hPSD: A Hybrid PU-Learning-Based Spammer Detection Model for Product Reviews

Zhiang Wu, *Member, IEEE*, Jie Cao, Yaqiong Wang, Youquan Wang, Lu Zhang, and Junjie Wu

***Abstract*—Spammers, who manipulate online reviews to promote or suppress products, are flooding in online commerce. To combat this trend, there has been a great deal of research focused on detecting review spammers, most of which design diversified features and thus develop various classifiers. The widespread growth of crowdsourcing platforms has created large-scale deceptive review writers who behave more like normal users, that the way they can more easily evade detection by the classifiers that are purely based on fixed characteristics. In this paper, we propose a hybrid semisupervised learning model titled hybrid PU-learning-based spammer detection (hPSD) for spammer detection to leverage both the users' characteristics and the user–product relations. Specifically, the hPSD model can iteratively detect multitype spammers by injecting different positive samples, and allows the construction of classifiers in a semisupervised hybrid learning framework. Comprehensive experiments on movie dataset with shilling injection confirm the superior performance of hPSD over existing baseline methods. The hPSD is then utilized to detect the hidden spammers from real-life Amazon data. A set of spammers and their underlying employers (e.g., book publishers) are successfully discovered and validated. These demonstrate that hPSD meets the real-world application scenarios and can thus effectively detect the potentially deceptive review writers.**

***Index Terms*—Positive and unlabeled dataset learning (PU-learning), semisupervised learning, spammer detection, user–product relations.**

I. Introduction

REVIEWS and ratings have become the necessary ingredients of online commerce, ranging from e-commerce sites (e.g., Amazon, eBay, and Taobao), and online-to-offline sites (e.g., Yelp and Dianping), to travel booking sites (e.g., TripAdvisor, hotels.com, and Ctrip). Online product ratings and reviews strongly influence the purchase decisions of a vast number of customers, and thus are strongly correlated with sales [1]–[3]. A high proportion of positive reviews combined with a high average rating undoubtedly bring significant financial gains. In contrast, the negative reviews cause sales loss. Driven by these financial incentives, some product providers and/or merchants try to register a number of fake user IDs or hire people to promote or suppress target products by creating falsified, biased online reviews, and extreme ratings. Meanwhile, the brokers have emerged to guide people who want to make money for promoting products or online stores, like "shuakewang"[1] in China. Surprisingly, it has been shown that up to 6% of the reviews on sites like Yelp and TripAdvisor may be falsified [4]. Hence, product review spammer detection [5]–[7] has attracted an increasing amount of attention in the recent years.

There have been considerable research on identifying the spam users [5], [8]–[12], where most approaches rely on analyzing the review contents and reviewer behaviors to construct various classification models. Accordingly, the performance of existing approaches is largely determined by whether the abnormal behaviors of spam users can be characterized by the key elements. However, some spammers, e.g., the users hired by malicious brokers, are unlikely to leave identifiable footprints [7]. Let us take the homepage of an Amazon.cn user named *w_feifei*,[2] for example, who has bought a lot of books, and has written plenty of reviews that gained a good deal of useful votes from other users. From these points, she should be identified as a legitimate user. Nevertheless, when we take a close look at the products she reviewed, it is surprising that nearly all the books were published by a same press: Zhongxin. This unusual phenomenon indeed indicates that she is more like a promoter for this particular book publisher. We

Manuscript received August 7, 2018; accepted October 16, 2018. This work was supported in part by the National Key Research and Development Program of China under Grant 2016YFB1000901, in part by the National Natural Science Foundation of China (NSFC) under Grant 71571093, Grant 91646204, Grant 71701089, and Grant 71801123, in part by the National Center for International Joint Research on E-Business Information Processing under Grant 2013B01035. The work of J. Wu was supported by NSFC under Grant 71725002, Grant 71531001, Grant U1636210, and Grant 71471009, and in part by the Fundamental Research Funds for Central Universities. This paper was recommended by Associate Editor Y. Tan. *(Corresponding authors: Jie Cao; Junjie Wu.)*

Z. Wu, Y. Wang, and L. Zhang are with the Jiangsu Provincial Key Laboratory of E-Business, Nanjing University of Finance and Economics, Nanjing 210003, China (e-mail: zawuster@gmail.com; youq.wang@gmail.com; luzhang@njue.edu.cn).

