HySAD: A Semi-Supervised Hybrid Shilling Attack Detector for Trustworthy Product Recommendation

Zhiang Wu[1], Junjie Wu[2*], Jie Cao[1], and Dacheng Tao[3]
[1]Jiangsu Provincial Key Laboratory of E-Business, Nanjing University of Finance and Economics, China
[2*](Corresponding Author) School of Economics and Management, Beihang University, China
[3]Centre for Quantum Computation & Intelligent Systems, University of Technology, Sydney
zawu@seu.edu.cn, wujj@buaa.edu.cn, caojie690929@163.com,
Dacheng.Tao@uts.edu.au

ABSTRACT

Shilling attackers apply biased rating profiles to recommender systems for manipulating online product recommendations. Although many studies have been devoted to shilling attack detection, few of them can handle the *hybrid* shilling attacks that usually happen in practice, and the studies for real-life applications are rarely seen. Moreover, little attention has yet been paid to modeling both labeled and unlabeled user profiles, although there are often a few labeled but numerous unlabeled users available in practice. This paper presents a Hybrid Shilling Attack Detector, or HySAD for short, to tackle these problems. In particular, HySAD introduces MC-Relief to select effective detection metrics, and Semi-supervised Naïve Bayes (SNB_λ) to precisely separate Random-Filler model attackers and Average-Filler model attackers from normal users. Thorough experiments on MovieLens and Netflix datasets demonstrate the effectiveness of HySAD in detecting hybrid shilling attacks, and its robustness for various obfuscated strategies. A real-life case study on product reviews of Amazon.cn is also provided, which further demonstrates that HySAD can effectively improve the accuracy of a collaborative-filtering based recommender system, and provide interesting opportunities for in-depth analysis of attacker behaviors. These, in turn, justify the value of HySAD for real-world applications.

Categories and Subject Descriptors

H.2.8 [**Database Management**]: Database Applications—*Data Mining*

General Terms

Algorithms, Experimentation

Keywords

Hybrid Shilling Attack, Semi-supervised Learning, Naïve Bayes, Recommendation Systems

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
KDD'12, August 12–16, 2012, Beijing, China.
Copyright 2012 ACM 978-1-4503-1462-6 /12/08 ...$15.00.

1. INTRODUCTION

Recent years have witnessed the rapid growth of the Internet. More and more people now prefer online shopping over conventional shopping. The all-round infiltration of Web 2.0 these days further promotes this process. People enjoy new online shopping experiences by publishing, browsing, or sharing product reviews written by themselves or others. Indeed, research shows that people rely on the opinions of others when selecting something for the first time [1]. This selection behavior, together with the increasingly apparent problem of information overload, has motivated the development of recommender systems for automatic and personalized recommendation of books, movies, electronics, etc.

However, there is always a natural profit incentive to promote one's own products by leaving biased online comments and/or extreme ratings for the competition's products, which has spawned the so-called "shilling attacks". It appears that shilling attacks are successfully threatening recommender systems, and continuously generating huge volumes of spam or misleading review comments. Shilling attacks are therefore emerging as a considerable challenge to the sustainable development of online shopping.

In the literature, a great deal of research has been undertaken to detect shilling attacks from large-scale rating profiles. These studies mainly focus on the three subareas: the shilling attack generation models [21, 15], the shilling attack detection metrics [3, 6, 21, 22], and the shilling attack classification methods [6, 21, 13, 8, 10], which have greatly advanced the research onto a much higher level.

However, existing studies have some limitations. First, hybrid attacks consisting of multiple attack models are still a challenging problem in both academia and industry. Moreover, most existing studies used either supervised or unsupervised learning methods for attack detection, which do not agree with the fact that, in practice, we are often given only a few labeled but a majority of unlabeled user profiles. Finally, most existing studies are only validated by simulation results on artificially injected attacker profiles, rather than real-life applications. These drawbacks severely limit the use of the shilling attack detection methods in practice.

In light of this, we present Hybrid Shilling Attack Detector, or HySAD for short, to solve these problems. In general, HySAD has two distinct properties. First, HySAD is a hybrid-attack oriented system that collects many popular shilling attack detection metrics for the purpose of feature selection via a wrapper called MC-Relief. These metrics then become the informative features of the training data.

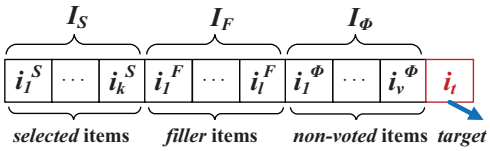


Figure 1: The illustration of a shilling profile.

Second, HySAD is a semi-supervised learning system that employs the Semi-supervised Naïve Bayes (SNB_λ) classifier for the categorization of both labeled and unlabeled profiles.

Extensive experiments on both the MovieLens and Netflix datasets demonstrate that, compared with some state-of-the-art methods proposed in the literature, HySAD is particularly effective for hybrid shilling attacks, even when attack variants generated by some obfuscated strategies are presented. A real-life application on product recommendation for Amazon.cn is also provided to demonstrate the effectiveness of HySAD. The results indicate that HySAD indeed improves the accuracy of a collaborative-filtering based recommender system, identifies some deeply hidden shilling attackers on Amazon.cn, and provides interesting opportunities for further study of attacker behaviors from various perspctives.

2. RELATED WORK

The word "shilling" was first termed in 2004 [9, 17]. Since then, several attack profile generation models have been proposed in the literature [21, 15], and a lot of research has been undertaken to employ supervised learning for shilling attack detection [3, 14]. For instance, a simple detector using the average similarity and the Rating Deviation from Mean Agreement (RDMA) metric was presented in [6]. A decision-tree based detector was also proposed in [21]. Many detection metrics were also presented to characterize the attacker profiles [6, 21, 3, 22]. An early unsupervised shilling attack detector is PCASelectUsers [13], which employs the principal component analysis on the original rating data. Hurley et al. presented the Neyman-Pearson statistical detector, with both supervised and unsupervised versions [8]. Recently, Lee et al. proposed a new detector using a clustering method and the Group RDMA (GRDMA) metric [10]. Other specific detectors include the segment-attack detector [4], the group-attack detector [19], and the time-series based detector [22]. Given the limited space, it is difficult to include all the related work in, and we hope the cited review [18] about robust collaborative recommendation can point to some of those missing references.

3. PRELIMINARIES AND THE SYSTEM OVERVIEW

Shilling attackers often have three intents: *push*, *nuke* and *vandalism* [21]. An attacker may insert ratings to make an item more likely (push) or less likely (nuke) to be recommended, or just make the entire recommender system malfunction (vandalism). As we only consider the intents that bring economic benefits to attackers, vandalism will not be considered in this study.

