World Wide Web (2013) 16:729–748
DOI 10.1007/s11280-012-0164-6

# Shilling attack detection utilizing semi-supervised learning method for collaborative recommender system

**Jie Cao · Zhiang Wu · Bo Mao · Yanchun Zhang**

Received: 13 June 2011 / Revised: 28 March 2012 /
Accepted: 5 April 2012 / Published online: 26 April 2012
© Springer Science+Business Media, LLC 2012

**Abstract** Collaborative filtering (CF) technique is capable of generating personalized recommendations. However, the recommender systems utilizing CF as their key algorithms are vulnerable to shilling attacks which insert malicious user profiles into the systems to push or nuke the reputations of targeted items. There are only a small number of labeled users in most of the practical recommender systems, while a large number of users are unlabeled because it is expensive to obtain their identities. In this paper, *Semi-SAD*, a new semi-supervised learning based shilling attack detection algorithm is proposed to take advantage of both types of data. It first trains a naïve Bayes classifier on a small set of labeled users, and then incorporates unlabeled users with EM-λ to improve the initial naïve Bayes classifier. Experiments on MovieLens datasets are implemented to compare the efficiency of *Semi-SAD* with supervised learning based detector and unsupervised learning based detector. The results indicate that *Semi-SAD* can better detect various kinds of shilling attacks than others, especially against obfuscated and hybrid shilling attacks.

**Keywords** semi-supervised learning · shilling attack detection · collaborative filtering · naïve Bayes · EM

J. Cao · Z. Wu (✉) · B. Mao
Jiangsu Provincial Key Laboratory of E-Business, Nanjing University of Finance and Economics, Nanjing, China
e-mail: zawu@seu.edu.cn

J. Cao
e-mail: caojie690929@163.com

B. Mao
e-mail: maoboo@gmail.com

Y. Zhang
School of Computer Science & Mathematics, Victoria University, Melbourne, Australia
e-mail: Yanchun.Zhang@vu.edu.au

## 1 Introduction

The prevalence of Web 2.0 technologies leads to a great increasing of real time generated data in various formats, such as text, XML, multimedia, time series, video, image, and many others [22]. Therefore, it is getting harder for the common users to acquire the information they really want timely and precisely. Recommender systems utilize the opinions from a community to help the users in the same community to identify their interested content from an overwhelming set of potential choices [3, 5]. Collaborative filtering (CF) is one of the most successful technologies used in the recommender systems, from the e-business sites (Amazon.com), news, information sites (dig and slashdot) to the social networking sites (Facebook).

The filtering process of CF is based on the profiles of other users, so the recommender systems using CF are vulnerable to shilling attacks, in which attackers benefit themselves by introducing biased rating profiles [9]. Suppliers have motivation to utilize it to promote their own products and to bespatter their competitors. For example, an author, motivated by selling more of his book, might create a collection of shilling profiles rating the book highly to artificially promote the ratings given to genuine users by the system. That will reduce the accuracy and damage the faith of the users in the recommender system. Therefore, how to detect the shilling attacks is a big challenge in the studies of recommender systems.

Some e-commerce web sites, e.g. Amazon and Taobao, have quite many users but only a small number of the users can be identified as normal users or shilling attackers. For instance, in Taobao, crown shop owners and buyers with high positive ratings can be identified as normal users, and the users with fraud history or very low positive ratings can be identified as shilling attackers. But the natures of other users with moderate positive ratings cannot be identified without manual analysis. The identified users are called *labeled data*, and others are *unlabeled data*. Generally, the unlabeled data *alone* is insufficient to yield better-than-random classification because there is no information about the class label [4]. But the unlabeled data do contain the information about the joint distribution of the data features. Therefore, methods of exploiting such unlabeled data can save both time and cost greatly.

Semi-supervised learning that began in the mid-1990s is a kind of machine learning (ML) technique which can effectively exploit both unlabeled data and labeled data [17]. It has attracted much attention from different fields, such as text classification, web page classification, medicinal image analysis, etc. In this paper, the semi-supervised learning will be applied in the shilling attack detection. There are three kinds of semi-supervised learning algorithms: generative model, transductive inference and co-training paradigm [25]. The most straightforward algorithms use a generative model for the classifier and employ Expectation Maximization (EM) to model the label estimation [20]. In this paper, based on generative model, the proposed approach utilizes naïve Bayes as the initial detector and EM-$\lambda$, an augmented EM, to improve the detector.

As described before, it is common that the number of labeled users is much smaller than unlabeled users. In that case, the great majority of the data determining EM's parameter estimates comes from unlabeled data. Therefore, EM is almost equivalent to an unsupervised clustering. Because labeled users are valuable for shilling attack detection, we add a weighting factor $\lambda(0 \leq \lambda \leq 1)$ to decrease the impact of unlabeled data for EM, which is called EM-$\lambda$ [16]. Experiments on the MovieLens

datasets show that carefully selecting λ could significantly improve the performance of detector, especially for the case with small number of labeled users.

Most previous shilling attack detection methods use supervised learning, and few methods use unsupervised learning. To the best of our knowledge, there is none shilling attack detection method utilizing semi-supervised learning except our abbreviated report [23]. In this paper, we propose to utilize semi-supervised learning to identify attack profiles and describe how to apply semi-supervised learning to shilling attack detection in detail. We evaluate the algorithm in scenarios of various obfuscated attacks and hybrid attacks.

The rest of the paper is organized as follows. Section 2 reviews the related work on CF algorithms, shilling attacks classification and shilling attack detection methods. Section 3 presents data format and problem statement. In Section 4, Semi-supervised learning based Shilling Attack Detection (Semi-SAD) algorithm is described in details. In Section 5, we demonstrate the effectiveness and efficiency of our Semi-SAD algorithm by experiments. Section 6 summarized the whole paper.

## 2 Related work

CF algorithm is a widely used recommendation algorithm. But the recommender systems using CF algorithm are vulnerable to the shilling attack. Meanwhile, malicious users can utilize different methods to create shilling attack profiles and employ obfuscated techniques to make the detection of attack more difficult. Section 2.1 reviews three different types of CF algorithms, Section 2.2 introduces the classification of shilling attacks and various obfuscated techniques, and Section 2.3 discusses existing detection methods for shilling attacks and compares these methods with the proposed method.

### 2.1 Collaborative filtering (CF) algorithms

The initial algorithm used by recommender systems is the Content-based algorithm which recommends the similar items preferred by user in the past [14]. Content-based algorithm relies on rich content descriptions of the items or services. For instance, if a system wants to recommend a coat, the Content-based algorithm requires the information about brand, style, color, price, etc. Therefore, the designer of the Content-based algorithm has to obtain abundant domain knowledge which may not be readily available or not straightforward to maintain.

