately yet today, but working together to commit swarm and

⋆ Corresponding author.

S. Zhou, S. Zhang, and G. Karypis (Eds.): ADMA 2012, LNAI 7713, pp. 675–688, 2012.

Table 1. Generative models of shilling attacks

Attack models	Push	Nuke
Random attack	$I_s = \emptyset$; $I_F = r_{ran}$; $i_t = r_{max}$	$I_s = \emptyset$; $I_F = r_{ran}$; $i_t = r_{min}$
Average attack	$I_s = \emptyset$; $I_F = r_{avg}$; $i_t = r_{max}$	$I_s = \emptyset$; $I_F = r_{avg}$; $i_t = r_{min}$
Segmented attack	$I_s = r_{max}$; $I_F = r_{min}$; $i_t = r_{max}$	$I_s = r_{min}$; $I_F = r_{max}$; $i_t = r_{min}$
Bandwagon attack	$I_s = r_{max}$; $I_F = r_{max}$; $i_t = r_{max}$	$I_s = r_{min}$; $I_F = r_{ran}$; $i_t = r_{min}$

Note: (1) r_{ran}: random ratings; (2) r_{avg}: average ratings;
(3) I_s in segmented attack: a set of similar items of the target item;
(4) I_s in bandwagon attack: a set of frequently rated items.

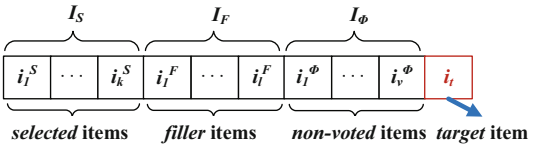


Fig. 1. The illustration of a shilling profile

massive attacks after a premeditated planning. This new type of shilling attack was coined as "group shilling attacks" [13]. Every attacker in the shilling group has carefully designed its ratings to conceal the malicious intention. An individual attacker may not successfully promote (or suppress) the target products, but when they work together, they will attain the attack goal.

This paper focuses on a generation method to construct the effective group shilling attacks which can not only avoid the detection of the existing detectors but also successfully modify the recommendation output of the system. Based on the observation that the *low diversity* attackers can be easily recognized by the traditional detectors [14], we propose two versions group shilling generative models to create *highest diversity* and *high diversity* profiles, respectively.

The remainder of this paper is organized as follows. In Section 2, we briefly introduce background knowledge about shilling attack, define the problem and summarize the characteristics of group shilling attack. In Section 3, we present two versions of group shilling attack generation algorithm. Section 4 shows the experimental results. We finally present the related work and conclude our work in Section 5 and Section 6, respectively.

2 Preliminaries

Shilling attackers can be classified as *push* and *nuke* attacks according to their intent–making a target item more likely (push) or less likely (nuke) to be recommended [6]. The rating records of a user on various items construct the profile of that user. The profile of a shilling attacker (or a *shilling profile* for short) usually consists of the ratings on four types of items: target item, filler items, selected items, and non-voted items, as show in Fig. 1. Target item often has the highest rating in a push attack, or lowest rating in a nuke attack. Filler items can make

a shilling profile look normal and yield profound impact against a recommender system. Selected items are often used to make friends with as many genuine users as possible. Finally, non-voted items are the remaining unrated items.

Four generative models, i.e. random, average, segment, and bandwagon, are often used to generate shilling profiles by carefully rating the four types of items, as illustrated in Table 1. In the literature [1, 3, 4, 7, 8, 10, 11], the random rating r_{ran} is often generated by Gaussian distribution. For instance, in the 5-grade marking data set `MoiveLens`, r_{ran} is assumed to obey a Gaussian distribution with $\mu = 3.6$ and $\sigma = 1.1$ [1].

2.1 Problem Definition

We consider a rating matrix of which the row vectors represent a set of user profiles and every element is the rating that a user gave for a item. Let $\mathbb{B} = \{\mathbf{B}_1, \cdots, \mathbf{B}_l\}$ denote l groups of shilling attackers. Actually, $\mathbf{B}_i (1 \leq i \leq l)$ is a $n_i \times m$ rating matrix where n_i is size of $\mathbf{B}_i$ and m is the number of items. If let b_j denote the j-th row vector of $\mathbf{B}_i$, we have $\mathbf{B}_i = \{b_1, \cdots, b_{n_i}\}$.

