A Comparative Study of Shilling Attack Detectors for Recommender Systems

Youquan Wang[1,2], Lu Zhang[1*], Haicheng Tao[2], Zhiang Wu[1], Jie Cao[1,2]
[1] *Jiangsu Provincial Key Laboratory of E-Business, Nanjing University of Finance and Economics, Nanjing, China*
[2] *College of Computer Science and Engineering, Nanjing University of Science and Technology, Nanjing, China*
∗*Corresponding author: luzhang@njue.edu.cn*

***Abstract*—Uncovering shilling attackers hidden in recommender systems is very crucial to enhance the robustness and trustworthiness of product recommendation. Many shilling attack detection algorithms have been proposed so far, and they exhibit complementary advantage and disadvantage towards various types of attacks. In this paper, we provide a thorough experimental comparison of several well-known detectors, including supervised and unsupervised methods. MovieLens 100K is the most widely-used dataset in the realm of shilling attack detection, and thus it is selected as the benchmark dataset. Meanwhile, seven types of shilling attacks generated by average-filling and random-filling model are tested in our experiments. As a result of our analysis, we show clearly causes and essential characteristics insider attackers that might determine the success or failure of different kinds of detectors.**

Keywords-Recommender System; Shilling Attack Detection; Supervised Learning; Unsupervised Learning; MovieLens

I. INTRODUCTION

Shilling attacks, a.k.a. profile injection attacks, attempt to influence the system's behavior by injecting a set of fake profiles into the database of normal user profiles [1,2]. Even the most robust of the recommender systems studied have not been unaffected by shilling attacks, and no collaborative system could be [3]. In recent years, extensive studies have been proposed for detecting and reducing the effects of shilling attacks [1-22]. These studies mainly focus on the three subareas: the shilling attack generative models [2,3], the shilling attack detection features [3,10,11], and the shilling attack detection algorithms [3-6,10,12-22]. Many kinds of techniques have been applied to the detector design, such as statistical methods [7,8,12], classification models [3,5,6,17], matrix factorization models [7,14,22], etc. Most classification based detectors represent every user profile in a feature space and then use supervised or semi-supervised learning for training the classification model. Statistical and matrix factorization based detectors are usually unsupervised, and they essentially try to compute a rank score of each user for measuring its suspicious degree to be an attacker.

Supervised classification and unsupervised matrix factorization are two mainstream techniques used in shilling attack detectors. They have been adopted as baseline tools as a new detector is developed [13,17]. However, none of work has conduct a complete yet in-depth analysis on which type of shilling attacks these detectors are/are not be qualified, and more importantly why they can/cannot identify a special type of attackers. Furthermore, we would like to summarize the essential characteristics of attackers that might determine the success or failure of different kinds of detectors.

In this paper, we offer a thorough comparative study on four well-known detectors, i.e., supervised C4.5 and NB, as well as unsupervised PCA and MDS. In particular, we first provide overall performance comparison of four detectors against seven types of shilling attacks on MovieLens 100K dataset. A number of further experiments are then done to show essential characteristics and differences of various detectors. Our experimental results validate give several interesting observations and the corresponding deeper analysis clearly show the causes to these observations. We believe that this work will play an important role in promoting the understanding on mechanisms inside various detectors and inspiring the design of novel detectors.

II. SHILLING PROFILE GENERATION

The profile of a shilling attacker, i.e., shilling profile, is in essence a rating record on various items, and it is usually composed of ratings on three-typed items: target items, filler items, and non-voted items, as shown in Fig. 1. Rating target items is determined by the attack intent, i.e., rating the highest score in a push attack or the lowest score in a nuke attack. Filler items can make a shilling profile look normal, yet exert profound impacts on other users. In certain attack types, such as bandwagon, a subset of filler items, a.k.a. selected items, may be pre-selected for a precise impact [1,3].

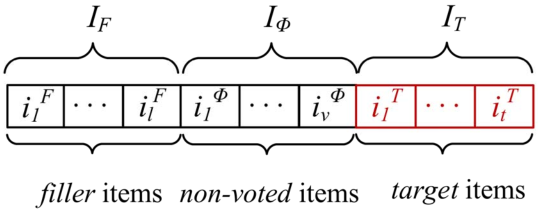


Figure 1. Illustration of a shilling profile.