J. Cao is with the School of Information Engineering, Nanjing University of Finance and Economics, Nanjing 210003, China, and also with the College of Computer Science and Engineering, Nanjing University of Science and Technology, Nanjing 210094, China (e-mail: jie.cao@njue.edu.cn).

Y. Wang is with the Carlson School of Management, University of Minnesota, Minneapolis, MN 55455 USA (e-mail: wang6285@umn.edu).

J. Wu is with the School of Economics and Management, Beihang University, Beijing 100191, China, also with the Beijing Advanced Innovation Center for Big Data and Brain Computing, Beihang University, Beijing 100191, and also with the Beijing Key Laboratory of Emergency Support Simulation Technologies for City Operations, Beihang University, Beijing 100191, China (e-mail: wujj@buaa.edu.cn).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TCYB.2018.2877161

[1] http://www.shuakewang.com

[2] https://www.amazon.cn/gp/pdp/profile/A3AOAUPEFI527X/

can summarize from this typical example that the signals of spam humans, rather than spam bots, are not easily identifiable and thus make the ongoing detection quite challenging. The positive side also exists; that is, the linkage between the user and product could unveil some latent but valuable clues to spammers. The logic is, if a user often write reviews for polluted products, this user behaves more like a spammer; in contrast, if a product is usually reviewed by spammers, then the product is more polluted and the merchant selling this product is highly suspicious.

Despite of the promise, the challenges remain significant. On one hand, the spamming behaviors can be categorized into multiple types, for example, Jindal and Liu [13] summarized three types of them, but it is difficult to unify the detection of multitype spammers with different characteristics into one model. On the other hand, it remains largely unclear how many spammers reside in a real-life e-commerce site and which users are true spammers, since labeling a user manually is very expensive. Hence, the fact that the labeled users are scarce but the unlabeled users abound should be fully considered when building a detection model. In addition, the lack of true labels could make the model validation on real-world data very difficult.

In this paper, we focus on review spammer detection or identifying users who are the source of spam reviews or biased ratings. Specifically, we establish a hybrid PU-learning-based spammer detection (hPSD) model to connect two forms of data, including users' features and user–product relations. Several characteristics of hPSD model makes it well suited for the spammer detection in real-life scenario. First, it adopts the positive and unlabeled dataset learning (PU-learning) [14], [15] scheme to detect multitype spammers, where the positive (P) and unlabeled (U) set denote injected spammers and unknown users, respectively. Also, a novel reliable negative (RN) set extraction algorithm is suggested to be a key component of PU-learning. Second, a hybrid learning model based on Bayesian inference is included so that the user–product relationship can be incorporated into the traditional learning on the feature space. Third, hPSD is a semisupervised learning model that makes full use of both labeled and unlabeled datasets.

Comprehensive experiments are carried out on both the movie data with shilling injection and the Amazon data with true yet hidden spammers. Experiments on movie datasets provide comparative results with eight state-of-the-art shilling attack detectors to demonstrate the advantage of hPSD, and explore several of the impact factors of hPSD. Then, the hPSD model is devoted to detect the duplicate spammers and promoters from the Amazon.cn dataset. As will be shown later, we are sure to identify a host of spammers and book publishers under high suspicion of hiring a water army for marketing.

Overview: The reminder of this paper is organized as follows. In Section II, we present the related work. Section III defines the problems affecting the study and gives a model overview. In Section IV, we address technical details of each step within the hPSD model. We show the experimental results in Section V, and finally conclude this paper in Section VI.

II. Related Work

In the literature, a great deal of research has been undertaken to detect three target variables: 1) review/opinion spam; 2) spam users; and 3) spammer groups. These studies mainly focus on two subareas: 1) the spamming behavior features [5], [8]–[10], [16]–[19] and 2) the spam classification approaches [7], [9], [10], [12], [20], [21]. In what follows, we shall review some representative studies in both subareas.