From the perspective of the attacker, the best attack against a system is one that yields the biggest impact for the least amount of effort [3]. The descriptive dimensions of single shilling attacks have been discussed in [3], while we are primarily interested in the following dimensions for group shilling attacks:

- Attack intent: There are usually two intents: *push* and *nuke*. The group shilling attackers aim to make the target items more likely (push) or less likely (nuke) to be recommended.
- The number of target items T_i: All shillers in a group work together to promote or demote a set of target items rather than one item. For example, a shilling group may works for a brand consisting of many products.
- The number of groups l: Many shilling groups may co-exist in a recommender system.
- The size of group shilling attacks n_i: Big size of shilling group leads to the big cost of the creation of the group, because on-line registrations require human intervention.
- Profile size $|b_j|$: The number of ratings assigned by an attacker in a shilling group is equal to *filler size* in the single shilling attack.

It is interesting to note that as the increase of T_i, l and $|b_j|$, the group shilling attacks may threaten the recommender systems more seriously, however, the cost for generating the attack profiles is also increasing. Therefore, the tricksy attackers should balance the power for damaging the recommender systems and the cost required for generating the attack profiles.

2.2 Characteristics of Group Shilling Attack

Before modeling the group shilling attack, we summarize two important characteristics to support the proposed group shilling attack model in Section 2. The two fundamental properties are presented as following:

Property 1. Effective group shilling attacks can push (or nuke) target items successfully and affect the recommender system as much as possible.

Property 2. Every individual shiller in group attacks looks like normal users and should not be discovered by detector designed for single shilling attack.

The property 1 is the original target of shilling attackers. Group shilling attack is not the combination of many single shilling profiles generated by attack models, since the group shilling attacks can be easily filtered by the traditional detectors. Profiles in group shilling attacks are quite different from traditional attack profiles, for they have well designed filling ratings to conceal their malicious intentions. So, we present the property 2. The generation model of group shilling attacks should not only make every profiles in the group looks like normal users to avoid the detection by traditional algorithms, but also keep a grave threat to recommender systems.

3 The Generation Model of Group Shilling Attacks

Shilling profiles created by above-mentioned four models, i.e. random, average, segment, and bandwagon, have high similarity. These low-diversity attackers can be easily detected by the existing detection algorithm such as C4.5-, PCA- and clustering-based detectors [3, 9, 10]. In this section, we target at proposing a method to generate high-diversity shilling profiles that are used to create group shilling attacks. The idea behind the method is that every pair of shilling attackers in the group keeps low similarity to avoid the detection by the existing methods designed for single shilling attack. Specially, we propose two versions for the generative model of group shilling attacks. The first is a strict version denoted as $GSAGen_s$, which can guarantee the PCC similarity of each pair of profiles in a group be -1. However, the second version $GSAGen_l$ employs a loose condition which guarantee the PCC similarity of each pair of profiles in a group be -1 or 0.

3.1 $GSAGen_s$: A Strict Version

Our group shilling attack generative model takes a set of shilling profiles created by attack models, i.e. random or average, as the input. Then, we aims to generate a group of shilling attackers $\mathbf{B}_i$ with high diversity satisfying the following:

Definition 1 (The Strict Condition of Group Shilling Attack). *Let* $\mathbf{B}_i = \{b_1, \cdots, b_{n_i}\}$ *be a group of shilling attackers. We say* $\mathbf{B}_i$ *satisfies the strict condition iff the following three conditions are satisfied simultaneously: (1)* $\forall u \neq v, b_u \cap b_v \neq \emptyset$*; (2)* $\forall u \neq v \neq t, b_u \cap b_v \cap b_t = \emptyset$*; (3)* $\forall u \neq v, PCC_{uv} = -1$.

In Definition 1, PCC_{uv} is the Pearson Correlation Coefficient (PCC) between b_u and b_v. As is known, PCC is in [-1,1] and $PCC_{uv} \leq 0$ indicates that b_u and b_v are not similar, and thus $PCC_{uv} = -1$ implies that b_u and b_v have the highest diversity. The procedure of $GSAGen_s$ as shown in Algorithm 1 generates a group

of shilling attackers satisfying Definition 1 based on a set of shilling profiles. In line 2, $\mathbf{B}_i$ is initialized with a shilling profile a_1. In lines 3-17, every profile $a_u \in \mathbf{A}_i$ is examined, and b_u is constructed in terms of a_u and current rating records in $\mathbf{B}_i$. In lines 4-15, every rated item of a_u is traversed and if the current item is not be rated by all attackers in $\mathbf{B}_i$, the rating value of j-th item by a_u is assigned to b_u (see lines 12-14). Lines 5-11 describe the case that the j-th item is rated by a_u and at least one attacker in $\mathbf{B}_i$. If the j-th item is rated by only one attacker in $\mathbf{B}_i$, b_{uj} is set to $2\overline{b_{v,\kappa}} - b_{vj}$. Note that $\overline{b_{v,\kappa}}$ is the average value on the intersection of a_u and b_v. If there exists one item rated by a_u that has been rated by more than one attacker in $\mathbf{B}_i$, a_u cannot be utilized to generate b_u and thus $GSAGen_s$ jumps to examine the next shilling profile a_{u+1}(see lines 9-11).