The rating records of a user on various products (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 shown in Fig. 1. A target item often has the highest rating in a push attack, or the lowest rating in a nuke attack. Filler items can make a shilling profile look normal, yet have a profound impact on a recommender system. Selected items are often used to make friends with as many genuine users as possible. Finally, non-voted items form the remaining unrated items.

Table 1: Four shilling attack models (intent: push).

Type	Model	Ratings
RFM	random attack	I_S=∅; Give random ratings to items in I_F; $r(i_t)$=$r_{\max}$.
	bandwagon attack (random)	I_S contains frequently-rated items, and $r(i^S)$=$r_{\max}$; Give random ratings to items in I_F; $r(i_t)$=$r_{\max}$.
AFM	average attack	I_S=∅; The ratings to items in I_F are distributed around the mean of each item i^F; $r(i_t)$=$r_{\max}$.
	bandwagon attack (average)	I_S contains frequently-rated items, and $r(i^S)$=$r_{\max}$; The ratings to items in I_F are distributed around the mean of each item i^F; $r(i_t)$=$r_{\max}$.

Note: $r_{\max}$ ($r_{\min}$) - the highest (lowest) rating to an item.

Table 1 includes four generative models, i.e. random, average, bandwagon (random), and bandwagon (average), which are popular for generating shilling profiles by carefully rating these four types of items. Note that this is the case for the push intent, and the nuke-intent case can therefore be easily figured out. We can further categorize the four models into two types, i.e. Random-Filler models (RFM) and Average-Filler models (AFM). As a result, we have three types of user profiles in total: Normal users (N), RFM attackers (RF), and AFM attackers (AF).

Two obfuscated attack techniques, i.e. noise-injection and target-shifting, are often used for shilling attackers to avoid detection [21]. Recently, a simple, but effective strategy named Average-over-Popular (AoP) was also proposed to obfuscate the average attack [8]. The three techniques are listed below:

Noise-injection: Add a Gaussian distributed random number multiplied by α to each rating within a subset of filler items. This noise can be used to blur the profile characteristics that are associated with the above four attack types.

Target-shifting: For the push intent, this 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. It can also be used with all the four types of attacks.

AoP: This technique chooses filler items with equal probability from the top x% of the most popular items, instead of the entire items. It is often used with the average attack.

3.1 Problem Definition

The combined use of the four attack models and the three obfuscated techniques leads to the *hybrid* shilling attacks, which inject various types of shilling profiles into a recommender system simultaneously. From a practical view, a hybrid shilling attack is very likely to be adopted by malicious users to make their attacks much more difficult to detect. Moreover, different groups of attackers may use different attack models for the same online system, which will, practically, result in hybrid attacks. However, most existing algorithms concentrate on detecting shilling profiles generated by a single model, and have not been evaluated by hybrid attacks. To narrow the gap, the first concern is:

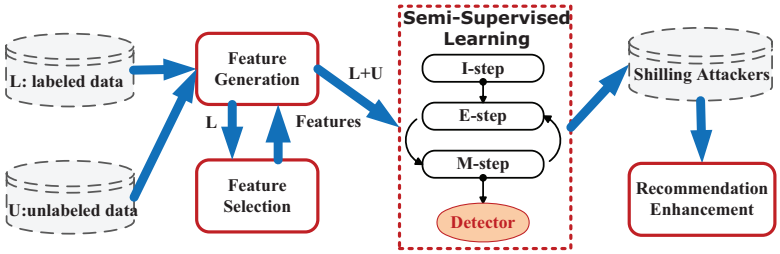


Figure 2: The procedure of HySAD.

T1: Design a detector to identify hybrid shilling attacks.

Another challenge in shilling attack detection is the lack of labeled users. It is known that, with the guidance of the label information, supervised learning generally has higher accuracy than unsupervised learning in data categorization [5]. However, given the huge volume of users in an online shopping system, such as Amazon.com, to manually label even ten percent of users is prohibitively expensive. Therefore, the real situation is, we are to be given a vast number of unlabeled users, together with a very small seed-set of labeled users, for the shilling attack detection. This directly leads to the second concern of this study:

T2: Design a detector that can use both labeled and unlabeled data.

T1 and T2 clearly define the problem we study in this paper. Specifically, a system named HySAD is proposed for hybrid shilling attack detection in the following subsection.

3.2 System Overview

Fig. 2 illustrates the diagram of the HySAD system. We provide a brief explanation as follows.

Firstly, we are given the user profile data, including a majority of unlabeled profiles U, and a small seed-set of labeled profiles L. To capture the essential information from different attacks, we collect as many as possible widely-adopted evaluation metrics in the literature, and then generate new features for all the users, as indicated by the "Feature Generation" box. These features will then be subjected to a feature-selection phase by using a wrapper method named MC-Relief on the labeled users, as indicated by the "Feature Selection" box. The training data is now ready for user profile classification.

A semi-supervised learning algorithm called Semi-supervised Naïve Bayes (SNB_λ) will then be launched for classification. That is, the labeled data will be firstly used to train a Naïve Bayes classifier, and predict the posterior probabilities of the unlabeled data; then the initial classifier will be improved by using an expectation-maximization-like iterative process, until certain stopping criterion is met. We finally obtain details of all the shilling attackers with higher probabilities for attacker classes. This valuable information will be further utilized to enhance the performance of a recommender system, as indicated by the "Recommendation Enhancement" box.

Discussion. Altogether we used ten metrics, i.e. Entropy, DegSim, LengthVar, RDMA, WDMA, WDA, MeanVar, FMTD, GFMV, and TMF, collected from the literature [3, 6, 21, 22] for feature generation. As a metric is often designed purposefully for one attack model, some may not be very suitable for hybrid attack detection, which is why a feature selection phase is required for HySAD. The reason for employing a semi-supervised learning in the classification modeling phase is also straightforward; that is, it will not only make use of the valuable label information, but also capture the proximity information between labeled and unlabeled data, and thus is more suitable for the shilling attack detection. The use of MC-Relief and SNB_λ is indeed optional, although these two algorithms do work surprisingly well in our study. Moreover, both models are very simple and explainable, which are the important merits for real-world applications. Some other classifiers, such as SVMs, may not have good interpretability, and thus are not considered for HySAD.

MC-Relief
Input:
L: the set of labeled user profiles
S: the sampling times
m: the number of total features
d: the number of features to select
c: the number of classes
Output:
F: the set of selected features
Variable:
ω: the set of feature weights
Procedure:
1. for l=1:m
2. $\omega_l = 0$;
3. end for
4. for j=1:S
5. u=RandomSelect(L);
6. u_p=SNN(u);
7. $(u_n^{(1)}, \cdots, u_n^{(c-1)})$=DNN($u$);
8. for l=1:m
9. Update$(\omega_l, u_p, u_n^{(1)}, \cdots, u_n^{(c-1)})$;
10. end for
11. end for
12. $F = \{f_l | \omega_l$ is among the top-d weights$\}$;

Figure 3: The MC-Relief algorithm.