A. Spamming Behavior Features

The indicative features of spam are constructed from available metadata (e.g., ratings, timestamps, review text, etc.), and they can be converted to prior class probabilities that are then used to build a classifier. Generally, the indicative features can be categorized into two classes: 1) linguistic and 2) behavioral features. Meanwhile, one feature can be defined on a single review or all reviews of a user/product. As in the earlier studies, the reviews containing similar content are identified as spam reviews [8], [16]. To detect such spam review, an abundant of linguistic features (e.g., ratio of objective/subjective words computed by WordNet [22]) have been designed based on review content [22], [23]. However, the behavioral features are based on ratings, timestamps, ranks, distributions, and so on. For example, in shilling attack detection [17]–[19], a number of features are defined based on the length variance, deviation, and similarity among rating vectors. Also, many of the behavioral features (e.g., max number of reviews in one day) based on timestamps and ranks of reviews are suggested in [9] and [10]. These studies inspire us in defining the characteristics of interest in our experiments.

Researchers have also begun analyzing how to detect groups of review spammers [9], [24], [25]. Spammers within a group usually work together to commit swarm attacks, which implies that analyzing user relationships might be helpful to reveal the deceptive behavior. Along this line, some recent studies [7], [12] proposed utilizing these relationships to define new topological features. So far, commonly used topological features include: degree, PageRank score, centrality scores, triangle count, and cluster_ID generated by graph clustering.

B. Spam Classification Approaches

The existing spam detection algorithms can also be categorized into two classes. First, a handful of studies attempt to employ unsupervised methods for assigning a score (i.e., probability) to rank suspected spammers [9], [10], [26], [27]. For instance, Mehta and Nejdl [28] utilized the principal component analysis (PCA) on user–product rating matrix to detect shilling attackers, and Mukherjee *et al.* [10] adopts the topic model to identify opinion spammers. Second, a majority of approaches leverage indicative features of spam for building supervised classification models [7], [12], [19], [20], [29]. Very few research has been contributed in developing the semisupervised detection models. For instance, Wu *et al.* [21] have shown the effectiveness of semisupervised learning on shilling attack detection, that is, it is more suitable for the real-world e-commerce sites that are lacking labeled spammers. Nevertheless, the spam users who utilize clever disguises are

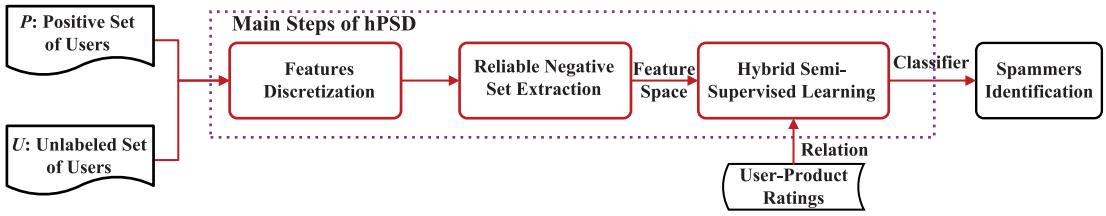


Fig. 1. Overview of hPSD model.

likely to evade detection by these classifications solely based on their tracked characteristics alone. A few studies propose to exploit the relationship network to enhance the produced classifications [6], [7], [12], [20]. Wang *et al.* [6] considered the user–review–product network and utilizes an iterative algorithm similar to HITS to compute scores for trustworthiness of users. Wu *et al.* [20] designed a supervised hybrid learning model based on logistic regression for detecting marionette microblog users. Fakhraei *et al.* [12] employed the topological features and their sequential combinations to train a supervised classifier. To the best of our knowledge, however, no theoretical semisupervised learning models to connect the behavioral features with the relation network have been proposed as solutions.

III. hPSD: Model Overview

Given a set U of n users and a set I of m products, $R_{i,j} \in \{0, 1\}$ denotes a binary relation variable, indicating user i has rated product j. A rating usually comes with other information, such as a review, time, etc., based on which a set of features (denoted as F) can be obtained. Each user is then represented as a feature vector $u_i \in \mathbb{R}^{|F|}$. Generally, the hPSD model exploits PU-learning for spammer detection. So, only a positive seed set (denoted as P) of labeled spammers is needed for training the hPSD model, rather than both the labeled spammers and legitimate users. The benefits of this PU-learning setting inside hPSD are twofold: it not only makes the model simpler to use and more flexible but also saves on the labor-intensive effort of labeling legitimate users manually.