As can be seen from the procedure of $GSAGen_s$, we can easily observe that $\forall u \neq v, b_u \cap b_v \neq \emptyset$ and $\forall u \neq v \neq t, b_u \cap b_v \cap b_t = \emptyset$. In the following theorem, we will prove that $\forall u \neq v, PCC_{uv} = -1$.

Theorem 1. $\forall b_u, b_v \in \mathbf{B}_i$, $PCC_{uv} = -1$.

PROOF:Let $\kappa = b_u \cap b_v$ and $PCC_{uv} = \frac{\sum_{j\in\kappa}(b_{uj}-\overline{b_{u,\kappa}})(b_{vj}-\overline{b_{v,\kappa}})}{\sqrt{\sum_{j\in\kappa}(b_{uj}-\overline{b_{u,\kappa}})^2}\sqrt{\sum_{j\in\kappa}(b_{vj}-\overline{b_{v,\kappa}})^2}}$. $\because b_{uj} = 2\overline{b_{v,\kappa}} - b_{vj}$, and $\overline{b_{u,\kappa}} = \overline{b_{v,\kappa}}$,
$\therefore b_{uj} - \overline{b_{u,\kappa}} = -(b_{vj} - \overline{b_{v,\kappa}})$,
$\therefore PCC_{uv} = -1$, which completes the proof. □

From Theorem 1, the group of shilling attack generated by $GSAGen_s$ possesses the strict condition as shown in Definition 1. So, every pair of shilling profiles in the group has the *highest diversity*, i.e.$PCC = -1$, which is also the reason that we call $GSAGen_s$ the strict version.

Algorithm 1. The strict version of Group Shilling Attack Generation Algorithm

```
1: procedure GSAGen_s(A_i = {a_1, ··· , a_{n_i}})
2:     b_1 ← a_1, B_i ← {b_1};
3:     for u ← 2 : n do
4:         for j ← 1 : m do
5:             if ∃b_v ∈ B_i, a_uj ≠ 0 && b_vj ≠ 0 then
6:                 if ∀b_t ∈ B_i, b_tj = 0 then
7:                     b_uj ← 2b̄_{v,κ} − b_vj, κ = a_u ∩ b_v;
8:                 end if
9:                 if ∃b_t ∈ B_i, b_tj ≠ 0 then
10:                    Goto line 3;
11:                end if
12:            else if ∀b_v ∈ B_i, a_uj ≠ 0 && b_vj = 0 then
13:                b_uj ← a_uj;
14:            end if
15:        end for
16:        B_i ← B_i ∪ {b_u};
17:    end for
18:    return B_i;
19: end procedure
```

3.2 $GSAGen_l$: A Loose Version

Although the shilling attack group generated by $GSAGen_s$ bids fair to escape the detection of the existing algorithms, the group size may be limited due to the rigorous condition for the shilling group. Meanwhile, $PCC \leq 0$ indicates the dissimilarity of two attackers. Therefore, to find a loose condition of the group shilling attack is a natural idea, and thus we have the following definition:

Definition 2 (The Loose Condition of Group Shilling Attack). *Let* $\mathbf{B}_i = \{b_1, \cdots, b_{n_i}\}$ *be a group of shilling attackers. We say* $\mathbf{B}_i$ *satisfies the loose condition iff* $\forall u \neq v, \kappa = b_u \cap b_v, \kappa = \emptyset$ ***or*** $\kappa \neq \emptyset, \exists j \in \kappa, \forall t \neq u \neq v, b_{tj} = 0$, ***and*** $\forall u \neq v, PCC_{uv} \leq 0$.

In Definition 2, we allow the intersection of any pair of attackers to be empty, however, if the intersection is not empty, there exists at least one item that is rated only by these two attackers. Furthermore, we can guarantee the PCC between any pair of attackers be smaller or equal to 0.