Different from the content-based algorithm, CF algorithm collects opinions from users in the form of ratings on items. Its advantage over the Content-based algorithm is that only historical information of users is required instead of the representation of the items. CF algorithms can be classified into three categories [1, 24]: User-based CF (UCF), Item-based CF (ICF) and Hybrid CF (HCF). Any CF algorithm consists of two phase: (1) compute the similarity and find $k$ nearest neighbors ($k$NN) of user or item; (2) predict on the basis of $k$ nearest neighbors. The similarity computation often employs Pearson correlation coefficient (PCC) as shown in (1).

$$sim(u, v) = \frac{\sum_{a \in I}(r_{ua} - \overline{r_u})(r_{va} - \overline{r_v})}{\sqrt{\sum_{a \in I}(r_{ua} - \overline{r_u})^2 \sum_{a \in I}(r_{va} - \overline{r_v})^2}} \tag{1}$$

*k*NN often removes the top *k* similar neighbors from the set, if their similarities are equal or smaller than 0. It will enhance the prediction quality, so it is adopted in this paper.

The first phases of the three kinds of CF algorithms are same, and the discrimination lies in their second phases described as below.

– **UCF** Prediction is generated based on *k* nearest neighbors of the active user. $p_{u,i}$ represents the predicted rating of user $u$ on item $i$ and can be calculated by (2).

$$p_{u,i} = \overline{r_u} + \frac{\sum_{v \in N_{u,i}} sim(u, v)(r_{vi} - \overline{r_v})}{\sum_{v \in N_{u,i}} sim(u, v)} \tag{2}$$

where $N_{u,i}$ is user $u$'s top $k$ neighbors with respect to item $i$ and consists of $k$ users who have rated $i$ and have the greatest PCC with $u$.

– **ICF** On the basis of *k* nearest neighbors of item, the formula used to compute the prediction rating of user $u$ on item $i$ is:

$$p'_{u,i} = \frac{\sum_{j \in N_i} sim(i, j) r_{uj}}{\sum_{j \in N_i} sim(i, j)} \tag{3}$$

– **HCF** HCF method synthesizes UCF and ICF by a weighting model. The prediction rating is computed by (4).

$$p_{u,i} = \varphi p_{u,i} + (1 - \varphi) p'_{u,i} \tag{4}$$

When the value of $\varphi$ increases, HCF squints towards UCF. Conversely, HCF is altered to squints towards to ICF. If $\varphi$=1 HCF is altered to UCF and if $\varphi$=0 HCF is altered to ICF.

The above-mentioned HCF method combines UCF and ICF. Another hybrid recommender system combines content-based algorithms and CF methods. For instance, Gunawardana et al. utilize unified Boltzmann machines to encode collaborative and content information as features, and then learn weights reflecting how well each feature predicts user actions [7]. Manouselis et al. have employed multi-criteria decision making methods to facilitate recommendation [12]. One recent interesting extension for CF is the use of sentiment analysis to augment ratings for performing CF [11]. Combing sentiment analysis with CF can utilize user-generated reviews in the context of the article instead of numerical ratings.

### 2.2 Shilling attacks classification

From the intention perspective, shilling attacks can be divided into *push* and *nuke* attacks to make a target item more or less likely to be recommended respectively. There is economic motivation for the attacker to carry out push attack on their own items, or pose nuke attack on others. Burke et al. [2] described five models for generating shilling attack profiles. Figure 1 and Table 1 describe these five attack strategies for push and nuke intent.

A shilling profile consists of three parts: target item, filler items and non-voted items. Target item is often assigned the highest rating in a push attack, or the lowest in a nuke attack. Filler items are a set of items that make the shilling profile look normal and yield a bigger impact against a system. Quality of the filler items depends on the

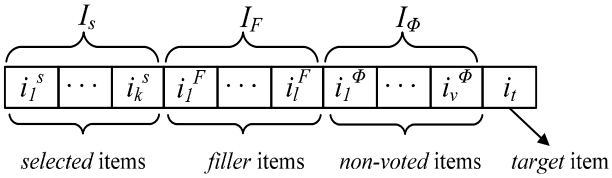



**Figure 1** The general form of a user profile.

knowledge about the recommender system. The more the knowledge is obtained, the more sophisticated shilling attack can be generated. Non-voted items are the remaining unrated items. The difference of five shilling attack models is the rating method for filler items.

Williams et al. proposed three obfuscated attack models, *noise injection*, *user shifting* and *target shifting*, for shilling attackers in order to make the detection more difficult [19]. Since *user shifting* is the general expression of *noise injection*, *noise injection* and *target shifting* are selected in this paper to obfuscate the attack profiles. These two techniques are described as follows.

- **Noise injection.** Add a Gaussian distributed random number multiplied by to each rating within a subset of filler item set. This noise can be used to blur the profile characteristic that are often associated with the known attack types.
- **Target shifting.** For the push attack, the technique shifts the rating given to the target item from the maximum rating to a rating one step lower; or for the nuke attack, it increases the target rating to one step above the lowest rating.

Apart from above-mentioned obfuscated techniques, Hurley et al. presented a simple and effective strategy to obfuscate the average attack, *Average over Popular items (AoP)* [8]. AoP $x$% attack chooses filler items with equal probability from the top $x$% of most popular items, rather than from the entire catalogue of items.

We have noticed that besides the above-mentioned attacks, hybrid shilling attack, which injects various types of shilling attacker profiles into the recommender system

**Table 1** Five shilling attack models for push and nuke intent.

| Attack models | Push | Nuke |
|---|---|---|
| Random attack | $I_s = \emptyset$; give $I_F$ random ratings; $i_t = r_{\max}$ | $I_s = \emptyset$; give $I_F$ random ratings; $i_t = r_{\min}$ |
| Average attack | $I_s = \emptyset$; the ratings for $I_F$ are distributed around the mean for each item $i$; $i_t = r_{\max}$ | $I_s = \emptyset$; the ratings for $I_F$ are distributed around the mean for each item $i$; $i_t = r_{\min}$ |
| Segmented attack | $I_s$ are the target item's similar items and $I_s = r_{\max}$; $I_F = r_{\min}$; $i_t = r_{\max}$ | $I_s$ are the target item's similar items and $I_s = r_{\min}$; $I_F = r_{\max}$; $i_t = r_{\min}$ |
| Bandwagon attack | $I_s$ are the frequently rated items and $I_s = r_{\max}$; give $I_F$ random ratings; $i_t = r_{\max}$ | $I_s$ are the frequently rated items and $I_s = r_{\min}$; give $I_F$ random ratings; $i_t = r_{\min}$ |
| Sampling attack | $I_s = \emptyset$; copy a existing user profile as $I_F$; $i_t = r_{\max}$ | $I_s = \emptyset$; copy a existing user profile as $I_F$; $i_t = r_{\min}$ |

simultaneously, also cannot be detected easily. Previous work has not considered detection against the hybrid shilling attack, and the existing algorithms also do not evaluate their performance under the hybrid shilling attacks.

### 2.3 Shilling attack detection methods

The detection algorithm of shilling attack aims to construct a set of abnormal activity. If the generated set is consistent with shilling attack, the detected abnormal activity source will be considered as shilling attacker.

Lam et al. had evaluated the impact of random attack and average attack on user-based and item-based CF algorithms [9]. The first shilling detection algorithm based on features of shilling profiles was proposed by Chirita et al. [6] for random attack and average attack. Then, Mobasher et al. proposed a detection algorithm for segmented attack [15]. Except random attack, shilling attacks require knowledge of the system which is available in recommender systems. For instance, MovieLens and Internet Movie Database (IMDb) publish the mean of all items and the user profiles. Therefore, attackers are able to use the open knowledge to generate various sophisticated shilling attacks. Detection algorithm against these shilling attack models should be further studied. Hurley et al. propose to use Neyman-Pearson statistical detection to identify attack profiles [8].