4. FEATURE SELECTION

In this section, we present a simple heuristic MC-Relief to select the right metrics for the subsequent modeling task. MC-Relief is an extension from Relief [11], a feature selection algorithm for binary-class data. As MC-Relief is a wrapper method using the labeled data, we should first determine the number of classes we should use.

In the literature, user profiles are often categorized into two classes, i.e. normal users and attackers. However, from a hybrid attack perspective, setting only one attacker class may not be adequate. Indeed, the attacker using an average attack and the attacker using a random attack may have quite different feature values on metrics, such as WDMA

and MeanVar, that use average ratings of items. The former attacker's ratings to filler items have a strong correlation with the average ratings, whereas the latter attacker's ratings tend to have a much weaker correlation with the average ratings. This implies that we should further divide the attackers into two subclass according to the attack models outlined in Section 3. This results in three types of users in total: normal users (N), AFM attackers (AF), and RFM attackers (RF). We will also demonstrate the benefits of setting three, rather than two classes in experiments.

We then describe the feature selection heuristic. MC-Relief is actually an extension of the Relief algorithm for multi-class problems. In general, MC-Relief aims to estimate the quality of features to distinguish the user profiles that are near to each other. Fig. 3 shows the pseudocodes, where ω_l denotes the weight of feature l, $l = 1, \cdots, m$, and the d features with top-d weights are finally selected. The weights will be updated S times during the selection process (Line 4). Each time MC-Relief randomly selects a profile u from L (Line 5), searches L for u's c nearest neighbors in c different classes (Lines 6 and 7), and then updates the weights sequentially. Note that the SNN function (Line 6) finds the nearest neighbor u_p in the same class of u, whereas the DNN function (Line 7) finds the nearest neighbors $\{u_n^{(1)}, \cdots, u_n^{c-1}\}$ in other classes. The proximity function used in SNN and DNN is the Pearson correlation coefficient. The Update function (Line 9) calls Eq. 1:

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

to update the weights, where $x_{u,l}$ denotes the l-th feature value of u. To avoid the scale bias, we should normalize the feature values to [0,1] before calling MC-Relief.

5. CLASSIFICATION

Feature generation and selection provide training data that contains a very small portion of labeled data. We then propose a semi-supervised learning scheme named SNB$_\lambda$ for classification. In general, SNB$_\lambda$ is an EM-like algorithm that extends the Naïve Bayes classifier [20] to the situation where both labeled and unlabeled data are present.

Suppose we are given a dataset $\mathcal{D} = \{\mathbf{x}_i\}_{i=1}^n$ of n instances drawn independently from a mixture density of c components:

$$p(\mathbf{x}_i|\boldsymbol{\theta}) = \sum_{j=1}^{c} p(\mathbf{x}_i|C_j, \boldsymbol{\theta}_j) P(C_j), \quad (2)$$

where $\boldsymbol{\theta} = (\boldsymbol{\theta}_1, \cdots, \boldsymbol{\theta}_c)$ contains unknown parameter $\boldsymbol{\theta}_j$ that uniquely determines $p(\mathbf{x}_i|C_j)$, $P(C_j)$ is the prior probability of class C_j, $j = 1, \cdots, c$. Assume the class-conditional probability of each dimension of $\mathbf{x}_i$ is independent of other dimensions and satisfies a normal distribution, i.e., $p(x_{il}|C_j, \boldsymbol{\theta}_j) = \frac{1}{\sqrt{2\pi}\sigma_{jl}} \exp(\frac{(x_{il}-\mu_{jl})^2}{2\sigma_{jl}^2})$, $1 \le l \le d$.

As $\mathcal{D}$ consists of a few labeled but majority unlabeled instances, we let $\mathcal{D} = \mathcal{D}^U \bigcup \mathcal{D}^{L_1} \bigcup \cdots \bigcup \mathcal{D}^{L_c}$, where $\mathcal{D}^U$ contains unlabeled instances, and $\mathcal{D}^{L_j}$ contains instances from class C_j. A weight $\lambda \in [0, 1]$ is also introduced to lower the impact of unlabeled instances on the classifier modeling. Given the above settings, we launch the maximum-likelihood estimation to find the value $\hat{\boldsymbol{\theta}}$ that maximizes $p(\mathcal{D}|\boldsymbol{\theta})$ (A similar method named EM$_\lambda$ can be found in [16], but it uses a Bayesian estimation, and is designed purposefully for text classification on discrete attributes.).

Given instance independency and the weight λ, we have

$$p(\mathcal{D}|\boldsymbol{\theta}) = (\Pi_{\mathbf{x}_i \in \mathcal{D}^U} p(\mathbf{x}_i|\boldsymbol{\theta}))^\lambda \Pi_{j=1}^c (\Pi_{\mathbf{x}_i \in \mathcal{D}^{L_j}} p(\mathbf{x}_i|\boldsymbol{\theta})). \quad (3)$$

Taking the natural logarithm, we obtain the objective function as follows:

$$\max_{\boldsymbol{\theta}} \quad \sum_{\mathbf{x}_i \in \mathcal{D}} \Lambda_i \ln p(\mathbf{x}_i|\boldsymbol{\theta}) \quad (4)$$
$$s.t. \quad \sum_{j=1}^{c} P(C_j) = 1,$$

where

$$\Lambda_i = \begin{cases} \lambda, & \text{if } \mathbf{x}_i \in \mathcal{D}^U, \\ 1, & \text{otherwise.} \end{cases} \quad (5)$$

Let $\mathcal{L} = \ln p(\mathcal{D}|\boldsymbol{\theta}) - \rho(\sum_{j=1}^c P(C_j) - 1)$. Since $p(\mathbf{x}_i|\boldsymbol{\theta}) = \sum_{j=1}^c p(\mathbf{x}_i|C_j, \boldsymbol{\theta}_j) P(C_j)$, we have

$$\nabla_{\boldsymbol{\theta}_j} \mathcal{L} = \sum_{\mathbf{x}_i \in \mathcal{D}} \Lambda_i \frac{1}{p(\mathbf{x}_i|\boldsymbol{\theta})} \nabla_{\boldsymbol{\theta}_j} p(\mathbf{x}_i|C_j, \boldsymbol{\theta}_j) P(C_j). \quad (6)$$

Note that

$$P(C_j|\mathbf{x}_i, \boldsymbol{\theta}) = \frac{p(\mathbf{x}_i|C_j, \boldsymbol{\theta}_j) P(C_j)}{p(\mathbf{x}_i|\boldsymbol{\theta})}. \quad (7)$$