The shilling attacks can be categorized in several types according to the methods for selecting filler items and assigning ratings to them. There are three ways to select filler items: random selection, selection over popular items, and the combination of random and popular. The random-filler model (RFM) and average-filler model (AFM) represent two different ways to assign ratings to items in I_F. Therefore, the shilling attack models are finally summarized as six types, as shown in

Table 1. The random attack and average attack are basic attack types introduced in [2] and further generalized in [3]. Choosing filler items with equal probability from the top *x*% of the most popular items improves the random and average attack into more effective versions, i.e., Random-over-Popular (RoP) and Average-over-Popular (AoP) [12]. The hybrid selection strategy chooses a set of popular items, gives them the maximum allowed ratings, and then randomly selects filler items to which either random ratings or average ratings are assigned. This gives birth to the bandwagon attack including random version and average version [7]. Note that RoP attack has not been referred in any literature, yet it is an intuitive extension of AoP.

TABLE 1. SHILLING ATTACK TYPES.

Rating Assignment **Item Selection**	**RFM**	**AFM**
Random	Random Attack	Average Attack
Popular	RoP	AoP
Combination	Bandwagon (random)	Bandwagon (average)

III. SUPERVISED SHILLING DETECTION ALGORITHMS

The most common way models shilling attack detection as a classification problem, and utilizes supervised or semi-supervised learning technique for training models [3,5]. This type of detectors is feature-based, that is, it first defines a set of metrics and thus transforms each profile (i.e., rating record) into a vector in the feature space. Various classification models, such as C4.5, SVM, Naïve Bayesian, etc, can then be used. In this section, we introduce feature selection and thus supervised shilling detectors in a wrapped view.

A. Feature Selection

In the literature [3,5,10,15], a number of features are defined for distinguishing shilling attackers against normal users. However, some of them are not disjointed. Altogether 10 features are collected from the literature, including Entropy [15], DegSim [10], LengthVar [5], RDMA [10], WDMA [5], WDA [5], MeanVar [5], FMTD [5], GFMV [3], and TMF [5]. Due to the limited space, we omit the definitions of these features here.

In fact, among 10 candidate features, some of them are not disjointed. For example, WDMA and WDA [5] are the same as RDMA [10] except on the normalization method. Meanwhile, a feature is usually designed purposefully for one attack model. For instance, FMTD is designed for bandwagon attacks and MeanVar is intended for average attacks. Therefore, it is necessary to adopt the feature selection to screen several discriminative features.

Feature selection is also a supervised learning process, which is not in conflict with the supervised detectors. That is, based on labeled instances for training, feature selection tries to find a subset of most discriminative features on the training data. Since the our target is to compare shilling attack detectors rather than feature selection algorithms, we here use a simple heuristic MC-Relief [17] to select the right metrics for the subsequent supervised training.

For clarity, we give a brief introduction to the MC-Relief heuristic. Generally, MC-Relief aims to estimate the weight of features to distinguish the user profiles that are near to each other. Given *m* features ω_l ($1 \le l \le m$) is the weight of a feature. For Estimating ω_l, MC-Relief does a random sampling on training data. Each time, assuming the user *u* is picked out, *u*'s nearest neighbors in *c* different classes are searched, and then the weight of each feature will be updated once according to Eq. (1).

$$\omega_l \leftarrow \omega_l - \frac{|x_{u,l}-x_{u_p,l}|}{S} + \frac{\sum_{k=1}^{c-1}|x_{u,l}-x_{u_n,l}^{(k)}|}{S(c-1)}, \quad (1)$$

where $x_{u,l}$ denotes the *l*-th feature value of *u*, u_p and u_n are the *u*'s nearest neighbor in the same and different classes, and *S* is the sampling size. Intuitively, the larger difference of a feature on two instances in the same class, the less discriminative of this feature, and vice versa for a pair of instances from different classes.

B. Supervised Learning

Given a number of selected features, every user profile in both training and test data is represented as a feature vector. Since the training examples often fall in two classes, i.e., normal and shilling, the shilling attack detection problem is transformed to a binary classification problem. Rather, all kinds of algorithms can then be used for training the classifier. In this paper, we consider two kinds of classification models, including the C4.5 decision tree and Naïve Bayesian (NB). C4.5 was ever used for building the decision tree to identify various types of shilling attackers in [3]. NB was used for shilling attack detection in [6,18] due to its good interpretability. An important assumption within NB is that the continuous values of each feature yield the Gaussian distribution, which lays the foundation of parameter estimation. Since C4.5 and NB model are preliminaries in data mining, we here omit the details on the model training.