From the machine learning perspective, most previous shilling detection algorithms are the supervised learning approaches, such as algorithms reported in [2, 6, 15, 19]. Three disadvantages of the supervised detection algorithms are:

1. They need a large number of training data, and can only detect the profiles similar to the profiles in the training data. They are successful in detecting the attacks with dense attacker profiles, but unsuccessful against attacks which are small in profile size or have high sparsity.
2. They cannot cope with the obfuscated shilling profiles if the training data does not include various obfuscated examples.
3. They perform badly against hybrid shilling attacks because different shilling attack types have different characteristics and the mixture will misguide the training of detector.

Mehta et al. have proposed an unsupervised shilling attack detection algorithms using *principal component analysis* (PCA) [13]. Hurley et al. have reported that PCA-based detector is ineffective to AoP attack [8], which reveals the fact that the unsupervised learning approach does not make good use of the labeled data which indeed exists and is significant for the detection. Recently, Lee et al. proposed a new detector using a clustering method and the Group RDMA (GRDMA) metric [10].

Therefore, semi-supervised approach has great potentiality to build an effective shilling attack detector by exploiting both labeled data and unlabeled data. Most of the popular semi-supervised approaches employ the Expectation Maximization (EM) to model the parameter estimation process. EM-$\lambda$, an augmented EM proposed by Nigam et al. [16], modulates the influence of the unlabeled data by adding a weighting factor $\lambda$ to estimation. EM and Naïve Bayes are two of the most influential algorithms that have been widely used in the data mining community [20]. Naïve Bayes is a supervised classification method and it is very easy to construct, not needing any complicated iterative parameter estimation schemes. In this paper, we

combine Naïve Bayes and EM-λ to build a new shilling attack detector exploiting both labeled data and unlabeled data.

## 3 Data format and problem statement

Before performing classification on user profiles, we pre-process each user profile to format its features. The proposed shilling detection algorithms will perform on the formatted data as shown in Figure 2. UID uniquely identifies a user and is the main key of a profile. There are two possible classes for each profile: *Shilling (S)* means that the detector has determined that the profile is an instance of an attack profile and *Normal (N)* means that the detector has determined that the profile is that of a genuine system user. For training data, the class label is known. For testing data, the class label is unknown and will be determined by the detection algorithms. Metrics are used to describe the characteristics of the normal users or the shilling attackers. No single metric can distinguish the normal users from the shilling attackers entirely. Therefore, we should select and define a series of metrics.

### 3.1 Definition of detection metrics

Previous work has defined a multitude of detection metrics [2, 6, 19]. In this paper, the selected metrics are based on the aim and essence of shilling attackers, which means that if an attacker does not violate these metrics, its aim will not be achieved or its power of influence to the recommender system will be reduced.

The first metric is *entropy* that measures the degree of dispersal or concentration of user profiles. Shilling attackers often generate random ratings with Gaussian distribution. For instance, in MovieLen dataset with five-point, shilling attackers utilize the Gaussian distribution with the mean 3.6 and standard deviation 1.1 [9]. Therefore, the *entropy* of shilling attackers tends to be smaller than the normal users. The definition of *entropy* is given in Definition 1.

**Definition 1** (Entropy) Let $X_u = \{n_i, i = 1, 2, \cdots, r_{\max}\}$ be an statistical set of user $u$, where $n_i$ means the occurring time of the $i$-th possible rating in the $u's$ profile. The entropy $H(u)$ can be computed as (5).

$$H(u) = -\sum_{i=1}^{r_{\max}} \frac{n_i}{S} \log_2 \frac{n_i}{S} \quad where \quad S = \sum_{i=1}^{r_{\max}} n_i \tag{5}$$

The value of $H(u)$ is in [0,$\log_2 r_{\max}$]. The minimum value 0 means all ratings are the same ($r_{\max} = 1$) and the maximum value $\log_2 r_{\max}$ indicates $n_i = n_j$ for all $i,j$.

According to [13], the influence to a recommender system is maximized if the PCC of the shilling attacker is maximized with all the users' profiles. We define *Degree of Similarity with Top Neighbors* (DegSim) to calculate the average similarity of a profile's $k$ nearest neighbors.

**Figure 2** Data format after pre-processing.

| **UID/Profile** | **Class** **S/N** | **Metric 1** *H(X)* | **Metric 2** *ADegSim* | …… | **Metric *m*** *RDMA* |
|---|---|---|---|---|---|

**Definition 2** (DegSim) The average similarity of $u$'s $k$ nearest neighbors can be computed as:

$$DegSim_u = \frac{\sum_{v=1}^{k} sim_{uv}}{k} \tag{6}$$

Shilling attackers can add rated items to increase DegSim. Chirita had discovered that DegSim resembles *Number of Prediction-Differences* (NPD) defined for each user as the number of net prediction changes in the system after her removal from the system [6]. Therefore, we only select one metric between DegSim and NPD.

*Length Variance* (LengthVar) is introduced to measure the difference between the length of a given profile and the average length among all user profiles.

**Definition 3** (LengthVar) Let $\#(Pu)$ be the length of $u$, and $N$ be the total number of user profiles. $LengthVar_u$ can be computed as:

$$LengthVar_u = \frac{|\#(P_u) - \overline{\#}|}{\sum_{i=1}^{N} (\#(P_u) - \overline{\#})^2} \tag{7}$$

*Rating Deviation from Mean Agreement* (RDMA) proposed by Chirita [6] identifies attackers through examining the average deviation per item, weighted by the reciprocal of the rating number of that item.

**Definition 4** (RDMA) Let $NR_i$ be the number of ratings provided for item $i$ by all users, $N_u$ be the number of items rated by $u$, $RDMA_u$ can be computed as:

$$RDMA_u = \frac{\sum_{i=0}^{N_u} \frac{|r_{ui} - \overline{r_i}|}{NR_i}}{N_u} \tag{8}$$

Bandwagon and segment attackers give a group of items the highest ratings, which leads to the large difference between average rating of items with the highest ratings and average rating of other rated items. Therefore, we define the *Filler Mean Target Difference* (FMTD) metric which can be computed as:

**Definition 5** (FMTD) Let $P_{uT}$ be the item set with the highest rating, and $P_{uF}$ be the remaining rated item set. $FMTD_u$ is given as:

$$FMTD_u = \left| \left( \frac{\sum_{i \in P_{uT}} r_{ui}}{|P_{uT}|} \right) - \left( \frac{\sum_{j \in P_{uF}} r_{uj}}{|P_{uF}|} \right) \right| \tag{9}$$

For segmented attack profile $u$, $FMTD_u = 4$ and FMTD is effective to the segmented attack.