If we substitute $p(\mathbf{x}_i|\boldsymbol{\theta})$ in Eq. 6 by the one in Eq. 7, we have

$$\nabla_{\boldsymbol{\theta}_j} \mathcal{L} = \sum_{\mathbf{x}_i \in \mathcal{D}} \Lambda_i P(C_j|\mathbf{x}_i, \boldsymbol{\theta}) \nabla_{\boldsymbol{\theta}_j} \ln p(\mathbf{x}_i|C_j, \boldsymbol{\theta}_j). \quad (8)$$

Let $\nabla_{\boldsymbol{\theta}_j} \mathcal{L} = 0$. Since

$$p(\mathbf{x}_i|C_j, \boldsymbol{\theta}_j) = \Pi_{l=1}^d p(x_{il}|C_j, \boldsymbol{\theta}_j)$$
$$= \Pi_{l=1}^d \frac{1}{\sqrt{2\pi}\sigma_{jl}} \exp(\frac{(x_{il}-\mu_{jl})^2}{2\sigma_{jl}^2}), \quad (9)$$

we can easily have $\forall\ 1 \le j \le c$, $1 \le l \le d$,

$$\hat{\mu}_{jl} = \frac{\lambda \sum_{\mathbf{x}_i \in \mathcal{D}^U} P(C_j|\mathbf{x}_i, \hat{\boldsymbol{\theta}}) x_{il} + \sum_{\mathbf{x}_i \in \mathcal{D}^{L_j}} x_{il}}{\lambda \sum_{\mathbf{x}_i \in \mathcal{D}^U} P(C_j|\mathbf{x}_i, \hat{\boldsymbol{\theta}}) + |\mathcal{D}^{L_j}|}, \quad (10)$$

$$\hat{\sigma}_{jl}^2 = \frac{\lambda \sum_{\mathbf{x}_i \in \mathcal{D}^U} P(C_j|\mathbf{x}_i, \hat{\boldsymbol{\theta}})(x_{il} - \hat{\mu}_{jl})^2 + \sum_{\mathbf{x}_i \in \mathcal{D}^{L_j}} (x_{il} - \hat{\mu}_{jl})^2}{\lambda \sum_{\mathbf{x}_i \in \mathcal{D}^U} P(C_j|\mathbf{x}_i, \hat{\boldsymbol{\theta}}) + |\mathcal{D}^{L_j}|}. \quad (11)$$

Furthermore, let $\frac{\partial \mathcal{L}}{\partial P(C_j)} = 0$, we have $\forall\ 1 \le j \le c$

$$\hat{P}(C_j) = \frac{\lambda \sum_{\mathbf{x}_i \in \mathcal{D}^U} P(C_j|\mathbf{x}_i, \hat{\boldsymbol{\theta}}) + |\mathcal{D}^{L_j}|}{\lambda |\mathcal{D}^U| + \sum_{j=1}^c |\mathcal{D}^{L_j}|}. \quad (12)$$

Combining the above results produces expectation-maximization iterative process as follows:

- **I**nitial step: $\forall\ j, l$, compute $\hat{\mu}_{jl}$, $\hat{\sigma}_{jl}^2$ and $\hat{P}(C_j)$ using Eqs. 10, 11 and 12, respectively, on labeled data. Then for $\mathbf{x}_i \in \mathcal{D}^U$, compute $p(\mathbf{x}_i|C_j, \hat{\boldsymbol{\theta}}_j)$ using Eq. 9. (This step is equivalent to building the Naïve Bayes classifier and using it to classify unlabeled data.)
- **E**-step: $\forall\ j$, compute $P(C_j|\mathbf{x}_i, \hat{\boldsymbol{\theta}})$ for all $\mathbf{x}_i \in \mathcal{D}^U$ using Eq. 7. For $\mathbf{x}_i \in \mathcal{D}^{L_j}$, let $P(C_j|\mathbf{x}_i, \hat{\boldsymbol{\theta}}) = 1$; and for $\mathbf{x}_i \in \mathcal{D}^{L_k}$, $k \ne j$, let $P(C_j|\mathbf{x}_i, \hat{\boldsymbol{\theta}}) = 0$.

Table 2: Experimental data sets.

Name	#User	#Movie	#Rating	Density
MovieLens_u2.test	943	1682	20000	1.26%
Netflix_s1	2000	2000	234467	5.86%

Table 3: The five selected features.

Data set	No.1	No.2	No.3	No.4	No.5
MovieLens	DegSim	TMF	Entropy	MeanVar	GFMV
Netflix	TMF	DegSim	Entropy	LengthVar	MeanVar

- **M**-step: $\forall\, j, l$, compute $\hat{\mu}_{jl}$, $\hat{\sigma}^2_{jl}$ and $\hat{P}(C_j)$ using Eqs. 10, 11 and 12, respectively. Then for $\mathbf{x}_i \in \mathcal{D}^U$, compute $p(\mathbf{x}_i|C_j, \hat{\boldsymbol{\theta}}_j)$ using Eq. 9.

The iterative process will be suspended if there is a very small change in the parameters of the objective-function in Eq. 4, so we have $P(C_j|\mathbf{x}_i, \hat{\boldsymbol{\theta}})$ for all $\mathbf{x}_i \in \mathcal{D}^U$ and all classes. Returning to the shilling-attack detection task, we have only three classes: N, AF, and RF. We let

$$\Omega_{AF}(\mathbf{x}_i) = \frac{P(AF|\mathbf{x}_i, \hat{\boldsymbol{\theta}})}{P(N|\mathbf{x}_i, \hat{\boldsymbol{\theta}})}, \Omega_{RF}(\mathbf{x}_i) = \frac{P(RF|\mathbf{x}_i, \hat{\boldsymbol{\theta}})}{P(N|\mathbf{x}_i, \hat{\boldsymbol{\theta}})}, \quad (13)$$

and label $\forall\ \mathbf{x}_i \in \mathcal{D}^U$ as an attacker if $\Omega_{AF}(\mathbf{x}_i) > \eta$ or $\Omega_{RF}(\mathbf{x}_i) > \eta$, where η is an empirical threshold.

6. EXPERIMENTAL VALIDATION

In this section, we present experimental results on two well-known real-world datasets: MovieLens and Netflix. We demonstrate that three factors, i.e. the semi-supervised learning, the feature selection, and the three-class setting, are crucial for obtaining excellent performance from HySAD.

6.1 Experimental Setup

Datasets. Table 2 lists the information of the two datasets. MovieLens is a benchmark movie-evaluation dataset widely used in previous studies [9, 21, 13, 8, 6], and u2.test is one of its subsets. Netflix is also a well-known movie-evaluation dataset [2], from which a subset s1 was obtained for our study. In detail, we first extracted 2000 movies whose sizes of rating files ranged from 170K to 1.44M; then we randomly extracted a sample of 2000 users who had rated no less than 50 of the selected movies. Note that these two datasets have quite different densities, and only the sub-sets were used since one of the comparative detectors, i.e. PCASelectUsers (PCA for short), which will be introduced below, does not work for large-scale data.