We now define the spammer detection problem as: given P, U represented in the feature space and relation variable R, our goal is to analyze these data jointly and to recognize spam users from U. Fig. 1 illustrates the diagram of our hPSD model. As can be seen, the operations in our model can be divided into three main steps.

1) *Feature discretization* transforms every continuous feature into categorical values, based on the distribution of feature values on the entire dataset $P \cup U$. The statistical distribution of unlabeled set is expected to enhance the discriminative power of each feature. Also, the following two steps are designed upon discretized features.
2) *RN set extraction* targets at singling out a set of legitimate users from U with respect to P. Then, this RN set combined with the P set will serve as the labeled set that will be used to guide the next model training step.
3) *Hybrid semisupervised learning* undertakes the core task to build the classifier for spammer detection. This framework utilizes both users' feature vectors and user–product relationship for so-called "hybrid learning," and also exploits the "semisupervised learning" for building the classifier on both labeled and unlabeled sets.

One salient feature of the hPSD model is that it aims to identify the spammers who are similar with those users in the seed set P. When facing multitype spammers with different characteristics, we can construct the set P more than once, and repeatedly run above three steps to iteratively detect spammers.

IV. hPSD: Technical Details

In this section, we address the technical details of hPSD, involving feature discretization, RN set extraction, and inference for hybrid learning.

A. Feature Discretization

Most behavioral features used for spammer detection are numerical [5], [9], [10], [19]. A common way to model continuous features is to assume a probabilistic distribution, for example, Gaussian distribution. However, without any prior knowledge about the dataset, the choice of a reasonable assumptive distribution is often difficult. Meanwhile, even if the distribution is known in advance, the distributions on labeled and unlabeled sets are likely to be nonidentical, given that only a few instances could be manually labeled. We, therefore, propose to discretize each continuous feature using a clustering method.

Without loss of generality, let S be a feature value list in ascending order of any feature f on $P \cup U$. Thus, the feature discretization aims to find $v - 1$ cut points in S, that is, discretizing f into v categories. In [30] and [31], finding the "best" cut points is not a trivial task. Included in the criterion for choosing a cut point, the minimal weighted average variance (WAV) [31] is used for simplicity. Let $S_1^v \cup S_2^v = S$ denote a partition of S according to a value v. The WAV of v on S is defined as

$$\mathrm{WAV}_{vS} = \frac{|S_1^v|}{|S|}\mathrm{Var}\left(S_1^v\right) + \frac{|S_2^v|}{|S|}\mathrm{Var}\left(S_2^v\right) \tag{1}$$

where $|S|$ is the number of points in this list and $\mathrm{Var}(S_1^v)$ is the variance of all points in this list [analogical to $|S_1^v|$, $|S_2^v|$ and $\mathrm{Var}(S_2^v)$]. Thus, the "best" cut point is selected according to the maximum value of $\Delta_v = \mathrm{Var}(S) - \mathrm{WAV}_{vS}$.

To implement the multi-interval discretization, we develop a binary-recursive algorithm called bisecting V-clustering algorithm (BiVC) to divide S into v sublists. The pseudo-codes of BiVC is summarized in Algorithm 1. BiVC first divides all the