### 3.2 Problem statement

This section presents the formal statement for shilling attack detection problem. The original input is the set of rating records of all users. Let $L$ denote the labeled user profile set with size $|L|$ $L = \{(u_1, c_1), (u_2, c_2), \cdots, (u_{|L|}, c_{|L|})\}$, and $U$ denote unlabeled user profile set with size $|U|$ $U = \{u'_1, u'_2, \cdots, u'_{|U|}\}$. If the class label of a

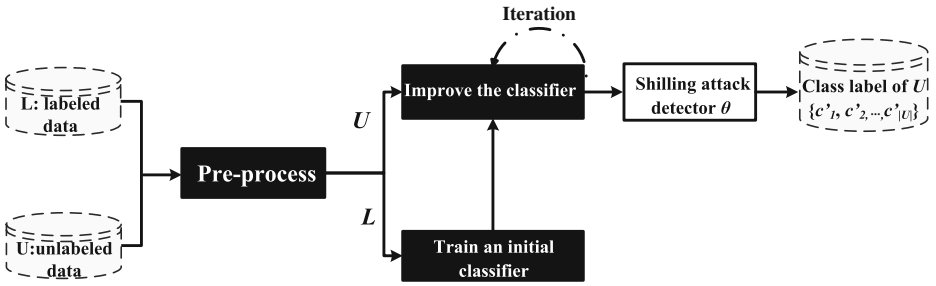



**Figure 3** The steps involved in shilling attack detection.

user (shilling or normal) is known, the user belongs to $L$, otherwise the user belongs to $U$. There is a one-to-one correspondence between any user in $U$ and its class label. The class labels of $U$ is written $\{c'_1, c'_2, \cdots, c'_{|U|}\}$ which is the output of shilling detection algorithms.

The shilling attack detection algorithms try to learn the function $\theta : U \rightarrow C$ in order to accurately predict class label $c$ for any user profile $u$. $u_i, u'_j \in U$ are the UID as shown in Figure 2. $c_i \in C$ is the class label categorized into *normal(N)* and *shilling(S)*.

Figure 3 summarizes the steps involved in shilling attack detection problem. Note that not all steps can take place in any single particular scenario. For example, supervised learning based detectors do not need unlabeled data set and there is no step for improving the classifier, then the initial classifier trained on labeled data set is taken as the final detector. Conversely, there is no step for training an initial classifier, and we can think the step for improving the classifier as performing unsupervised clustering.

## 4 Semi-supervised learning based Shilling Attack Detection (Semi-SAD) algorithm

The main contribution of this paper is how semi-supervised learning can be utilized to detect shilling attacks. The proposed Semi-SAD algorithm consists of two phases. The first phase trains a naïve Bayes classifier on a small set of labeled data and the second phase incorporates unlabeled data with EM-λ to improve the initial naïve Bayes classifier.

### 4.1 Phase 1: Training a Naïve Bayes classifier on labeled data

This section presents how to utilize naïve Bayes, a well-known probabilistic classifier, to train an initialized classifier on labeled data. Naïve Bayes is the foundation upon which we will later build the method to incorporate unlabeled data.

The naïve Bayes classifier is employed to estimate the probability of an unknown user profile belonging to a certain class. In shilling attack detection, a class can be categorized into *normal(N)* and *shilling(S)* with label written in $C = \{S, N\}$. All metrics defined in Section 3.1 are successive values. We assume the $i$-th metric $M_i$ follows the Gaussian probability distribution with mean $\mu_i$ and standard deviation

$\sigma_i$. $P(x_i|C)$ is the probability of a user profile belonging to the class $C$ if its $i$-th metric $M_i = x_i$.

$$P(x_i|C) = g(x_i, \mu_{Ci}, \sigma_{Ci}), \textit{where} \quad g(x, \mu, \sigma) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(x-\mu)^2}{2\sigma^2}} \tag{10}$$

In (10), $\mu_{Ci}$ and $\sigma_{Ci}$ are the mean and standard deviation of the $i$-th metric of the labeled data in class $C$. $g(x, \mu, \sigma)$ is the probability density function of Gaussian distribution. For the user profile $u$ in the data format of Figure 2, the probability of $u$ belonging to the class $C$ is given by (11).

$$P(u|C) = \prod_{i=1}^{n} P(x_{ui}|C) \tag{11}$$

With (10) and (11), we can calculate the probability of an unknown user profile belonging to a certain class. As a matter of fact, the labeled data is utilized to determine the parameters, i.e. mean and standard deviation, of probability distribution of each class.

### 4.2 Phase 2: Incorporating unlabeled data with EM-λ to improve the classifier

The advantage of semi-supervised learning method over supervised learning method is that the unlabeled data is exploited to improve the classifier. Especially when the labeled data is sparse, the parameter variance from the naïve Bayes classifier is so big that the accuracy of shilling attack detection is unacceptable.

EM algorithm is a widely used approach to exploit the unlabeled data. EM-λ, an augmented EM proposed by Nigam et al. [16], modulates the influence of the unlabeled data by adding a weighting factor λ in the estimation. EM iteratively re-estimates parameters by repeating two steps (E-step and M-step) until converging to a stationary value for the estimation. The estimated parameters in shilling attack detection are the mean and standard deviation of both normal class and shilling class.

- E-step: Calculate the probability of each user profile belonging to a class from (12).

$$P(u_k \in C) = P(c|u_k) = \frac{P(C)\,P(u_k|C)}{P(u_k)} \tag{12}$$

  The class probability *P(C)* represents *P(N)* and *P(S)* which are either taken to be uniform, or estimated by counting the number of times each label appears in the labeled data. *P(u)* is taken to be constant. Therefore, $P(u_k \in C)$ is only determined by $P(u_k|C)$ that can be get from (10) and (11).
- M-step: Calculate the estimated parameters based on the probability calculated in E-step. We have the mean of the $i$-th metric on the data belonging to class $C$ as follows:

$$\mu_{Ci} = \frac{1}{|C|} \Sigma_{u=1}^{|C|} \omega_u x_{ui} \tag{13}$$

The standard deviation of the $i$-th metric on the data belonging to class $C$ can be computed as:

$$\sigma_{Ci} = \sqrt{\frac{1}{|C|}\Sigma_{u=1}^{|C|}\omega_u^2(x_{ui} - \mu_{Ci})^2} \tag{14}$$

Note that in (13) and (14), the number of each class $|C|$ is also computed by adding weight $\omega_u$. $|C|$ represents $|N|$ and $|S|$ computed by (15).

$$|C| = |L_C| + \Sigma_{u=1}^{|U|}\omega_u \tag{15}$$

In (15), $|L_C|$ is the number of users labeled $C$ in $L$. The probability of a user belonging to certain class ($N$ or $S$), is employed as a weighting factor in the estimation. In (13)–(15), $\omega_u$ can be computed by (16):

$$\omega_u = \frac{P(u \in C)}{\sum_j P(u \in C_j)} \tag{16}$$

In labeled data set, $\omega_u$ is either 0 or 1. EM-λ has the same E-step as EM, but the M-step is different with the following entail for (16). Then $\Lambda(u)$ is defined to be the weighting factor if $u$ is in the unlabeled set, and 1 if $u$ is in the labeled set:

$$\Lambda(u) = \begin{cases} \lambda & if\ u \in U \\ 1 & if\ u \in L \end{cases} \tag{17}$$

Then, we can rewrite (18) as:

$$\omega_u = \Lambda(u)\frac{P(u \in C)}{\Sigma_j P(u \in C_j)} \tag{18}$$

The weighting factor λ is used to adjust the impact of unlabeled data to estimation. It is obvious that as λ is decreasing, the unlabeled data will have less influence to the shape of EM's hill-climbing surface. When $\lambda = 1$, each unknown user profile will be weighted as known user profiles, and EM-λ squints towards to the original EM algorithm.