Tools. Four shilling attack detectors, i.e. HySAD, NB, C4.5, and PCA, were used in the experiments for the purpose of comparison. HySAD employs a semi-supervised learning scheme, NB (Naïve Bayes classifier) and C4.5 [20] employ supervised learning schemes, and PCA (PCASelectUsers) [13] employs an unsupervised learning scheme. Both HySAD and NB were coded in C++ and used on the data with generated features, with $\eta = 1$. However, PCA was coded in MATLAB to facilitate the principal-component computation, and used on the original rating data. C4.5 was the J48 version provided by WEKA with the default settings on the generated data. We selected C4.5 and PCA in this instance because they had been used for shilling attack detection and achieved good performance on MovieLens [21, 13].

Settings. We assumed the user profiles in the datasets were normal, and injected attacker profiles into them according to the four attack models and three obfuscated methods

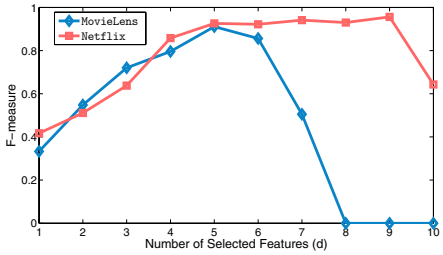


Figure 4: Impact of the number of selected features.

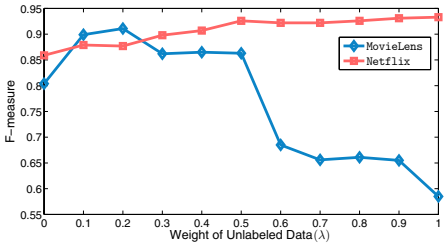


Figure 5: Impact of the weight of unlabeled data.

mentioned in Section 3. In detail, for MovieLens data, we sampled about 10% users (94 users) as labeled normal users, injected about 5% attackers (48 users) as labeled attackers, and thus formed the labeled data L with $|L| = 142$. To test the performance of detectors, 100 attackers were also injected into the remaining 849 users, and formed the unlabeled data U with $|U| = 949$. The Netflix was processed in a similar way, with 100 attackers and 200 normal users in L, and 200 attackers and 1800 normal users in U. For the hybrid attacks, attackers were divided evenly for different models, and the AFM and RFM attackers were labeled as AF and RF, respectively. Finally, note that the obfuscated methods were used for all the attackers in the unlabeled data, to verify whether detectors can adapt to attack variants.

Measures. We adopted the widely-used recall (R), precision (P), and F-measure (F) for the performance evaluation [20]. Specifically, we focused on the ability of detectors to recognize the hybrid attackers, where

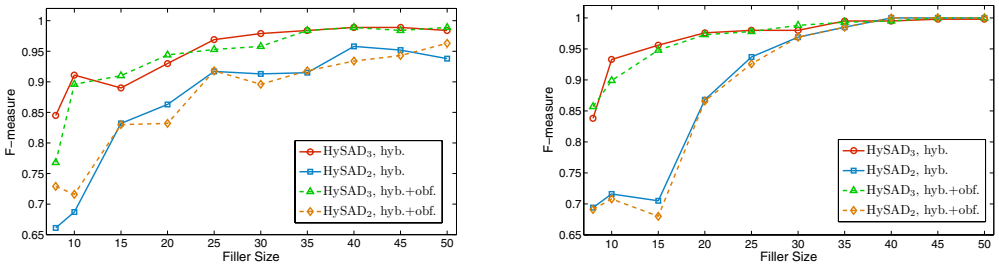
$$P = \frac{TP}{TP + FP}, R = \frac{TP}{TP + FN}, F = \frac{2PR}{P + R}, \quad (14)$$

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.

6.2 Parameter Analysis for HySAD

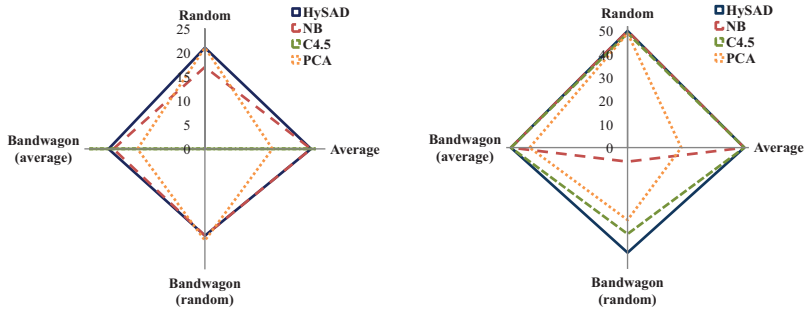
Here, we investigate the impact of the two parameters, i.e. the number of selected features (d) and the weight for unlabeled data (λ), on the performance of HySAD. We set the filler size (the size of filler-item set) to 10% of the total item size, i.e. $|I^F| = 10\%|I|$. The injected hybrid attack, denoted as "hyb.", was the mixture of four types of attacks: random, average, bandwagon(random), and bandwagon(average). No obfuscated methods were used. Ten-fold cross-validation was employed to provide robust results.

Fig. 4 shows the performance of HySAD with different numbers of selected features, when the weight $\lambda = 0.5$. As can be seen, when $d = 5$, the maximum F-measure value is



(a) MovieLens (b) Netflix

Figure 6: The effect of three-class modeling of HySAD.



(a) MovieLens (b) Netflix

Figure 7: The performance of detectors against attacks of different types.

obtained for the MovieLens dataset. Moreover, although the Netflix dataset reaches the the maximum F-measure value when $d = 9$, $d = 5$ still provides a fairly good performance. As a result, we set $d = 5$ for HySAD in all the experiments. Table 3 also lists the selected five features on MovieLens and Netflix, respectively. As can be seen, the two sets of features are very similar to each other, except for GFMV and LengthVar. This implies that the feature selection algorithm MC-Relief is fairly robust. Fig. 5 shows the performance of HySAD given different weights of unlabeled data. Similarly to d, we find that $\lambda = 0.5$ is a relatively robust choice, becoming the default setting in our experiments.

6.3 The Necessity of MultiClass Modeling

In Section 4, we mentioned that we would like to label user profiles by one normal class (N) and two attacker classes (AF and RF). Here we demonstrate why taking only one attacker class is not suitable for HySAD. The hybrid attack "hyb.", defined in Section 6.2, was again injected into data. We also used two obfuscated methods, i.e. noise-injection ($\alpha = 0.2$) and target shifting, on all the attacker profiles in unlabeled data after the injection of "hyb.". We denote this more complicated attack as "hyb.+obf.". The filler size was changed from 8 to 50, to test the consistency of results.