Algorithm 2. The loose version of Group Shilling Attack Generation Algorithm

```
1: procedure GSAGen_l(A_i = {a_1, ··· , a_{n_i}})
2:     b_1 ← a_1, B_i ← {b_1};
3:     for u ← 2 : n do
4:         bool flag ← TRUE;
5:         for t ← 1 : |B_i| do
6:             let κ = a_u ∩ b_t;
7:             if ∀j ∈ κ, |B_i[j]| > 1 then
8:                 flag ← FALSE;
9:                 break;
10:            else
11:                ∀j ∈ κ, |B_i[j]| = 1, b_uj ← 2·b̄_t − b_tj;
12:            end if
13:        end for
14:        if flag =TRUE then
15:            ∀j ∈ [1, m], a_uj ≠ 0 && |B_i[j]| = 0, b_uj = a_uj;
16:            ∀j ∈ [1, m], a_uj ≠ 0 && |B_i[j]| > 1, b_uj = b̄_u;
17:            B_i ← B_i ∪ {b_u};
18:        end if
19:    end for
20:    return B_i;
21: end procedure
```

The pseudocode of $GSAGen_l$ is shown in Algorithm 2. The input of $GSAGen_l$ is also a set of shilling profiles created by attack models, i.e. random or average. In line 2, $\mathbf{B}_i$ is initialized with a shilling profile a_1. In lines 3-19, every profile $a_u \in \mathbf{A}_i$ is examined, and a boolean variable $flag$ is defined to indicate the current a_u whether can be used to generate the b_u. For any b_t contained in the current $\mathbf{B}_i$, if all items in the intersection between a_u and b_t is rated by more than one user in

$\mathbf{B}_i$, $flag$ is set to FALSE and thus the current a_u is skipped(see lines 7-9). Note that $|\mathbf{B}_i[j]|$ is the number of ratings to the j-th item by all profiles $\{b_1, \cdots, b_{u-1}\}$ in $\mathbf{B}_i$. There are three different cases for the assignments to the nonzero items in a_u: (1) if none attacker in $\mathbf{B}_i$ has rated the item, just keep the rating value of a_u (see line 15); (2) if only one attacker in $\mathbf{B}_i$ has rated the item, select this profile $b_t \in \mathbf{B}_i$ and update the rating b_{uj} (see line 11); (3) if more than one attackers in $\mathbf{B}_i$ have rated the item, fill the ratings of these items with the average value of b_u (see line 16). In lines 15-17, the boolean variable $flag$ =TRUE indicates the cases "$\kappa \neq \emptyset, \exists j \in \kappa, \forall t \neq u \neq v, b_{tj} = 0$" or "$a_u$ has no intersection with the current $\mathbf{B}_i$" happened. However, Algorithm 2 does not tell us $\forall u \neq v, PCC_{uv} \leq 0$ directly, which will be shown in the following theorem.

Theorem 2. $\forall b_u, b_v \in \mathbf{B}_i$, $PCC_{uv} = \begin{cases} 0 & if \quad b_u \cap b_v = \emptyset \\ -1 & otherwise \end{cases}$

PROOF: Let $\kappa = b_u \cap b_v$ and $PCC_{uv} = \frac{\sum_{j \in \kappa}(b_{uj} - \overline{b_u})(b_{vj} - \overline{b_v})}{\sqrt{\sum_{j \in \kappa}(b_{uj} - \overline{b_u})^2}\sqrt{\sum_{j \in \kappa}(b_{vj} - \overline{b_v})^2}}$. Obviously, $PCC_{uv} = 0$ when $\kappa = \emptyset$. When $\kappa \neq \emptyset$, $\forall j \in \kappa, b_{uj} = \overline{b_u}$, the j-th item does not contribute to numerator and denominator of PCC_{uv}, since $b_{uj} - \overline{b_u} = 0$. However, $flag$ =TRUE in line 15 of Algorithm 2 guarantee that there is at least one item satisfying $a_{uj} \neq 0$ and $|\mathbf{B}_i[j]| = 1$.
$\because \exists j, b_{uj} = 2\overline{b_v} - b_{vj}$, and $\overline{b_u} = \overline{b_v}$.
$\therefore \exists j, b_{uj} - \overline{b_u} = -(b_{vj} - \overline{b_v})$, and thus $PCC_{uv} = -1$,which completes the proof. □

It is interesting to note that the computation of PCC in Theorem 2 utilizes the average on the whole rating, while the average on the intersection set κ is used in Theorem 1. In the literature, these two kinds of PCC computation methods are often exchanged equivalently. For instance, the average on the whole rating is adopted in [9, 15], while the average on the intersection set is employed in [14].