IV. USUPERVISED SHILLING DETECTION ALGORITHMS

Generally, unsupervised detectors [7,20-22] run on user-item rating matrix directly, rather than the feature space. In this section, we briefly introduce two well-known unsupervised detectors: PCA-based algorithm [14,21] and MDS-based algorithm [6]. These two detectors are in common with using the matrix factorization technique, but on user-user covariance (i.e., similarity) matrix and dissimilarity matrix respectively.

A. PCA-based Algorithm

The PCA-based algorithm is also known as *PCASelectUsers* [14,21], PCA for short in this paper. It is the first and the most famous unsupervised detector in shilling attack detection. Based on user-item rating matrix, PCA computes the user-user covariance matrix. Mathematically, if let r_u and r_v be rating vectors of user *u* and *v*, the covariance denoted by $Cov_{u,v}$ is

$$Cov_{u,v} = \frac{\sum_i (r_{u,i}-\bar{r}_u)(r_{vi}-\bar{r}_v)}{\sqrt{\sum_i (r_{u,i}-\bar{r}_u)^2}\sqrt{(r_{v,i}-\bar{r}_v)^2}}, \quad (2)$$

where $r_{u,i}$ is the rating given to item i by user u, and $\bar{r}_u$ denotes the average rating of user u. Then, the Eigen value decomposition is executed on the covariance matrix, and g components with the largest Eigen values are extracted as principal components (PCs). Commonly, g is set to 3. As a result, every user profile is projected to g PCs and the sum of squares on g PCs is adopted as the ranking score. Finally, top-r users with the smallest ranking scores are returned as shilling attackers.

B. *MDS-based Algorithm*

The MDS-based algorithm, MDS for short, is a two-phase detector. Phase 1 targets at extracting a subset of effective users, of which the idea is very similar to PCA. That is, MDS computes user-user dissimilarity matrix where each element is

$$d_{u,v} = 1 - Cov_{u,v}. \tag{3}$$

By minimizing a loss function called "Stress" [23], the $n \times n$ dissimilarity matrix is reduced to an $n \times g$ configuration matrix, where g is also set to 3. Thus, each user profile is also projected to a g-dimensional vector $\tilde{x}_u$ $(1 \leq u \leq n)$. The ranking score is defined as the sum of Euclidean distances between a user and other users on the g-dimensional space. Formally,

$$y_u = \sum_{v=1}^{n} \tilde{d}_{u,v}, \text{ where } \tilde{d}_{u,v} = \sqrt{\sum_{l=1}^{g} \left(\tilde{x}_{u,l} - \tilde{x}_{v,l}\right)^2}. \tag{4}$$

where $\tilde{x}_{u,l}$ and $\tilde{x}_{v,l}$ are the lth-dimension value of vector $\tilde{x}_u$ and $\tilde{x}_v$. Given y_u $(1 \leq u \leq n)$, sort every user in y's ascending order, and return top users with its y greater than a threshold λ. According to [7], $\lambda = y_{min} + (y_{max} - y_{min})/3$.

Phase 2 invokes K-means to divide the selected users into K clusters. Note that the clustering is directly executed on original user-item matrix and the 1-Spearman rank similarity is employed as the distance function. Among K clusters, MDS utilizes a feature called GRDMA, an extension on RDMA, to select a cluster with the maximal GRDMA value as the shilling group. That is, all users in this group are returned as shilling attackers. $GRDMA_K$ is defined as:

$$GRDMA_K = max_j \left\{ \left|\bar{r}_{K,j} - \bar{r}_j\right| * \left(\frac{|r_{K,j}|}{|r_j|}\right)^2 \right\}, j = 1, \cdots, |r_K|, \tag{5}$$

where $\bar{r}_{K,j}$ is the average rating value for item j given by users belonging to the cluster K, $\bar{r}_j$ is the global average rating value of item j, $|r_{K,j}|$ is the number of ratings on item j given by users belonging to the user cluster K, $|r_j|$ is the number of ratings on item j given by all users in the system, and $|r_K|$ is the number of rating items rated by users within the cluster K.