Fig. 6 shows the results after employing HySAD on three-class data (denoted as HySAD_3) and binary-class data (denoted asHySAD_2), respectively. As can be seen, the impacts of the "hyb." and "hyb.+obf." attacks on HySAD are not significantly different. This implies that HySAD is robust for some variants of attacks. More importantly, HySAD_3 shows consistently higher F-measure values than HySAD_2, although the gap tends to be narrowed with the increase of the filler size. This indicates that it is better to divide attackers into AFM attackers and RFM attackers for the modeling of HySAD.

To explain the observation, let us look into the recall (R) and precision (P) values of HySAD_3 and HySAD_2. As can be seen from Table 4, HySAD_2 may have slightly higher recall values, e.g. on Netflix, but its precision values are typically much smaller than the values of HySAD_3. This implies that employing HySAD on binary data may help to find more attackers, but at the cost of wrongly classifying much more normal users as attackers, which largely degrades the overall performance. This clearly demonstrates that the random attack and the average attack indeed behave very differently, as discussed in Section 4.

6.4 The Comparison of Different Detectors

This section compares the performance of the four detectors: HySAD, NB, C4.5, and PCA, to demonstrate the effectiveness of semi-supervised learning for hybrid attack detection. Two hybrid attacks, i.e. "hyb." and "hyb.+obf.", were again injected. The filler size was changed from 8 to 30. Note that NB and C4.5 only uses the labeled data L for modeling, whereas PCA works on the original rating data. We assumed that PCA knew the number of injected attackers k, and returned the top-k users as attackers.

Table 4 lists the evaluation results of the four detectors, where the value larger than the one of HySAD is in italics. As can be seen, according to the F-measure, HySAD_3 performs the best among the four detectors in most of the cases, especially when the filler size is relatively small. As the filler size increases, both HySAD_3 and NB provide better prediction power, and the gap narrows. In contrast, PCA generally has the poorest performance for the two hybrid attacks, and the situation becomes even worse as the filler size increases. This implies that PCA may not be very suitable for hybrid attack detection. C4.5 is a bit unusual; that is, it is generally poorer than NB and better than PCA, but shows fluctuant results as the filler size increases.

We take a closer look at the recall and precision values of NB. As can be seen, NB often has a slightly higher preci-

Table 4: The performance of detectors against hybrid attacks.

Attack	Measure	Detector	Filler size(%) MovieLens 8	15	20	25	Netflix 8	15	20	25
Hyb.	*R*	$HySAD_3$	0.957	0.862	0.989	0.989	0.765	0.995	1	1
		$HySAD_2$	0.819	*0.872*	0.936	*1*	*1*	*1*	1	1
		NB	0.872	0.809	0.862	0.936	0.765	0.780	0.855	1
		C4.5	0.500	0.500	0.500	0.750	*0.820*	0.930	0.675	1
		PCA	0.798	0.723	0.564	0.170	0.755	0.725	0.725	0.700
	P	$HySAD_3$	0.756	0.920	0.877	0.949	0.927	0.917	0.952	0.962
		$HySAD_2$	0.554	0.796	0.800	0.847	0.532	0.545	0.766	0.881
		NB	0.683	0.894	*0.910*	0.946	*0.968*	*0.945*	*0.967*	*0.995*
		C4.5	*1*	0.691	*0.979*	0.949	0.854	*0.949*	*0.988*	*0.971*
		PCA	*0.798*	0.723	0.564	0.170	0.755	0.725	0.725	0.700
	F	$HySAD_3$	0.845	0.890	0.930	0.969	0.838	0.954	0.976	0.980
		$HySAD_2$	0.661	0.832	0.863	0.917	0.694	0.716	0.705	0.868
		NB	0.766	0.849	0.885	0.941	*0.855*	0.855	0.924	*0.998*
		C4.5	0.667	0.580	0.662	0.867	0.837	0.939	0.802	*0.985*
		PCA	0.798	0.723	0.564	0.170	0.755	0.725	0.725	0.700
Hyb.+Obf.	*R*	$HySAD_3$	0.670	0.968	0.979	0.979	0.795	1	1	1
		$HySAD_2$	*0.957*	0.777	0.894	0.957	*1*	1	1	1
		NB	*0.872*	0.734	0.862	0.915	0.760	0.750	0.805	1
		C4.5	0.479	0.500	0.500	0.766	*0.810*	0.880	0.675	1
		PCA	*0.777*	0.723	0.564	0.255	0.785	0.755	0.725	0.700
	P	$HySAD_3$	0.900	0.858	0.911	0.929	0.930	0.901	0.948	0.957
		$HySAD_2$	0.588	*0.890*	0.778	0.882	0.528	0.515	0.763	0.862
		NB	0.732	*0.908*	0.910	*0.945*	*0.968*	*0.949*	*0.970*	*0.990*
		C4.5	*1*	0.373	*0.916*	*0.935*	0.806	*0.941*	*0.988*	*0.971*
		PCA	0.777	0.723	0.564	0.255	0.785	0.755	0.720	0.695
	F	$HySAD_3$	0.768	0.910	0.944	0.953	0.857	0.948	0.973	0.978
		$HySAD_2$	0.729	0.830	0.832	0.918	0.691	0.680	0.866	0.926
		NB	*0.796*	0.812	0.885	0.930	0.852	0.838	0.880	*0.995*
		C4.5	0.648	0.427	0.662	0.842	0.808	0.909	0.802	*0.985*
		PCA	*0.777*	0.723	0.564	0.255	0.785	0.755	0.720	0.695

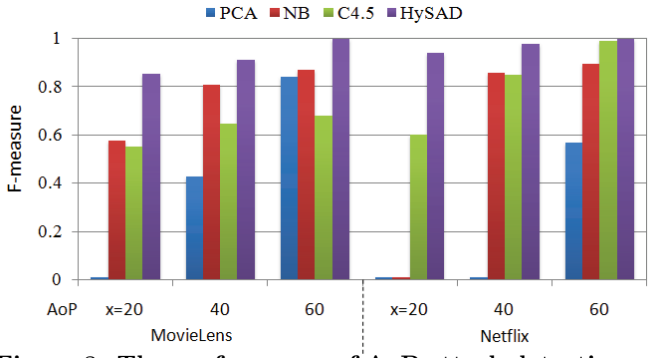


Figure 8: The performance of AoP attack detection.

sion value than $HySAD_3$, but its recall values are typically much smaller than the values of $HySAD_3$. This observation indicates that $HySAD_3$ indeed achieves a subtle balance between NB and $HySAD_2$, to detect as many hybrid attackers as possible, but avoids many false alarms in the meanwhile.

We also observe the comprehensiveness of the four detectors in detecting different types of attacks. Fig. 7 shows the numbers of recognized attackers generated by different attack models, where "hyb." was injected and filler size was set to 15. As can be seen, compared with $HySAD_3$, NB and C4.5 seem mediocre for the random attack, and PCA works poorly for the average attack. In contrast, $HySAD_3$ detects all types of attacks simultaneously, thereby demonstrating that it is particularly suitable for hybrid attack detection.