Discussion: Indeed, $GSAGen_s$ is similar to the method proposed in [14] for generating *high diversity* single attack profiles. In this paper, we borrow ideas from [14] to design the strict version of generative model for shilling group. However, the size of the output profiles of $GSAGen_s$ is rather scarce due to the strict condition for the generated attackers, which will be shown in Section 4.2. The new $GSAGen_l$ extends the generative condition in order to create more profiles in a shilling group that can maintain the big negative effect on recommender systems and have the good anti-detection performance.

4 Experimental Study

In this section, we conduct sets of experiments on `MovieLens`[1] data set to illustrate the effectiveness of the proposed group shilling attack model. And the experimental results show: (1) $GSAGen_l$ is more suitable to generate the shilling

[1] `http://movielens.umn.edu/`

group including more attack profiles than $GSAGen_s$. (2) The attackers in the shilling group works together to exert big negative effect to recommender systems. (3) The group shilling attacks generated by both $GSAGen_s$ and $GSAGen_l$ can avoid the detection by the existing methods.

4.1 Experiment Setup

Data Sets. `MovieLens` data set is published by GroupLens and consists of 100,000 ratings on 1682 movies by 943 users. All ratings are integer values ranged from 1 to 5 where 1 indicates disliked and 5 indicates most liked. `MovieLens` data set is widely used in the realm of shilling attack detection [1, 3, 4, 7, 8, 10, 11].

Vulnerability Measures. For the purpose of evaluating the vulnerability of the recommender system against shilling attacks, two kinds of measures are employed, i.e., Average Prediction Shift ($\overline{\Delta}$) for rating prediction and Hit Ratio (HR) for ranking prediction.

$$\overline{\Delta} = \sum_{(u,i)\in N} \frac{|p'_{ui} - p_{ui}|}{|N|}. \tag{1}$$

where p'_{ui} represents the prediction rating after the attack and p_{ui} before, and N is the set of missing ratings of normal users.

$$HR = \frac{\#hits}{K \cdot T_i}. \tag{2}$$

where $\#hits$ indicates the number of target items in the top K recommendation list, and T_i is the number of target items. Since whether target items enter the recommendation list is the key issue that attackers considered, HR can better reflect the power of shilling attacks.

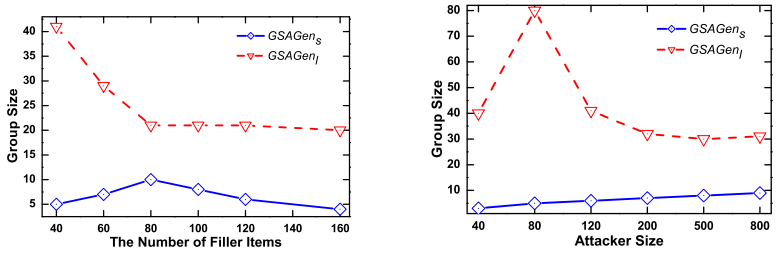
Detection Measures. The widely-used recall(R), precision(P), and F-measure(F) are adopted for the detection performance evaluation.

$$P = \frac{TP}{TP + FP},\ R = \frac{TP}{TP + FN},\ F = \frac{2PR}{P + R}. \tag{3}$$

with TP being the number of truly identified attackers, TN the number of truly identified normal users, FP the number of wrongly identified attackers, and FN the number of missed attackers. In general, R and P highlight the completeness and accuracy of a detector, respectively, and F provides a global view.

Generators and Detectors. We have implemented $GSAGen_s$ and $GSAGen_l$ as two generators in C++. Meanwhile, two traditional shilling detectors, i.e. C4.5 [3] and PCA (PCASelectUsers) [9] are employed for anti-detection performance evaluation. PCA was coded in MATLAB to facilitate the principal-component computation. C4.5 was the J48 version provided by WEKA[2] with the default settings.

[2] `http://www.cs.waikato.ac.nz/ml/weka/`



(a) Filler size (b) Input attack size

Fig. 2. A comparison on the generated shilling group size

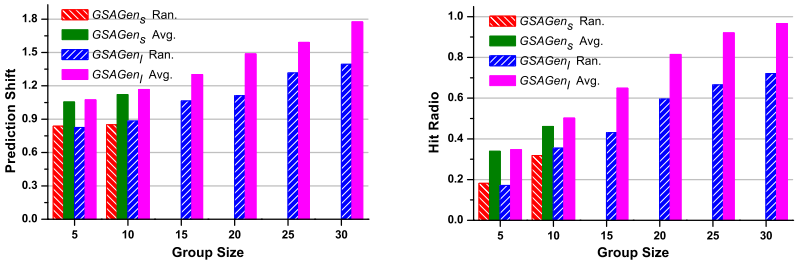
Procedure. We assumed the users in the `MovieLens` data set were normal. Then, we generated attacker profiles according to the attack models mentioned in Table 1, and utilized two generators to produce group shilling attacker profiles which were injected into `MovieLens` data set. Finally, various detectors were invoked to classify the normal users and shilling attackers.