### 4.3 Pseudo-code of Semi-SAD

Semi-SAD algorithm is designed to integrate the naïve Bayes classifier presented in Section 4.1 with the EM-λ in Section 4.2. The pseudo-code of Semi-SAD is given in Algorithm 1.

Semi-SAD starts with the labeled user profile set, $L$, and the unlabeled user profile set, $U$. The user profile is the user's rating for some items, for example, a user rates movies he/she watched in the MovieLen dataset. The naïve Bayes classifier are trained on the labeled user profile set, $L$, as the initial classifier. Then, Semi-SAD starts the iteration of E- and M-steps until the estimated parameters become stable. In practice, we define that the convergence has occurred if observing a below-threshold change of (13) and (14). Once the iteration stops, the initial classifier has been improved by EM-λ, and returned it as the shilling attack detector.

**Algorithm 1** Pseudo-code of the Semi-SAD algorithm.

**Input:**

*L:* Labeled user profile set

*U:* Unlabeled user profile set

**Output:**

A shilling attack detector, $\theta$ that takes an unknown user profile and predicts a class label.

1: Take pre-process on $L$ and $U$ to derive the data format shown in Figure 2.
2: Train an initial naïve Bayes classifier, $\theta$, based on labeled user profile set, $L$, only.
3: **repeat until** none of estimated parameters ($\mu_{Ci}$ and $\sigma_{Ci}$) changes
4: (E-step) Utilize the current classifier, $\theta$, to calculate the probability of each user profile belonging to each class from (12).
5: (M-step) Improve the current classifier, $\theta$, by utilizing (13)–(18).
6: **end of repeat**
7: Take the classifier, $\theta$, to be the shilling attack detector.
8: Compute the ratio

$$\Omega = ln\frac{P(S|u)}{P(N|u)} = ln\frac{P(S)}{P(N)} + \Sigma_{k=1}^{n} ln\frac{P(x_k|S)}{P(x_k|N} \tag{19}$$

9: **for** all user profiles in $U$
10: **if**($\Omega > \eta$) the user is Shilling attacker.
11: **else** the user is Normal user.
12: **end of for**

We employ *ln* function, a monotonic increasing function, to compute the ratio $\Omega$. For any unknown user profile, the shilling attack detector can predict its class label by comparing the ratio $\Omega$ with the threshold $\eta$. The threshold $\eta$ is selected through an empirical evaluation in practice and can be employed to obtain the *Receiver Operating Characteristic (ROC)* curve.

## 5 Experiments on MovieLens datasets

MovieLens published by GroupLens datasets[1] are used in the experiments. This dataset consists of 100,000 ratings on 1682 movies by 943 users. All ratings are integer values between one and five where one is the lowest (disliked) and five is the highest (most liked). The push attack is demonstrated in the experiments. Table 2 shows the generation method of five types of push attacker profiles. Then, these shilling attack profiles are injected into MovieLens datasets.

We continue to utilize two staple metrics *specificity* and *sensitivity* to evaluate the performance of Semi-SAD. To define *specificity* and *sensitivity* explicitly, a two by two confusion matrix is defined for shilling attack detection problem as shown in Figure 4.

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

**Table 2** Generation methods for five shilling attack models.

| Attack models | Generation methods |
|---|---|
| Random attack | rate 5 to a target item; give filler items random ratings conforming to a Gaussian distribution with mean 3.6 and standard deviation 1.1 |
| Average attack | rate 5 to a target item; the ratings for filler items are distributed around the mean for each item |
| Sampling attack | rate 5 to a target item and its similar items; rate 1 to filler items |
| Segmented attack | rate 5 to a target item and 20 bandwagon movies which are those with the most ratings in the dataset; give filler items random ratings conforming to a Gaussian distribution with mean 3.6 and standard deviation 1.1 |
| Bandwagon attack | copy existing user profiles including maximum rating records; rate 5 to a target item |

Based on the confusion matrix, *specificity* and *sensitivity* are defined in (20) and (21) [18]:

$$sensitivity = \frac{\#true\ positives}{\#true\ positives + \#false\ negatives} \tag{20}$$

$$specificity = \frac{\#true\ negatives}{\#true\ negatives + \#false\ positives} \tag{21}$$

The *specificity* measures the proportion of correctly identified normal profiles, and the *sensitivity* measures the proportion of correctly detected attack profiles.

We compare Semi-SAD algorithm with supervised and unsupervised learning based algorithms. The supervised learning based algorithm utilizing naïve Bayes to train a detector on labeled data, and this detector is named Bayes-SAD. PCA-based detector proposed in [13] is selected as the representative of unsupervised algorithm, and this detector is named PCA-SAD.

### 5.1 Performance of Semi-SAD against obfuscated single shilling attack

We divide the datasets into training dataset and test dataset. For Semi-SAD, training dataset is relatively small and consists of only 100 normal users and 50 shilling attackers. For Bayes-SAD, training dataset contains 943 normal users and 94 shilling attackers, and the attack size is set to 10% in the experiments. PCA-SAD detects shilling attacks based only on test dataset. All three algorithms can effectively detect

| | | Predicted | |
|---|---|---|---|
| | | Shilling Profile | Normal Profile |
| Actual | Shilling Profile | true positives | false negatives |
| | Normal Profile | false positives | true negatives |

**Figure 4** Confusion matrix for shilling attack detection problem.

**Table 3** Comparison of detection performance against *obfuscated random attacks*.

| Filler size | | 5% | 10% | 20% | 30% | 40% | 50% |
|---|---|---|---|---|---|---|---|
| *Sensitivity* | Semi-SAD | 0.957 | 0.978 | 0.989 | 1 | 1 | 1 |
| | Bayes-SAD | 0.755 | 0.857 | 0.798 | 0.744 | 0.766 | 0.777 |
| | PCA-SAD | 0.925 | 0.942 | 0.953 | 0.968 | 0.982 | 0.976 |
| *Specificity* | Semi-SAD | 0.985 | 0.982 | 0.987 | 0.991 | 0.997 | 0.998 |
| | Bayes-SAD | 0.982 | **0.984** | 0.983 | 0.965 | 0.935 | 0.922 |
| | PCA-SAD | 0.898 | 0.918 | 0.926 | 0.924 | 0.936 | 0.965 |

single shilling attack because the profiles in test dataset are same as those in training dataset. Therefore, our experiments focus on the evaluation of the performance against obfuscated single shilling attacks.

Shilling attack profiles injected into training dataset are constructed according to Table 2. Then, noise injection and target shifting as mentioned in Section 2.2 are utilized to obfuscate the shilling attack profiles injected into the test dataset. Sensitivity and specificity of three algorithms against five obfuscated shilling attacks are listed in Tables 3, 4, 5, 6 and 7, according to which, Bayes-SAD has the lowest sensitivity because shilling user profiles in training dataset are not obfuscated and then supervised learning technique cannot identify obfuscated shilling attackers effectively. Since normal user profiles in training dataset and test dataset are similar, the specificity of Bayes-SAD is still passable.