It was reported in [8] that, AoP attack is a simple and effective obfuscated strategy, which is hard to detect with an average attack. We therefore also tested the four detectors on the average attack with AoP, where the parameter $x = 20, 40, 60$, respectively. Note that to increase the difficulty, we adjusted the set of labeled users L to contain only 60 normal users and 30 attackers for both datasets. Empirical results suggested that $d = 5$ and $\lambda = 1$ for HySAD. Fig. 8 displays the comparative results. As can be seen, HySAD performs the best among the four detectors, and the performance is relatively robust with all the x values. In contrast, NB and PCA do not clearly detect AoP attacks, especially when x is small. To our surprise, C4.5 is second only to HySAD in AoP attack detection.

7. APPLICATION TO PRODUCT RECOMMENDATION: THE AMAZON.CN CASE

Amazon China (http://www.amazon.cn), founded in May 2000 and headquartered in Beijing, is the China operation of Amazon, the world's largest e-commerce company. Everyday, over 2.6 million products are browsed by tens of thousands of users on Amazon.cn, and a huge volume of product reviews are published online continuously. These reviews provide valuable information about products to consumers, but are probably fraudulently used by shilling attackers to promote special products. In this case study, we exploit HySAD for shilling attack detection on Amazon.cn.

7.1 Data Preprocessing

A review dataset from Amazon.cn was used in this situation, which contained 2347178 reviews from 49289 users on 504170 products. Each record contains the information with the following attributes: review_ID, User_ID, ASIN (the product identifier), review title, overall rating, created date, updated date, and is_verified_purchase (a boolean variable indicates whether the user indeed bought that item).

Considering the fact that target items tend to have many ratings, we first extracted the products which had been rated over 100 times. We then selected the users who had rated more than 15 extracted products. After that, we obtained a

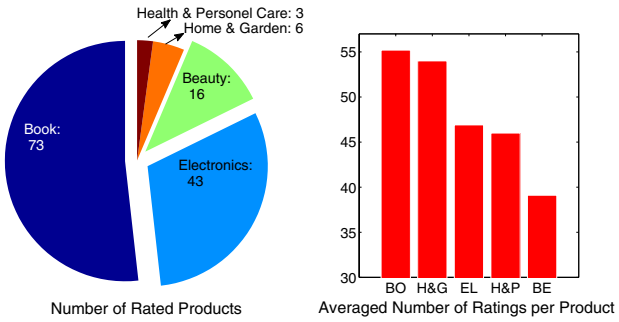


Figure 9: Product categories vulnerable to attacks.

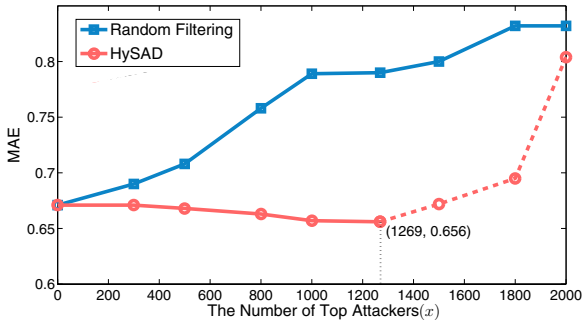


Figure 10: The enhancement of UCF by HySAD.

user subset D, which contained 130540 ratings of 3005 users on 2321 products. In general, D is expected to contain a much bigger proportion of attackers than the raw data.

7.2 Attacker Recognition

Two steps were employed for attacker recognition:

Step 1: Generating L. Let $L = \emptyset$, $d = 10$, $\lambda = 1$, and $\eta = 1$. We ran HySAD$_2$ on D to obtain the unsupervised classification results for two classes. The results were then sent to the experts at Amazon.cn for manual labeling, from whom we received 50 attackers and 100 normal users. We further divided the 50 attackers into 34 AFM attackers and 16 RFM attackers, according to their feature values, and finally obtained L of three classes.

Step 2: Running HySAD. We ran HySAD$_3$ on D using L, with $d = 5$, $\lambda = 1$, and $\eta = 1$, and got 1269 attackers and 1736 normal users. We then ranked the attackers in the decreasing order of $\max\{\Omega_{AF}, \Omega_{RF}\}$. The selected five features are TMF, WDA, $LengthVar$, $RDMA$ and $MeanVar$, which are quite different to the ones for the `MovieLens` and `Netflix` data sets.

We also investigated what types of products were more easily subjected to shilling attacks on Amazon.cn. To this end, we counted the ratings of attackers on the frequently rated products (≥ 30 ratings). As can be seen from Fig. 9, the five categories, Book (BO), Electronics (EL), Beauty (BE), Home & Garden (H&G), and Health & Personal Care (H&P), contain the products most rated by the attacks, although the average number of ratings per product indicates a different order of the five categories.

7.3 Recommendation Enhancement

Given the ranked 1269 attackers, we can now improve the accuracy of a recommender system. The User-based Collaborative Filtering (UCF) method [7] was used for product recommendation. It adopts the weighted Pearson correlation coefficient as the similarity measure (with a threshold 0.1), and sets the number of nearest neighbors to 20. The products with top-10 scores were recommended to users in the test set. The widely adopted measure Mean Absolute Error (MAE) [12], was used for recommendation evaluation. A smaller MAE indicates a higher recommendation quality.

Fig. 10 shows the recommendation quality of UCF after filtering out top-x attackers. As can be seen from the dotted line, the MAE value firstly decreases with the increase of x, until the maximum value 1269 is met. It then increases with $x \geq 1500$. This implies that the attackers detected by HySAD indeed degrade the performance of UCF, which in turn justifies the effectiveness of HySAD. For comparison, we also tested UCF after filtering out x users randomly (denoted as the *random filtering* scheme), and obtained the squared line in Fig. 10. Apparently, random filtering harms, rather than produces a good response for UCF.

7.4 Attacker Analysis

Here, we take a closer look at the attacker behaviors. We selected the top-20 attackers and analyzed them from three aspects: review behavior, purchase behavior, and Amazon.cn ranking. The review and purchase information in the original data and the ranking information on Amazon.cn were used for this study. It is interesting to note that the top-20 attackers can be further divided into three categories:

- **Pusher** (9 users). Pushers are the evident attackers, who use almost the same adwords as product reviews, purchase none of the reviewed products, and have no ranking from Amazon.cn. For instance, the attacker with ID 1095061968 rated many electronic products, but only left the following review: *Dirty cheap! Believe it or not! 40% discount for all products!*

- **Tricker** (7 users). Trickers are more cautious attackers. They often generate diverse reviews, and have Amazon.cn rankings as normal users, although the rankings may be low. However, the purchase behaviors betray them. They seldom buy rated products, although often claim that they did. Moreover, they only review products in the same category. The attacker "joe5138" is such an example, who gave five points to all the books in a certain financial series, but actually never bought any of them (information can be found at http://www.amazon.cn).