V. Experimental Results and Analysis

In this section, we present the overall performance comparison of both supervised and unsupervised detectors, and remark causes for the success or failure of these detectors.

C. *Experimental Setup*

Dataset. We use the *MovieLens* 100K dataset (ML-100K for short) that is the most widely-used data in the realm of shilling attack detection [1-3,5-22]. This dataset consists of 100,000 ratings on 1682 movies by 943 users. In practice, we use *u2.base*, a random split of the full dataset, as the test dataset, which is the same as the experimental settings in the literature [9,17,22].

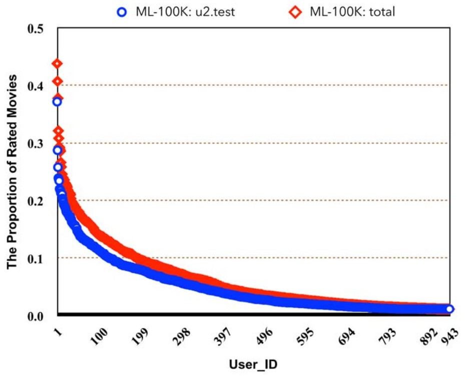


Figure 2. The distribution of length of user profiles.

Attack Injection. We inject 94 attacker profiles into *u2.base* that contains 943 users, i.e., the attacker size is about 10%. The premise is that the user profiles in *MovieLens* dataset are normal, and injected profiles are shilling attackers. Fig. 2 plots the proportion of users' rated movies, i.e., the length of user profiles, in *u2.base* and the original ML-100K dataset. Obviously, the distribution is long-tailed. Accordingly, we set three values for the filler size, i.e., FS = 5%, 10% and 30% respectively, to simulate short, medium and long user rating profiles. Average and random attack profiles under three cases of FS are injected, beside which the AoP attack with FS = 10% and x = 20% is also generated. So, altogether seven types of shilling attacks are used in our experiments.

Detection Algorithms. We use four detection algorithms for comparison and analysis, i.e., NB, C4.5, PCA, and MDS. NB and C4.5 are supervised detectors that are the NaiveBayes and J48 version provided by WEKA with the default settings. Since supervised detectors are feature-based, we invoke MC-Relief to select 5 effective features out of 10 features. Two unsupervised detectors, i.e., PCA and MDS are coded in MATLAB by ourselves, because MATLAB can provide more convenient implementations for the principal-component and matrix-decomposition computation.

Measures. The widely-used recall (R), precision (P), and F-measure (F) are adopted for performance evaluation. Three measures are defined on the class of shilling attackers, i.e., taking attackers as positive instances.

$$P = \frac{\#TP}{\#TP + \#FP}, \quad R = \frac{\#TP}{\#TP + \#FN}, \quad F = \frac{2PR}{P+R}, \tag{6}$$

where $\#TP$ and $\#FP$ denote the number of truly and wrongly identified attackers respectively, and $\#FN$ is the number of missed attackers.

B. *Overall Performance Comparison*

Table 2 shows comparative results of four detectors against five types of attacks. For supervised detectors, we conduct 10-fold cross validation and report the average values of every measure. To set the parameter *r* of PCA, we assume the number of attackers is known (i.e., *r* = 94), though materially impossible. This assumption leads to *P* = *R* = *F* in the results of PCA. For MDS with a tunable parameter *K*, we vary *K* from 5 to 15 with an interval 1, within which the best results are recorded.

TABLE 2. THE PERFORMANCE OF ALL DETECTORS.

Detector	Measure	*FS* = 5%		*FS* = 10%		*FS* = 30%		*AoP*
		Ran	*Avg*	*Ran*	*Avg*	*Ran*	*Avg*	
NB	*P*	0.972	1	0.965	1	0.992	1	0.582
	R	0.989	0.979	0.989	0.904	1	1	0.745
	F	0.981	0.989	0.977	0.945	0.996	1	0.649
C4.5	*P*	0.879	1	0.949	0.939	0.939	0.944	0.877
	R	0.969	0.904	0.926	0.872	1	1	0.830
	F	0.916	0.945	0.937	0.902	0.967	0.971	0.845
PCA	*P*	0.989	0.968	0.979	0.957	0.979	0.904	0
	R	0.989	0.968	0.979	0.957	0.979	0.904	0
	F	0.989	0.968	0.979	0.957	0.979	0.904	0
MDS	*P*	0	0.930	0.130	0.913	0.863	0.969	0.391
	R	0	0.703	0.006	1	0.468	1	0.287
	F	0	0.800	0.032	0.954	0.607	0.984	0.331