Tables 3–7 also show that sensitivity and specificity of Semi-SAD are higher than those of PCA-SAD. Shilling attack profiles with more filler items are readily identified by Semi-SAD, and Semi-SAD is verging to a ideal classifier with the increase of filler size. Shilling attackers with more filler items will have more neighbors and be more effective to recommender systems. Hence, the attacker should make a compromise between detectability and attack performance. Moreover, since sampling attack does not contain filler items, we vary the length of shilling attackers in Table 7, which indicate that the shilling attack will be more effective and detectable if the attacker copies longer profiles.

### 5.2 Performance of Semi-SAD against AoP attack

AoP attack introduced in Section 2.2 is a simple and effective obfuscated average attack. Experiments in Section 5.1 show that supervised learning based detector (e.g. Bayes-SAD) is vulnerable to obfuscated shilling attacks. Therefore, we compare the anti-AoP performance of Semi-SAD with PCA-SAD in this section.

**Table 4** Comparison of detection performance against *obfuscated average attacks*.

| Filler size | | 5% | 10% | 20% | 30% | 40% | 50% |
|---|---|---|---|---|---|---|---|
| *Sensitivity* | Semi-SAD | 0.989 | 0.989 | 1 | 1 | 0.989 | 1 |
| | Bayes-SAD | 0.564 | 0.702 | 0.734 | 0.691 | 0.713 | 0.776 |
| | PCA-SAD | 0.974 | 0.968 | 0.905 | 0.919 | 0.926 | 0.936 |
| *Specificity* | Semi-SAD | 0.858 | 0.921 | 0.948 | 0.939 | 0.941 | 0.937 |
| | Bayes-SAD | **0.87** | **0.932** | 0.927 | 0.927 | 0.909 | 0.916 |
| | PCA-SAD | 0.907 | 0.907 | 0.915 | 0.912 | 0.91 | 0.914 |

Note: The value larger than the one in "Semi-SAD" is in bold

**Table 5** Comparison of detection performance against *obfuscated segmented attacks*.

| Filler size | | 5% | 10% | 20% | 30% | 40% | 50% |
|---|---|---|---|---|---|---|---|
| *Sensitivity* | Semi-SAD | 1 | 1 | 1 | 1 | 1 | 1 |
| | Bayes-SAD | 0.766 | 0.904 | 0.872 | 0.883 | 0.894 | 0.883 |
| | PCA-SAD | 0.91 | 0.896 | 0.898 | 0.932 | 0.912 | 0.856 |
| *Specificity* | Semi-SAD | 0.952 | 0.97 | 0.989 | 0.976 | 0.999 | 1 |
| | Bayes-SAD | 0.927 | 0.963 | 0.982 | 0.948 | 0.971 | 0.958 |
| | PCA-SAD | 0.965 | 0.966 | 0.98 | 0.955 | 0.95 | 0.963 |

Note: The value larger than the one in "Semi-SAD" is in bold

**Table 6** Comparison of detection performance against *obfuscated bandwagon attacks*.

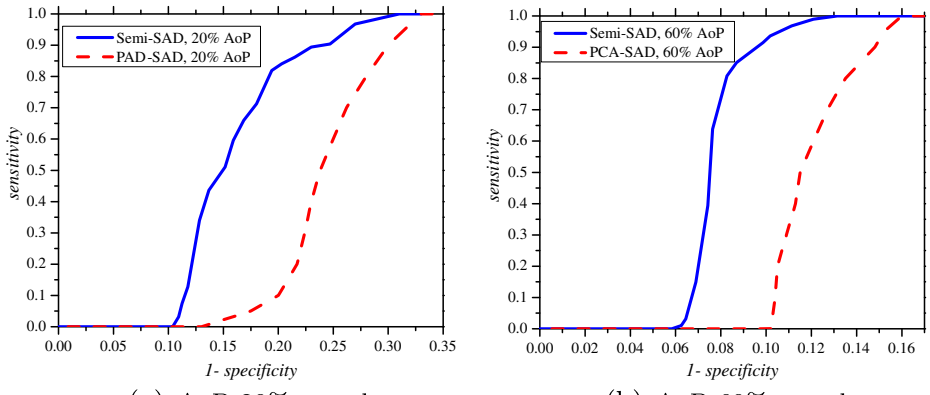
| Filler size | | 5% | 10% | 20% | 30% | 40% | 50% |
|---|---|---|---|---|---|---|---|
| *Sensitivity* | Semi-SAD | 0.947 | 0.957 | 1 | 1 | 1 | 1 |
| | Bayes-SAD | 0.904 | 0.83 | 0.787 | 0.84 | 0.819 | 0.798 |
| | PCA-SAD | 0.918 | 0.967 | 0.935 | 0.921 | 0.982 | 0.91 |
| *Specificity* | Semi-SAD | 0.915 | 0.972 | 0.999 | 0.996 | 1 | 1 |
| | Bayes-SAD | 0.934 | **0.969** | 0.967 | 0.958 | 0.952 | 0.961 |
| | PCA-SAD | 0.976 | 0.985 | 0.983 | 0.965 | 0.957 | 0.973 |

Note: The value larger than the one in "Semi-SAD" is in bold

**Table 7** Comparison of detection performance against *obfuscated sampling attacks*.

| Length of profiles | | [20,30) | [30,50) | [50,100) | [100,200) | ≥ 200 |
|---|---|---|---|---|---|---|
| *Sensitivity* | Semi-SAD | 0.914 | 0.904 | 0.935 | 0.946 | 0.979 |
| | Bayes-SAD | 0.872 | 0.883 | 0.851 | 0.862 | 0.84 |
| | PCA-SAD | 0.328 | 0.347 | 0.465 | 0.586 | 0.59 |
| *Specificity* | Semi-SAD | 0.823 | 0.879 | 0.834 | 0.865 | 0.893 |
| | Bayes-SAD | **0.897** | **0. 906** | **0.881** | **0.904** | **0.896** |
| | PCA-SAD | 0.58 | 0.576 | 0.839 | 0.826 | 0.875 |

Note: The value larger than the one in "Semi-SAD" is in bold




(a) AoP 20% attack (b) AoP 60% attack

**Figure 5** Semi-SAD vs. PCA-SAD against AoP attack.

**Table 8** Comparison of detection performance against *average + bandwagon attacks*.

| Filler size | | 5% | 10% | 20% | 30% | 40% | 50% |
|---|---|---|---|---|---|---|---|
| *Sensitivity* | Semi-SAD | 0.964 | 0.968 | 0.985 | 0.987 | 0.977 | 0.986 |
| | Bayes-SAD | 0.849 | 0.968 | 0.979 | 0.968 | 0.957 | 0.968 |
| | PCA-SAD | 0.96 | 0.958 | 0.954 | 0.962 | 0.952 | 0.958 |
| *Specificity* | Semi-SAD | 0.99 | 1 | 1 | 1 | 1 | 1 |
| | Bayes-SAD | 0.941 | 0.976 | 0.972 | 0.963 | 0.978 | 0.978 |
| | PCA-SAD | 0.94 | 0.966 | 0.968 | 0.976 | 0.97 | 0.97 |

ROC curve is employed to describe the performance of two detectors. ROC space is defined by 1- *specificity* and *sensitivity* as $x$ and $y$ axes respectively, which depicts relative trade-offs between true positive (benefits) and false positive (costs). For Semi-SAD, the ROC is generated by varying the threshold $\eta$. For PCA-SAD, the ROC is generated by varying the attack cluster size. Figure 5 illustrates the ROC curves of two detectors in AoP 20% attack and AoP 60% attack respectively. The results show that Semi-SAD has higher probability of good detection and lower probability of false alarm than PCA-SAD.