Algorithm 1 BiVC Algorithm

Input: The sorted value list S of a feature f; The number of categories ν;
Output: A set of cut points $\mathcal{V} = \{v_1, v_2, \cdots, v_{\nu-1}\}$;
1: **procedure** BiVC(S, ν)
2: **while** $|\mathcal{V}| < \nu$ **do** ▷ $\mathcal{V}$ divides S into a set of sub-lists $\{S_1, \cdots, S_{|\mathcal{V}|+1}\}$
3: Select a sub-list denoted as S_j, $1 \le j \le |\mathcal{V}| + 1$ with the largest range;
4: $\forall v_i \in S_j$, compute Δ_{v_i} on S_j with Eq. (**??**);
5: $q = \arg\max_i \Delta_{v_i}$, $\mathcal{V} \leftarrow \mathcal{V} \bigcup \{v_q\}$;
6: **end while**
7: **end procedure**

value-points of f (i.e., S) into two clusters based on the best cut point; and then repeatedly selects one cluster according to a certain criterion (e.g., with the largest cluster in this paper) and divides that cluster into two clusters based on the best cut point again. The procedure will continue unless ν clusters are found. After discretizing every feature into ν intervals, a user profile can thus be converted to a binary vector with $\nu|F|$ dimensions, that is, $u_{i,l} \in \{0, 1\}$, $1 \le i \le n$, $1 \le l \le \nu|F|$, denotes whether the ith user's feature is within the lth interval. In what follows, $u_i \in \mathbb{R}^{\nu|F|}$ refers to a categorical feature vector if not otherwise specified.

1) Discussion: It is worth noting that BiVC runs on the entire dataset (i.e., $P \cup U$) for feature discretization. This setting considers the value distribution of each feature on both the labeled and unlabeled datasets, and utilizes the clustering technique to derive a reasonable partitioning for every feature. In other words, the statistical distribution of the unlabeled datasets has been incorporated into the categorical features.

B. Reliable Negative Set Extraction

The RN set consists of a small set of instances extracted from U that are remarkably different from the instances in the positive set P [32]. We do not require RN to contain ample instances, but the included instances should be "reliable," that is, they should be with distinct characteristics compared with the instances in P. To alleviate the class imbalance in the ongoing learning step, we propose to extract an RN set with $|\text{RN}| \approx |P|$.

In text classification [14], an RN document is extracted from U if it does not contain any word that is listed in the core vocabulary of P. We, therefore, generalize the RN set extraction problem as an optimization problem; that is, it targets at extracting a set of RN instances from U so as to maximize the discriminative power of core features. The objective function is

$$O_1 : \max_{f \in F^c} D_f(P \cup \text{RN}) \tag{2}$$

where $F^c \subseteq F$ is the set of core features and $D_f(\cdot)$ denotes a feature strength function. Obviously, maximizing (2) is a combinatorial optimization, an NP-hard problem. As an alternative, we present a greedy RN set extraction heuristic that defines a proper D_f function to sort all features and maximizes $D_f(P \cup \text{RN})$ for every feature.

As mentioned in Section IV-A, every feature has been discretized into ν values and $\nu|F|$ feature-value pairs form a vocabulary. Let f be a feature-value pair, a general expression of u_{il}. Then, we can utilize a two-way table, as shown in Table I, to represent each binary feature. Intuitively, a lot of widely used metrics, such as information gain, χ^2 and odd ratio can be used to define D_f. Based on the fact that $a+c$ and $b+d$ are constants, we propose to define a simplified feature strength function exploiting only a and b as

TABLE I
ILLUSTRATIVE TWO-WAY TABLE

		Data Partitions	
		P	U
Feature Value	1	a	b
	0	c	d

Algorithm 2 RN Set Extraction

Input: Positive set P; Unlabeled set U;
Output: A set of reliable negative instances RN;
1: **procedure** RN_EXTRACTION(P, U)
2: **for** each feature $f_l \in P$ **do** ▷ only consider features appearing in P
3: Compute D_f using Eq. (3);
4: **end for**
5: $RN \leftarrow U$; ▷ Initially, RN contains all instances of U
6: **for** examine each feature f in D_f-decreasing order **do**
7: Remove instances containing f from RN;
8: **if** the size of RN is close to that of P **then**
9: **return** RN;
10: **end if**
11: **end for**
12: **end procedure**

$$D_f = n_P(f) \log \frac{|P| + |U|}{n_P(f) + n_U(f)} = a \log \frac{n}{a+b} \tag{3}$$

where $n_P(f) = a$ and $n_U(f) = b$ are the number of instances containing f in P and U, respectively. The D_f function shown in (3) implies that if a feature is discriminative with respect to the class P, it would frequently appear in the P set rather than the U set.