Several initial observations can be made from Table 2. First, due to the priori knowledge, supervised learning the average attack is easier to be detected than the random attack, while to detect AoP Attack is most difficult task. Second, none of detectors can effectively identify every type of shilling attacks. Therefore, in what follows, we try to clarify two important problems based on the deep analysis for various detectors. That is, why each detector can or cannot identify a special type of attackers, and what is the essential characteristic inside an attacker that might determine the success or failure of every detector.

C. Analysis on Supervised Detectors

The supervised detectors select 5 features out of 10 features by using MC-Relief, and assume each feature yields the Gaussian distribution. Table 3 lists the five selected features for every type of shilling attackers. Although the overall performance of supervised detectors is satisfactory, we remark that the supervised detectors behave unstably, because they heavily depend on the training data. To illustrate this, we depict the *F* values of every fold during 10-fold cross validation as boxplots in Fig. 3.

TABLE 3. TOP-5 FEATURES GENERATED BY MC-RELIEF.

	FS = 5%		*FS* = 10%		*FS* = 30%		*AoP*
	Ran	*Avg*	*Ran*	*Avg*	*Ran*	*Avg*	
No. 1	WDA	TMF	WDA	WDA	WDA	WDA	TMF
No. 2	TMF	WDA	TMF	TMF	DegSim	LengthVar	WDA
No. 3	RDMA	MeanVar	DegSim	WDMA	LengthVar	DegSim	WDMA
No. 4	DegSim	LengthVar	RDMA	RDMA	RDMA	TMF	RDMA
No. 5	LengthVar	RDMA	WDMA	DegSim	WDMA	RDMA	DegSim

From Fig. 3, the performance of supervised detectors fluctuates considerably, especially that of C4.5. NB is more stable than C4.5, of which the reason is that NB is affected by the joint probability of all selected features, yet C4.5 is commonly to use few features for constructing a decision tree. We take an outlier point in the case of random attack with *FS* = 10% for an example, as circled in Fig. 3. In this case, the feature *DegSim* plays a decisive role. Fig. 4(a) shows bell-shaped curves of *DegSim* on training and test data. As can be seen, *DegSim* is indeed very discriminative, that is, two classes both in training and test data are clearly separated. As a result, C4.5 only use one feature *DegSim* to construct the decision tree, as shown in Fig. 4(b). However, since the scale of *DegSim* varies on training and test data, the "shilling" class of test data has more overlaps with the "normal" class of training data, rather than the "shilling" class of the training data. This leads to the increase of the *FN*.

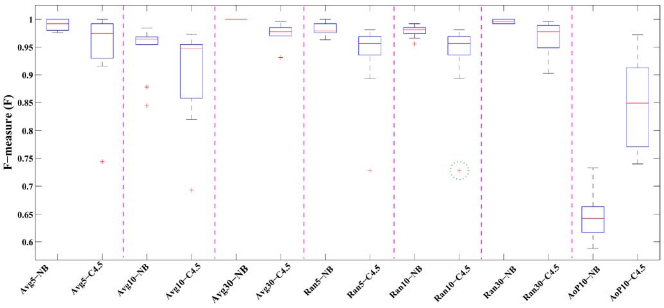
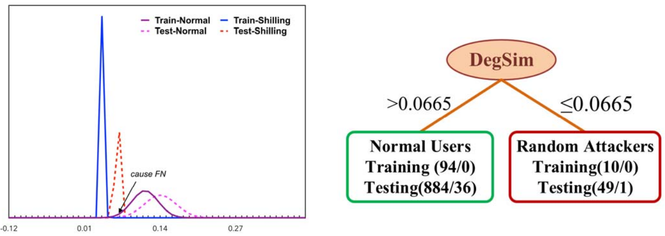


Figure 3. The stability of supervised detectors.



(a) Distribution of *DegSim*

(b) C4.5 decision tree

Figure 4. Analysis for the valley point in random attack with *FS* = 10%.