### 5.3 Performance of Semi-SAD against hybrid shilling attack

Hybrid shilling attack as mentioned in Section 2.3 injects various types of shilling attacker profiles into the recommender system simultaneously. The experiments in Section 5.3 evaluate the performance of Semi-SAD against hybrid shilling attacks.

Average attack and random attack are similar because they rate 5 to only one item; bandwagon attack and segmented attack are similar because they rate 5 to a group of items. Therefore, the combination of average attack and bandwagon attack is chosen as the representative of hybrid shilling attack. Firstly, we compare sensitivity and specificity of three algorithms against average + bandwagon attacks and obfuscated average + bandwagon attacks. The attack size is set to 10%, and the numbers of two kinds of attacks are equivalent. Tables 8 and 9 show the experimental results. Three algorithms all achieve high sensitivity and specificity. However, Semi-SAD has the highest sensitivity and specificity all the time.

Secondly, we evaluate Semi-SAD in more wretched conditions where five types of shilling attacks are injected simultaneously. The attack size is extended to 15%, and each type of attack accounts for 3%. Tables 10 and 11 illustrate the experimental results. Sensitivity of Bayes-SAD and PCA decreases dramatically comparing with Tables 8 and 9. However, sensitivity of Semi-SAD decreases slightly and still remains

**Table 9** Comparison of detection performance against *obfuscated average + bandwagon attacks*.

| Filler size | | 5% | 10% | 20% | 30% | 40% | 50% |
|---|---|---|---|---|---|---|---|
| *Sensitivity* | Semi-SAD | 0.922 | 0.936 | 0.93 | 0.904 | 0.918 | 0.906 |
| | Bayes-SAD | 0.916 | 0.883 | 0.894 | 0.883 | 0.904 | 0.894 |
| | PCA-SAD | 0.918 | 0.928 | 0.894 | 0.892 | 0.892 | 0.896 |
| *Specificity* | Semi-SAD | 0.98 | 1 | 1 | 1 | 1 | 1 |
| | Bayes-SAD | 0.898 | 0.904 | 0.913 | 0.886 | 0.897 | 0.894 |
| | PCA-SAD | 0.942 | 0.956 | 0.962 | 0.966 | 0.968 | 0.97 |

**Table 10** Comparison of detection performance against *hybrid attacks with five types*.

| Filler size | | 5% | 10% | 20% | 30% | 40% | 50% |
|---|---|---|---|---|---|---|---|
| *Sensitivity* | Semi-SAD | 0.818 | 0.792 | 0.828 | 0.864 | 0.883 | 0.908 |
| | Bayes-SAD | 0.521 | 0.613 | 0.671 | 0.628 | 0.649 | 0.66 |
| | PCA-SAD | 0.519 | 0.576 | 0.623 | 0.619 | 0.626 | 0.645 |
| *Specificity* | Semi-SAD | 0.965 | 0.975 | 0.963 | 0.98 | 0.985 | 0.97 |
| | Bayes-SAD | 0.954 | 0.968 | 0.953 | 0.971 | 0.963 | 0.968 |
| | PCA-SAD | 0.936 | 0.945 | 0.941 | 0.952 | 0.949 | 0.953 |

steady at a high level. The specificity of three algorithms does not change much, because normal user profiles are unaltered. The results in Tables 3–11 prove that Semi-SAD has a great advantage over supervised and unsupervised learning based algorithms especially in the case of hybrid shilling attacks.

### 5.4 Impact of λ

The experiments in Section 5.4 utilize EM-λ to reduce the weight of the unlabeled data by varying λ in (17) and (18). We range λ from 0 to 1 and the interval is set to 0.1.The size of labeled data is also considered in the experiments. Three scales of labeled data size are chosen: 50N-25S, 100N-50S, 150N-75S. AoP attack is tested, and the filler size is set to 5%. Shilling profiles in training dataset are AoP 30% attacks, and the *actual* attacks are AoP 60% attacks.

Figure 6 shows that sensitivity and specificity of Semi-SAD varies with the increase of λ in three scales of labeled data size. Three curves in Figure 6a exhibit similar variation tendency. The curves reach a peak when the λ value is intermediate. Meanwhile, the curve of small labeled data size changes more significantly than that of large labeled data size. However, as the λ value increases, the specificity of three different labeled data size varies irregularly. It arises because the normal user profiles in training dataset and test dataset are similar. However, specificity does not benefit from the weight adjustment of unlabeled data as in Figure 6b. Because the labeled shilling profiles are very useful to guide parameter estimation of EM, the smaller labeled data size is, the smaller weight unlabeled data should be given.

Experimental results shown in Figure 6 indicate that when the labeled data size is very small, carefully selecting λ could significantly improve the practical performance of shilling detection even further. Strictly speaking, we can use *cross-validation* approach to determine the optimal λ value for the specific data. However, in practice we are able to set λ between 0.4 and 0.6 to obtain satisfying results.

**Table 11** Comparison of detection performance against *obfuscated hybrid attacks with five types*.

| Filler size | | 5% | 10% | 20% | 30% | 40% | 50% |
|---|---|---|---|---|---|---|---|
| *Sensitivity* | Semi-SAD | 0.77 | 0.758 | 0.776 | 0.836 | 0.847 | 0.838 |
| | Bayes-SAD | 0.486 | 0.532 | 0.532 | 0.574 | 0.564 | 0.585 |
| | PCA-SAD | 0.468 | 0.549 | 0.567 | 0.576 | 0.57 | 0.615 |
| *Specificity* | Semi-SAD | 0.958 | 0.968 | 0.968 | 0.973 | 0.979 | 0.968 |
| | Bayes-SAD | 0.957 | 0.967 | 0.959 | 0.968 | 0.963 | 0.963 |
| | PCA-SAD | 0.94 | 0.948 | 0.946 | 0.945 | 0.938 | 0.942 |