For any feature, since $b \swarrow \Rightarrow D_f \nearrow$, to remove instances with $f = 1$ from U (i.e., decreasing b to 0) will maximize the objection function. Hence, the RN extraction problem is formulated as: given $\text{RN} = U$ initially and a list of features in D_f-decreasing order, to remove the instances containing this feature in RN, until the scale of RN is reduced approximately to that of P. Algorithm 2 gives the pseudo-codes of the above procedure. Note that due to the uncontrollable b, |RN| might not be equal to $|P|$ exactly. In line 8, we establish the exit criterion as |RN| being closest to $|P|$.

C. Hybrid Semisupervised Learning

The combination of injected P and extracted RN forms a two-class labeled set, denoted as $L = P \cup \text{RN}$. So, the entire data is $\mathcal{D} = L \cup U$ containing a few labeled but the majority being unlabeled users. Since, the spammer detection is essentially a binary classification problem, we denote the class labels as y_k, $k \in \{0, 1\}$ and assume that given a class label y_k, each feature is viewed as a Bernoulli trial with a probability of success equal to $\theta_{k,l}$ with respect to the class y_k. Thus, the multinomial distribution over all features is characterized by the parameters $\theta_k \in \mathbb{R}^{\nu|F|}$. In our scenario, each instance is assumed to be drawn independently from a mixture distribution of $|\{y_k\}|$ classes and thus the probability that u_i belongs to class y_k is

$$p(y_k|u_i; \theta_k) = \frac{z_k p(u_i|y_k; \theta_k)}{\sum_k z_k p(u_i|y_k; \theta_k)} \tag{4}$$

where $z_k = p(y_k)$ is the prior probability of class y_k satisfying $\sum_k z_k = 1$. According to the assumption of multinomial distribution of user features, we define the class-conditional probability of an instance as

$$p(u_i|y_k;\theta_k) \propto \prod_{l=1}^{v|F|} \theta_{kl}^{u_{il}}. \tag{5}$$

Based on above preliminaries, we seek to learn the classification model by maximizing conditional likelihood of the entire dataset. The proposed model tries to incorporate user–product relation into the traditional learning on the feature space. To this end, we formulate the conditional likelihood, i.e., the objective function, as follows:

$$O_2 : \max_{\theta} \log \prod_{u_i \in \mathcal{D}} \left\{ \underbrace{p(y_k|u_i;\theta_k)^{\Lambda_i}}_{\text{(a)}} \underbrace{\prod_{I_j \in R_{i\cdot}} p\big(y_k|I_j;\theta_k\big)^{\frac{d}{|R_{i\cdot}|}}}_{\text{(b)}} \right\} \tag{6}$$

satisfying $\sum_{l=1}^{v|F|} \theta_{k,l} = 1$, and $\sum_k z_k = 1$, where

$$p\big(y_k|I_j;\theta_k\big) = \prod_{u_i \in R_{\cdot j}} p(y_k|u_i;\theta_k)^{\Lambda_i}, \text{ and } \Lambda_i = \begin{cases} \lambda, & \text{if } u_i \in U \\ 1, & \text{if } u_i \in L. \end{cases}$$

In (6), $R_{i\cdot}$ is a set of products rated by u_i, and $R_{\cdot j}$ is a set of users rating I_j. The part (a) of (6) is the probability learned from the feature space (i.e., conditionally on u_i), where $\lambda \in [0,\ 1]$ is the weight that reduces the impact of the unlabeled set. In contrast, part (b) is responsible for modeling the potential of user–product relation (i.e., conditionally on products I_j rated by u_i), where $d \in [0, 1]$ is the coefficient that balances between the feature space and the user–product relation. With $p(u_i|y_k;\theta_k)$ and the Bayesian theorem, the conditional likelihood function for each instance, that is, $p(y_k|u_i;\theta_k)$, is defined as

$$p(y_k|u_i;\theta_k) = \frac{z_k p(u_i|y_k;\theta_k)}{\sum_k z_k p(u_i|y_k;\theta_k)} \propto \frac{z_k \prod_{l=1}^{v|F|} \theta_{kl}^{u_{il}}}{\sum_k z_k \prod_{l=1}^{v|F|} \theta_{kl}^{u_{il}}}. \tag{7}$$