Remark: With effective features, the supervised detector can deal with various types of shilling attacks. If the selected features are identically distributed on training and test data, the performance of supervised detector will be superior. On the contrary, the performance tends to be degraded, which leads to the instability of supervised detectors. Furthermore, a large number of labeled instances for training might be hard to be obtained, which limits the practical applications of the supervised methods.

D. Analysis on Unsupervised Detectors

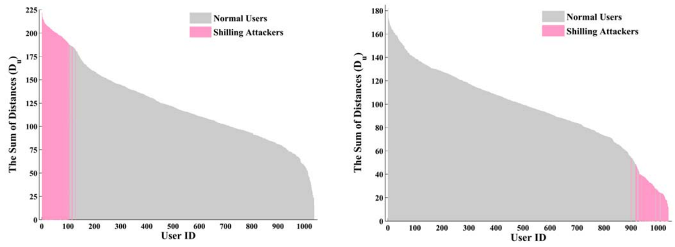
1) PCA-based Detector

The PCA-based detector works on user-item rating matrix directly, rather than the feature space. It takes the covariance between users' rating profiles as interrelated variables, and utilizes PCA to obtain *g* principal variables. The ranking score is computed as the sum of squares on *g* components to which each user is projected. Rather, the user with the largest score is the most representative one, while the user with the smallest score is a most conspicuous *outlier*. Therefore, suspicious attackers identified by PCA is those users who have entirely different rating styles with other users.

Intuitively, the outlier user is probably to have lower similarity with other users, i.e., exert smaller effects to other users. So shilling attackers acting like the outlier reduce their attack power to user-based CF recommender systems. PCA is apt to detect shilling attackers with smaller attack power. We design an experiment to illustrate this viewpoint. Two types attacks are selected including random attack with its *FS* = 5% on which PCA performed perfectly and the AoP on which PCA failed. Then, we define a new metric to measure the distance of a user to other users.

$$D_u = \sum_{v \epsilon U} d'_{u,v}\ , \text{ where } d'_{u,v} = 1 - PCC_{u,v} \qquad (7)$$

and $PCC_{u,v}$ is the Pearson correlation coefficient between u and v. Fig. 5 depicts D_u of attackers among normal users. As can be seen, random attackers with *FS* = 5% have the largest distances with other users, while AoP have the smallest distances. This observation conforms to the conclusion that AoP is the most effective attack [12]. Therefore, based on the above analysis on PCA, we can easily explain observations on the performance of PCA as shown in Table 2: (1) when *FS* is small, shilling attackers are probably to have low similarity with other users, i.e., D_u is large, and thus PCA performed well; (2) when selecting filler items among popular items, e.g., AoP, since the similarity with other users becomes higher, PCA failed; (3) since the average attack usually exerts heavier damage than the random attack, PCA performed better on detecting random attackers.



(a) Random 5%

(b) AoP

Figure 5. Illustration the success and failure cases of PCA.

Remark. In general, PCA is a stable shilling detector, that is, it can obtain satisfactory results in most cases. Since the nature of PCA is to catch outliers on rating styles, including item selection and rating assignment, it tends to be invalid when malicious users select popular items. More powerful PCA detector could be expected to be designed by incorporating discriminating features into the definition of covariance matrix. We should also note that the parameter *r* within PCA is difficult to be set in practice, since the number of attackers is unknown to us.

2) *MDS-based Detector*

The MDS-based detector consists of two phases. The Phase 1 extracts a number of *effective* users as the candidates for further identification of Phase 2. So, if the true shilling attackers cannot be retained by Phase 1, the recall will be decreased. Actually, the Phase 1 determines the maximal recall value of MDS. Table 4 lists the results of the Phase 1, where "*#Candidate*" is the number of effective users extracted by Phase 1 and the maximal recall R_{max} is

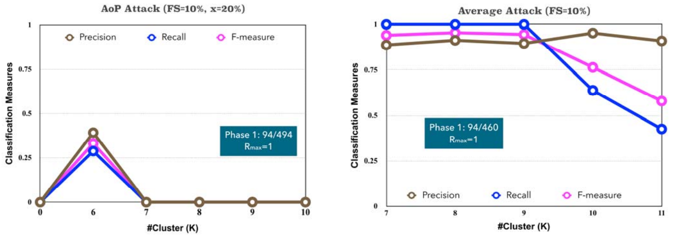
$$R_{max} = \frac{\#Retained\_Positive}{\#Positive}. \qquad (8)$$