4.2 $GSAGen_s$ vs. $GSAGen_l$ on Group Size

In Section 3.2, we have mentioned that the group size generated by $GSAGen_s$ is much smaller than $GSAGen_l$. Here, we demonstrate it through experiments. Two factors, i.e. the filler size b_j and the input attack size $|\mathbf{A}_i|$, affect the size of output shilling group. Obviously, the smaller filler size b_j, the larger group size, and the larger attack size $|\mathbf{A}_i|$, the larger group size. Fig. 2(a) shows the effect of filler size on the size of shilling group under $|\mathbf{A}_i| = 100$, and Fig. 2(b) depicts the effect of input attack size under $b_j = 40$. Note that every movie is selected as the filler item with equal probability. As can be seen from Fig. 2, $GSAGen_l$ can indeed generate more profiles in the shilling group than $GSAGen_s$. Specially, when b_j and $|\mathbf{A}_i|$ are small, $GSAGen_l$ can transform all profiles in $\mathbf{A}_i$ to high diversity profiles, since the overlapping between any pair of profiles are small (see two points $|\mathbf{A}_i| = 40$ and 80 in Fig. 2(b)). However, $GSAGen_s$ still generates less than 10 profiles in the shilling group.

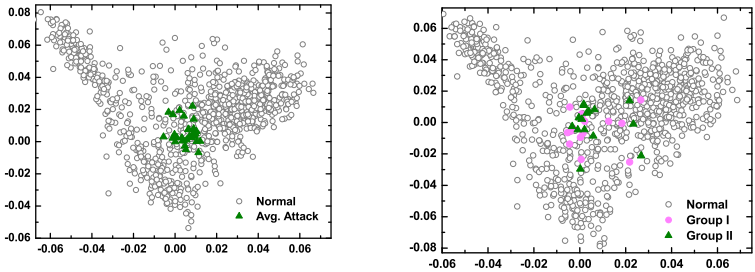
4.3 Vulnerability Analysis

Here, we investigate the vulnerability of the recommender system against the group shilling attack. The User-based Collaborative Filtering (UCF) method [16] is employed as the kernel algorithm of the recommender system. We set the number of nearest neighbors $k = 20$ and the length of recommendation list $K = 10$. A shilling group with 3 target items is inserted into user-item database. We utilize both $GSAGen_s$ and $GSAGen_l$ based on random and average attack model to generate 4 kinds shilling groups, respectively.



(a) Prediction shift (b) Hit Ratio

Fig. 3. Effect of shilling group to UCF based Recommender Systems



(a) Common Average Attacks (b) Group Shilling Attacks

Fig. 4. Dispersion shapes of common average attacks and group shilling attacks

As can be seen from Fig. 3, the shilling group generated by $GSAGen_l$ exerts bigger negative effect on the recommender system than $GSAGen_s$, and the shilling group created based on average attack model affects more than the random attack model. $GSAGen_s$ often generated less than 10 profiles in the shilling group, which led to the missing bars for the cases of "$GSAGen_s$ Ran." and "$GSAGen_s$ Avg." in Fig. 3. When we take a deep look at the hit ratio shown in Fig. 3(b), as the increase of the group size, the hit ratio becomes fairly high. For instance, the hit ratio a shilling group containing 20 attackers is larger than 0.8, which indicates the target items have entered the recommendation list of over 80% normal users!

4.4 The Anti-detection Performance

In this subsection, we ascertain the anti-detection performance of group shilling attack. Two kinds of detectors are chosen as the representatives of the existing methods, i.e., C4.5 for supervised learning based method and PCA for

unsupervised learning based method. Since C4.5 is a feature-based detector, we use a feature selection algorithm *MC-Relief* proposed in our previous work [11] to select effective features on `MovieLens` data set. Finally, we obtain 5 features among 10 features for C4.5, i.e., *Entroy*, *LengthVar*, *MeanVar*, *RDMA* and *DegSim*. PCA is a rating-based detector. So, we can run PCA on rating records directly. In addition, we assume that PCA know the number of group size $|\mathbf{B}_i|$, and return the top-$|\mathbf{B}_i|$ users as attackers. The number of filler items is set to 40 and the input attack size is 80.