- **Lurker** (4 users). Lurkers act almost like normal users. They review diverse products, give diffrent ratings and abundant reviews, buy some of the rated products, and may have a high Amazon.cn ranking. The only problem is, they often rate a lot of products in a very short period. From this aspect, a lurker may be a professional/parttime product reviewer of Amazon.cn, or a normal user who often seeks online discounts by publishing reviews. For instance, the user "Xiao'an" is lurker who reviewed two Motorola cell-phones, two Nokia cell-phones, one laptop, seven books, one sunscreen, and three bags for men and women in a very short period at September 11, 2008 (information can be found at http://www.amazon.cn).

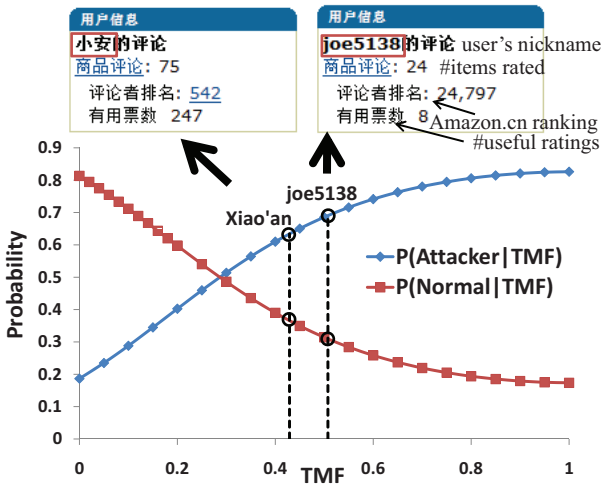


Figure 11: The TMF of two hidden attackers.

Fig. 11 shows one reason why HySAD considered "joe5138" and "Xiao'an" as attackers. As can be seen, judged by the Target Model Focus (TMF) metric, these two users deviate heavily from the normal users. This implies that both of them tend to give high ratings to items that are rated frequently and highly by some other users, i.e. the potential target items in shilling attacks. This, in turn, illustrates that HySAD has a good model interpretability, which is very important for real-world applications. Note that the two upper boxes in Fig. 11 are given by Amazon.cn.

8. CONCLUSIONS

This paper proposes a system called HySAD for hybrid shilling attack detection. In general, HySAD is a semi-supervised learning system that makes use of both unlabeled and labeled user profiles for multi-class modeling. Experimental results demonstrate that HySAD is particularly effective against hybrid attacks, even when presented with obfuscated strategies. A real-life case study on Amazon.cn also demonstrates the effectiveness of HySAD in improving the performance of a collaborative-filtering recommender system, and the ability of HySAD to help explore interesting attacker behaviors. These, in turn, justify the value of HySAD for real-world applications.

ACKNOWLEDGEMENTS

This research was partially supported by the National Natural Science Foundation of China (NSFC) (Nos. 61103229, 71072172), Industry Projects in the Jiangsu S&T Pillar Program (No. BE2011198), and Jiangsu Provincial Colleges and Universities Outstanding S&T Innovation Team Fund(No. 2001013). The second author was also supported in part by the National Natural Science Foundation of China (NSFC) (Nos. 70901002, 71171007, 70890080, 71028002).

9. REFERENCES

[1] S. Asch. Effects of group pressure upon the modification and distortion of judgments. In H. Guetzkow, editor, *Groups, Leadership, and Men: Research in Human Relations*, pages 17–190. Pittsburgh: Carnegie Press, 1951.

[2] J. Bennett and S. Lanning. The netflix prize. In *Proceedings of KDD Cup and Workshop*, San Jose, California, USA, 2007.

[3] R. Burke and B. M. et al. Classification features for attack detection in collaborative recommendation systems. In *KDD'06*, 2006.

[4] R. Burke, B. M. C. Williams, and R. Bhaumik. Segment-based injection attacks against collaborative filtering recommender systems. In *ICDM'05*, 2005.

[5] V. Castelli and T. Cover. On the exponential value of labeled samples. *Pattern Recognition Letters*, 16(1):105–111, 1995.

[6] P. Chirita, W. Nejdl, and C. Zamfir. Preventing shilling attacks in online recommender systems. In *WIDM'05*, pages 67–74, 2005.

[7] J. Herlocker, J. Konstan, L. Terveen, and J. Riedl. Evaluating collaborative filtering recommender systems. *TOIS*, 22(1):5–53, 2004.

[8] N. Hurley, Z. Cheng, and M. Zhang. Statistical attack detection. In *RecSys'09*, 2009.

[9] S. Lam and J. Riedl. Shilling recommender systems for fun and profit. In *WWW'04*, pages 393–402, 2004.

[10] J. Lee and D. Zhu. Shilling attack detection—a new approach for a trustworthy recommender system. *INFORMS Journal on Computing*.

[11] H. Liu, H. Motoda, and L. Yu. Feature selection with selective sampling. In *ICML'02*, 2002.

[12] Q. Liu, E. Chen, H. Xiong, H. Ding, and J. Chen. Enhancing collaborative filtering by user interest expansion via personalized ranking. *TSMCB*, 2011.

[13] B. Mehta and W. Nejdl. Unsupervised strategies for shilling detection and robust collaborative filtering. *User Modeling and User-Adapted Interaction*, 19(1-2):65–97, 2009.

[14] B. Mobasher, R. Burke, C. Williams, and R. Bhaumik. Analysis and detection of segment-focused attacks against collaborative recommendation. In *WebKDD Workshop*, 2006.

[15] B. Mobasher and R. B. et al. Toward trustworthy recommender systems: An analysis of attack models and algorithm robustness. *TOIT*, 7(4):1–40, 2007.

[16] K. Nigam, A. Mccallum, S. Thrun, and T. Mitchill. Text classification from labeled and unlabeled documents using em. *ML*, 39(2):103–134, 2000.

[17] M. O'Mahony, N. Hurley, N. Kushmerick, and G. Silvestre. Collaborative recommendation: A robustness analysis. *TOIT*, 4:344–377, 2004.

[18] F. Ricci, L. Rokach, B. Shapira, and P. Kantor. *Recommender Systems Handbook*. Spinger, 2011.

[19] X. Su, H.-J. Zeng, and Z. Chen. Finding group shilling in recommendation system. In *WWW'05*, 2005.

[20] P. Tan, M. Steinbach, and V. Kumar. *Introduction to Data Mining*. Addison-Wesley, 2005.

[21] C. Williams. Profile injection attack detection for securing collaborative recommender systems. Technical report, DePaul University, 2006.

[22] S. Zhang, A. Chakrabarti, J. Ford, and F. Makedon. Attack detection in time series for recommender systems. In *KDD'06*, 2006.