In Eq. (8), *#Retained_Positive* is the number of attackers retained by Phase 1, and *#Positive* is the total number of injected attackers. We can see from Table 4 that MDS can effectively reserve average attackers rather than random attackers. For example, when *FS* = 5%, over 90% average attackers can be extracted yet all random attackers are lost. Recall that the "effective" users are those who have *high* relationships with other users [7]. Since the average attackers tend to hold higher similarity with other users, they are easily be retained. Meanwhile, as the increase of *FS*, attackers become more effective and thus R_{max} is increasing. For instance, nearly 50% random attackers with *FS* = 30% are retained by the first phase of MDS. Based on the above analysis, we conclude that in contrast with PCA, MDS is good at identifying attackers with higher attack power.

TABLE 4. STATISTIC ON PHASE 1 OF MDS.

Measure	***FS* = 5%**		***FS* = 10%**		***FS* = 30%**		***AoP***
	Ran	***Avg***	***Ran***	***Avg***	***Ran***	***Avg***	
#Candidate	423	513	425	460	507	465	494
#Retained_Positive	0	85	3	94	44	94	94
R_{max}	0	0.904	0.032	1	0.468	1	1

In Phase 2, MDS invokes K-means to divide candidate users into *K* clusters, among which the cluster with the largest $GRDMA_K$ is returned as the shilling group. We find that the MDS is very sensitive to the *K* value. An extreme example comes from detecting the AoP attack, as shown in Fig. 6(a). As can be seen, only when *K* = 6, MDS can find the cluster containing true attackers, though the Phase 1 performed perfectly. Fig. 6(b) shows another general case: as the increase of *K*, each cluster becomes smaller, less attackers are identified. The recall *R* will then be significantly reduced, which leads to the decrease of *F* though *P* is slightly increased.



(a) AoP

(b) Average 10%

Figure 6. Impact of *K* on the performance.

Remark. The philosophy of MDS is opposite to that of PCA. That is, it tries to seek shilling attackers among effective users, rather than outlier users. It is reasonable that if shilling attackers cannot influence other users, i.e., resulting in little damage to recommender systems, they could be left out. Moreover, MDS is really sensitive to the parameter *K*, and it is difficult to set *K* without the post-evaluation.

VI. Conclusions

This paper briefly reviews the state of the art in shilling attack generative models and supervised/unsupervised learning based detection algorithms. A thorough experimental comparison of four detectors is provided on MovieLens 100K dataset, including C4.5, NB, PCA and MDS. Under RFM and AFM, seven kinds of shilling attacks are tested in experiments. The experimental results show that supervised detectors can deal with all types of attackers and generally exhibit superior performance, with the help of labeled instances. Nonetheless, the stability of supervised detectors, especially C4.5, is undesirable. On the contrary, unsupervised detectors are much more stable, yet they tend to be noneffective for some special type of attackers. More interestingly, PCA and MDS form a complementary relationship, that is, the former is suitable for identifying attackers acting as outliers, while the latter is good at capturing very effective attackers.

Acknowledgment

This research was partially supported by National Natural Science Foundation of China under Grant 71372188, National Center for International Joint Research on E-Business Information Processing under Grant 2013B01035, National Key Technologies R&D Program of China under Grant 2013BAH16F03, Industry Project in Jiangsu S&T Pillar Program under Grant BE2014141, Key/Surface Projects of Natural Science Research in Jiangsu Provincial Colleges and Universities under Grants 12KJA520001, 14KJB520015, and 14KJA520001.

References

[1] B. Mobasher, R. Burke, and J. Sandvig, “Model-based collaborative filtering as a defense against profile injection attacks,” Proceedings of the 21st National Conference on Artificial Intelligence, pp. 1388-1393, July 2006.

[2] S. Lam, and J. Riedl, “Shilling recommender systems for fun and profit,” Proceedings of the 13th International Conference on World Wide Web, pp. 393-402, May 2004.

[3] C. Williams, “Profile injection attack detection for securing collaborative recommender systems,” Technical Report, DePaul University, 2006.

[4] I. Gunes, C. Kaleli, A. Bilge, and H. Polat, “Shilling attacks against recommender systems: a comprehensive survey,” Artificial Intelligence Review, 2012, Vol. 42, pp. 767-799.