Table 2. The anti-detection performance comparison in terms of $FMeasure$

Gen. Method	Attack	Detector	Group Size					
			5	**10**	**15**	**20**	**25**	**30**
None	*Ran.*	C4.5	0.774	0.882	0.912	0.0.936	0.953	0.979
		PCA	1	1	1	1	1	1
	Avg.	C4.5	0.756	0.877	0.903	0.93	0.944	0.953
		PCA	0.8	0.9	0.933	0.95	1	1
$GSAGen_s$	*Ran.*	C4.5	0.375	0.346	–	–	–	–
		PCA	0.6	0.7	–	–	–	–
	Avg.	C4.5	0.23	0.232	–	–	–	–
		PCA	0.6	0.7	–	–	–	–
$GSAGen_l$	*Ran.*	C4.5	0.368	0.324	0.308	0.291	0.286	0.283
		PCA	0.4	0.6	0.6	0.65	0.56	0.567
	Avg.	C4.5	0.277	0.268	0.262	0.259	0.255	0.252
		PCA	0.4	0.6	0.533	0.5	0.52	0.533

Table 2 shows the results in terms of $FMeasure$. As can be seen, for common average and random attackers, PCA performs nearly perfect and C4.5 also performs fairly good. However, when the average and random attacks are transformed to group shilling attacks, the detection performance of two detectors decreases remarkably. Although PCA can recognize about half attackers, PCA is assumed to know the number of injected attackers. Therefore, the performance of PCA will be degraded in practise, which in turn, justify that the shilling groups generated by the proposed methods have good anti-detection performance.

Furthermore, we take a deeper look at the detection performance of PCA according to the dispersion shapes of genuine users and several shilling groups. Fig. 4 shows the dispersion shapes of common average attacks and group shilling attacks. As can be seen, the common average attackers tend to be gathered at the center of normal users. After we apply $GSAGen_l$ on these common average attackers to generate two shilling groups, the attackers in a group become scattered. However,attackers in different shilling groups may have overlapping, since different shilling groups often lack for premeditation. Fig. 4 implies that

the proposed $GSAGen_l$ can successfully make the high similar attackers be more diverse.

5 Related Work

The pioneer work on shilling attack detection can be traced to [1, 2]. Since then, several generative models of shilling attacks as shown in Table 1 have been proposed [3, 4]. From machine learning perspective, three kinds of shilling attack detection methods were presented in the literature, i.e., supervised learning based method [3, 7, 8, 17], unsupervised learning based method [9, 10, 18], and semi-supervised learning based method [19, 12, 11]. Different from the work on defending the shilling attack, Cheng and Hurley [14] studied the problem from the attackers' perspective, and presented an effective diverse and obfuscated attacks which has inspired our work on the strict version of group shilling attack generation algorithm.

To date, few of research has been carried out on the modeling and detecting of group shilling attacks. Only preliminary discussion on group shilling was presented in [13]. However, many research has been devoted to *group review spam* detection [20, 21, 22]. Our paper indeed fills this crucial void by proposing the $GSAGen_s$ and $GSAGen_l$ to model the group shilling attacks.

6 Conclusion and Future Work

In this paper, we proposed a group shilling attack generation approach under the observation that the attackers with low similarity are difficult to be discovered by the traditional detectors. Specifically, a strict version of generative method named $GSAGen_s$ was presented to create shilling profiles with highest diversity, i.e. $PCC = -1$. However, $GSAGen_s$ suffered from the small output group size. Therefore, we extended the strict version to obtain the loose version $GSAGen_l$ which guaranteed the output shilling profiles have high diversity, i.e. $PCC \leq 0$. Experimental results on MovieLens data set demonstrate that the shilling groups generated by the proposed model have great negative influence on recommender systems and can effectively avoid the detection by the traditional methods. Future study will include additional examinations on more tricksy shilling groups as well as their obfuscated techniques. The design of a new detector for group shilling attacks combing the HySAD detector [11] toward developing a complete shilling detectors is planned for future direction.

Acknowledgments. This research is supported by National Natural Science Foundation of China under Grants No.61103229, 71072172 and 41201486, Industry Projects in the Jiangsu S&T Pillar Program under Grants No. BE2011198, Jiangsu Provicial Colleges and Universities Outstanding S&T Innovation Team Fund under Grants No. 2011013, Key Project of Natural Science Research in Jiangsu Provincial Colleges and Universities under Grants No. 12KJA520001,

National Key Technologies R&D sub Program in 12th five-year-plan under Grants No. SQ2011GX07E03990,the Natural Science Foundation of Jiangsu Province of China under Grant BK2012863, and International S&T Cooperation Program of China under Grants No. 2011DFA12910.