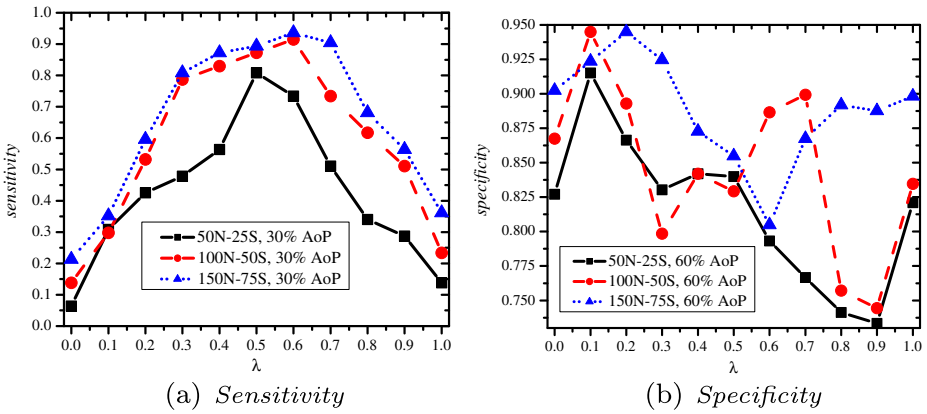



**Figure 6** Impact of λ on three different scales of labeled data.

## 6 Conclusion

It is observed that there are few labeled users and a great deal of unlabeled users in the real recommender systems. Existing research efforts on shilling attack detection are based on either labeled data, or unlabeled data. We present *Semi-SAD*, a novel semi-supervised learning based shilling attack detection algorithm.

This algorithm learns from both labeled and unlabeled user profiles based on the combination of naïve Bayes classifier and EM-λ. Since the labeled data is valuable and quite rare compared to the unlabeled data, EM-λ is employed to increase the weight of the labeled data. Experimental results demonstrate that it could significantly improve the performance of shilling detection if we carefully select λ, especially for the small labeled dataset.

Our experiments compare Semi-SAD with Bayes-SAD and PCA-SAD, the supervised and unsupervised learning based approaches, in the scenarios of obfuscated single shilling attack and hybrid shilling attacks. All three approaches are effective to various obfuscated single shilling attacks, but Semi-SAD performs better than others. Meanwhile, Semi-SAD has a great advantage over PCA-SAD against AoP attack. Besides, the conducted experiments on hybrid shilling attacks show that Semi-SAD performs significantly better than Bayes-SAD and PCA-SAD in the case of hybrid shilling attacks.

In the future, the techniques for rare class analysis [21] will be adopted to shilling attack detection, and an incremental Semi-SAD algorithm will be studied to deal with the ever-increasing number of users in the real recommender systems.

**Acknowledgements** This research is supported by National Natural Science Foundation of China (Nos. 71072172 and 61103229), Industry Projects in the Jiangsu Science & Technology Pillar Program (No. BE2011198), Jiangsu Provincial Colleges and Universities Outstanding Science & Technology Innovation Team Fund (No. 2001013), International Science & Technology Cooperation Program of China (No. 2011DFA12910), National Key Technologies R&D sub Program in 12th five-year-plan (No. SQ2011GX07E03990), and Jiangsu Provincial Key Laboratory of Network and Information Security (Southeast University) (No. BM2003201).

## References


1. Bell, R.M., Koren, Y.: Improved neighborhood-based collaborative filtering. In: Proceedings of the 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD'07), pp. 7–14 (2007)
2. 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'06), pp. 542–547 (2006)
3. Cacheda, F., Carneiro, V., Fernandez, D., Formoso, V.: Comparison of collaborative filtering algorithms: limitations of current techniques and proposals for scalable, high-performance recommender systems. ACM Trans. Web (TWEB'11) **5**(1), 3–34 (2011)
4. Castelli, V., Cover, T.M.: On the exponential value of labeled samples. Pattern Recogn. Lett. **16**(1), 105–111 (1995)
5. Chiang, M.F., Peng, W.C., Yu, P.S.: Exploring latent browsing graph for question answering recommendation. WWWJ (2012). doi:10.1007/s11280-011-0146-0
6. Chirita, P.A., 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'05), pp. 67–74 (2005)
7. Gunawardana, A., Meek, C.: A unified approach to building hybrid recommender systems. In: Proceedings of the Third ACM Conference on Recommender Systems (RecSys'09), pp. 117–124 (2009)
8. Hurley, N., Cheng, Z., Zhang, M.: Statistical attack detection. In: Proceedings of the Third ACM Conference on Recommender Systems (RecSys'09), pp. 149–156 (2009)
9. Lam, S.K., Riedl, J.: Shilling recommender systems for fun and profit. In: Proceedings of the 13th International Conference on World Wide Web (WWW'04), pp. 393–402 (2004)
10. Lee, J., Zhu, D.: Shilling attack detection—a new approach for a trustworthy recommender system. INFORMS J. Comput. **24**(1), 117–131 (2012)
11. Leung, C.W., Chan, S.C., Chung, F., Ngai, G.: A probabilistic rating inference framework for mining user preferences from reviews. WWWJ **14**(2), 187–215 (2011)
12. Manouselis, N., Costopoulou, C.: Analysis and classification of multi-criteria recommender systems. WWWJ **10**(4), 415–441 (2007)
13. Mehta, B., Hofmann, T., Fankhauser, P.: Lies and propaganda: detecting spam users in collaborative filtering. In: Proceedings of the 12th International Conference on Intelligent User Interfaces (IUI'07), pp. 14–21 (2007)
14. Melville, P., Mooney, R.J., Nagarajan, R.: Content-boosted collaborative filtering for improved recommendations. In: 8th National Conference on Artificial Intelligence (AAAI'02), pp. 187–192 (2002)
15. Mobasher, B., Burke, R., Williams, C., Bhaumik, R.: Analysis and detection of segment-focused attacks against collaborative recommendation. In: WebKDD Workshop, pp. 96–118 (2006)
16. Nigam, K., Mccallum, A., Thrun, S., Mitchill, T.: Text classification from labeled and unlabeled documents using em. Machine Learn. **39**(2), 103–134 (2000)
17. Shahshahani, B.M., Landgrebe, D.A.: The effect of unlabeled samples in reducing the small sample size problem and mitigating the hughes phenomenon. IEEE Trans. Geosci. Remote Sens. **32**(5), 1087–1095 (1994)
18. Tan, P.N., Steinbach, M., Kumar, V.: Introduction to Data Mining. Addison-Wesley (2005)
19. Williams, C.: Profile injection attack detection for securing collaborative recommender systems. Technical report, DePaul University (2006)
20. Wu, X., Kumar, V., Ross, J.Q., et al.: Top 10 algorithms in data mining. Knowl. Inf. Syst. **14**(1), 1–37 (2008)
21. Wu, J., Xiong, H., Chen, J.: COG: local decomposition for rare class analysis. Data Min. Knowl. Discovery **20**(2), 191–220 (2010)
22. Wu, D., Ke, Y., Yu, J.X., Yu, P.S., Chen, L.: Leadership discovery when data correlatively evolve. WWWJ **14**(1), 1–25 (2011)

23. 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'11), pp. 289–292. Chicago, IL, USA (2011)
24. Zheng, Z., Ma, H., Lyu, M.R., King, I.: WSRec: a collaborative filtering based web service recommender system. In: IEEE International Conference on Web Services (ICWS'09), pp. 437–444 (2009)
25. Zhou, Z., Li, M.: Tri-training: exploiting unlabeled data using three classifiers. IEEE Trans. Knowl. Data Eng. (TKDE'05) **17**(11), 1529–1541 (2005)