[5] R. Burke, B. Mobasher, C. Williams, and R. Bhaumik, “Classification features for attack detection in collaborative recommendation systems,” Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 542-547, August, 2006.

[6] J. Cao, Z. Wu, B. Mao, and Y. Zhang, “Shilling attack detection utilizing semi-supervised learning method for collaborative recommender system,” World Wide Web, 2013,vol. 16, pp. 729–748.

[7] J. Lee, and D. Zhu, “Shilling attack detection—a new approach for a trustworthy recommender system,” INFORMS Journal on Computing, 2011, Vol. 21, pp. 117-131.

[8] H. Xia, B. Fang, M. Gao, H. Ma, Y. Tang, and J. Wen, “A novel item anomaly detection approach against shilling attacks in collaborative recommendation systems using the dynamic time interval segmentation technique,” Information Sciences, 2015, Vol 306, pp.150–165.

[9] B. Mobasher, R. Burke, R. Bhaumik, and C. Williams, “Towards trustworthy recommender systems: An analysis of attack models and algorithm robustness, ” ACM Transactions on Internet Technology, 2007, Vol. 7, pp. 1-41.

[10] P. Chirita, W. Nejdl, and C. Zamfir, “Preventing shilling attacks in online recommender systems,” Proceedings of the 7th ACM International Workshop on Web Information and Data Management, pp. 67–74, October 2005.

[11] S. Zhang, Y. Ouyang, J. Ford, and F. Makedon, “Analysis of a low-dimensional linear model under recommendation attacks,” Proceedings of the 29th ACM SIGIR International Conference on Research and Development in Information Retrieval, pp. 517-524, August 2006.

[12] N. Hurley, Z. Cheng, and M. Zhang, “Statistical Attack Detection,” Proceedings of the 3rd ACM conference on Recommender systems, pp. 149-156, September 2009.

[13] R. Bhaumik, B. Mobasher, and R. Burke, “A clustering approach to unsupervised attack detection in collaborative recommender systems,” Proceedings of the 7th IEEE International Conference on Data Mining, pp 181–187, December 2011.

[14] B. Mehta, T. Hofmann, and P. Fankhauser, “Lies and propaganda: detecting spam users in collaborative filtering,” Proceedings of the 12th International Conference on Intelligent User Interfaces, pp.14-21, January 2007.

[15] S. Zhang, A. Chakrabarti, J. Ford, and F. Makedon, “Attack detection in time series for recommender systems,” Proceedings of the 12th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 809-814, August 2006.

[16] C. Li, and Z. Luo, A Hybrid Item-based Recommendation Algorithm against Segment Attack in Collaborative Filtering Systems. 2011 International Conference on Information Management, Innovation Management and Industrial Engineering, pp.403-406, 2011.

[17] Z. Wu, J. Wu, J. Cao, and D. Tao, “HySAD: A Semi-Supervised Hybrid Shilling Attack Detector for Trustworthy Product Recommendation,” Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, pp. 985-993, August 2012.

[18] N. Günnemann, S. Günnemann, and C Faloutsos, “Robust Multivariate Autoregression for Anomaly Detection in Dynamic Product Ratings,”, Proceeding of the 23rd World Wide Web Conference, pp. 361-371, October 2014.

[19] B. Mobasher, R. Burke, R. Bhaumik and J. J. Sandvig, “Attacks and remedies in collaborative recommendation,” IEEE Intelligent Systems, 2007, vol. 22, pp. 56-63.

[20] M. O’Mahony, N. Hurley, and G. Silvestre, “Detecting noise in recommender system databases,” Proceedings of the International Conference on Intelligent User Interfaces, pp. 109–115, January 2006.

[21] B. Mehta, M. Nejdl, “Unsupervised strategies for shilling detection and robust collaborative filtering,” User Modeling and User-Adapted Interaction, 2009, Vol. 19, pp. 65-97.

[22] K. Bryan, M. O'Mahony, and P. Cunningham, “Unsupervised retrieval of attack profiles in collaborative recommender systems,” Technical Report, University College Dublin, 2008.

[23] A. Buja, D. Swayne, M. Littman, N. Dean, H. Hofmann, and L. Chen, “Data visualization with multidimensional scaling,” Journal of Computational and Graphical Statistics, 2008, Vol. 17, pp. 444-472.