References

[1] Lam, S., Riedl, J.: Shilling recommender systems for fun and profit. In: Proceedings of the 13th International Conference on World Wide Web (WWW 2004), pp. 393–402 (2004)
[2] O'Mahony, M., Hurley, N., Kushmerick, N., Silvestre, G.: Collaborative recommendation: A robustness analysis. Transactions on Internet Technology (TOIT) 4(4), 344–377 (2004)
[3] Williams, C.: Profile injection attack detection for securing collaborative recommender systems. Technical report, DePaul University (2006)
[4] Mobasher, B., Burke, R., Bhaumik, R., Williams, C.: Toward trustworthy recommender systems: An analysis of attack models and algorithm robustness. Transactions on Internet Technology (TOIT) 7(4), 1–40 (2007)
[5] Chiang, M., Peng, W., Yu, P.: Exploring latent browsing graph for question answering recommendation. World Wide Web: Internet and Web Information Systems, WWWJ (2011)
[6] Zhang, S., Chakrabarti, A., Ford, J., Makedon, F.: Attack detection in time series for recommender systems. In: Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD 2006 (2006)
[7] Chirita, P., Nejdl, W., Zamfir, C.: Preventing shilling attacks in online recommender systems. In: Proceedings of the 7th Annual ACM International Workshop on Web Information and Data Management (WIDM 2005), pp. 67–74 (2005)
[8] Burke, R., Mobasher, B., et al.: Classification features for attack detection in collaborative recommendation systems. In: Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD 2006 (2006)
[9] Mehta, B., Nejdl, W.: Unsupervised strategies for shilling detection and robust collaborative filtering. User Modeling and User-Adapted Interaction 19(1-2), 65–97 (2009)
[10] Lee, J., Zhu, D.: Shilling attack detection—a new approach for a trustworthy recommender system. INFORMS Journal on Computing 24(1) (2011)
[11] Wu, Z., Wu, J., Cao, J., Tao, D.: HySAD: A semi-supervised hybrid shilling attack detector for trustworthy product recommendation. In: Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data mining (KDD 2012), pp. 985–993 (2012)
[12] Cao, J., Wu, Z., Mao, B., Zhang, Y.: Shilling attack detection utilizing semi-supervised learning method for collaborative recommender system. World Wide Web: Internet and Web Information Systems (WWWJ) (2012)
[13] Su, X., Zeng, H.J., Chen, Z.: Finding group shilling in recommendation system. In: Proceedings of the 14th International Conference on World Wide Web, WWW 2005 (2005)
[14] Cheng, Z., Hurley, N.: Effective diverse and obfuscated attacks on model-based recommender systems. In: Proceedings of ACM Conference on Recommender Systems (RecSys 2009), pp. 141–148 (2009)

[15] Rashid, A.M., Karypis, G., Riedl, J.: Influence in ratings-based recommender systems: An algorithm-independent approach. In: Proceedings of SIAM International Conference on Data Mining, SIAM 2005 (2005)
[16] Herlocker, J., Konstan, J., Terveen, L., Riedl, J.: Evaluating collaborative filtering recommender systems. ACM Transactions on Information Systems (TOIS) 22(1), 5–53 (2004)
[17] Mobasher, B., Burke, R., Williams, C., Bhaumik, R.: Analysis and detection of segment-focused attacks against collaborative recommendation. In: WebKDD Workshop (2006)
[18] Hurley, N., Cheng, Z., Zhang, M.: Statistical attack detection. In: Proceedings of ACM Conference on Recommender Systems, RecSys 2009 (2009)
[19] Wu, Z., Cao, J., Mao, B., Wang, Y.: SemiSAD: Applying semi-supervised learning to shilling attack detection. In: Proceedings of ACM Conference on Recommender Systems (RecSys 2011), Chicago, IL, USA, pp. 289–292 (2011)
[20] Mukherjee, A., Liu, B., Glance, N.S.: Spotting fake reviewer groups in consumer reviews. In: Proceedings of the 21th International Conference on World Wide Web (WWW 2012), pp. 191–200 (2012)
[21] Mukherjee, A., Liu, B., Wang, J., Glance, N.S., Jindal, N.: Detecting group review spam. In: Proceedings of the 20th International Conference on World Wide Web WWW 2011 (Companion Volume), pp. 93–94 (2011)
[22] Leung, C., Chan, S., Chung, F., Ngai, G.: A probabilistic rating inference framework for mining user preferences from reviews. World Wide Web: Internet and Web Information Systems (WWWJ) 14(2), 187–215 (2011)