By putting the objective and constraints together, we get the Lagrange function as

$$\begin{aligned} l : \max_{\theta} \log \prod_{u_i \in \mathcal{D}} \left\{ p(y_k|u_i;\theta_k)^{\Lambda_i} \prod_{I_j \in R_{i\cdot}} p\big(y_k|I_j;\theta_k\big)^{\frac{d}{|R_{i\cdot}|}} \right\} \\ + \sum_{k=0}^{1} \xi_k \left(\sum_{l=1}^{v|F|} \theta_{kl} - 1 \right) + \omega \left(\sum_{k=0}^{1} z_k - 1 \right). \end{aligned} \tag{8}$$

In what follows, we proceed to estimate the two sets of parameters z_k and θ_k via an EM-like algorithm. We consider a method called stochastic gradient descent (SGD) [33], which incrementally updates parameters to increase the conditional log likelihood through one example at a time. In SGD, the derivative based on a random sample is treated as an approximation to the derivative based on all the training data. Consider a single sample u_i, differentiating (8) with respect to z_k yields

$$\frac{\partial l}{\partial z_k} = \frac{\Lambda_i}{z_k} - \frac{\Lambda_i V_k}{\sum_k z_k V_k} + \sum_{I_j \in R_{i\cdot}} M_i \sum_{u_i \in R_{\cdot j}} \left(\frac{\Lambda_i}{z_k} - \frac{\Lambda_i V_k}{\sum_k z_k V_k} \right) + \omega \tag{9}$$

where $V_k = \prod_{l=1}^{v|F|} \hat{\theta}_{kl}^{u_{il}}$ and $M_i = [d/(|R_{i\cdot}|)]$. Thus, let the derivative in (9) be zero. We get

$$\begin{aligned} C &= \frac{1}{1-p+\frac{p}{z_0}} + \sum_{I_j \in R_{i\cdot}} \frac{d}{|R_{\cdot j}|} \sum_{u_i \in R_{\cdot j}} \frac{1}{1-q+\frac{q}{z_0}} - \omega z_0 \\ &= \frac{1}{1-\frac{1}{p}+\frac{1}{pz_1}} + \sum_{I_j \in R_{i\cdot}} \frac{d}{|R_{\cdot j}|} \sum_{u_i \in R_{\cdot j}} \frac{1}{1-\frac{1}{q}+\frac{1}{qz_1}} - \omega z_1 \end{aligned}$$

where $z_0 + z_1 = 1$, $C = 1 + d|R_{i\cdot}|$, $p = \prod_{l=1}^{v|F|} (\theta_{1l}/\theta_{0l})^{u_{il}}$, and $q = \prod_{l=1}^{v|F|} (\theta_{1l}/\theta_{0l})^{u_{il}}$. Based on the above equations, we obtain the following equation:

$$\begin{aligned} \frac{p-1}{p-pz_0+z_0} + \frac{C}{z_0} = \frac{C}{1-z_0} - \sum_{I_j \in R_{i\cdot}} M_i \\ \times \sum_{u_i \in R_{\cdot j}} \frac{q-1}{z_0 - qz_0 + q} = 0. \end{aligned} \tag{10}$$

Since z_0 is difficult to resolve directly, we make some approximations here. Specifically, we take βz_0 as an estimation of 1. This implies that z_0 is initially estimated as the ratio of unlabeled users, and then is estimated as z_0 of last iteration. β would also change with z_0 in each iteration. With this approximation, (10) could be rewritten as

$$\begin{aligned} \frac{p-1}{\beta p z_0 - p z_0 + z_0} + \frac{C}{z_0} = 0 \\ \frac{C}{1-z_0} - \sum_{I_j \in R_{i\cdot}} M_i \sum_{u_i \in R_{\cdot j}} \frac{q-1}{z_0 - qz_0 + \beta q z_0} = 0. \end{aligned} \tag{11}$$

We finally obtain the equations for updating both z_0 and z_1

$$\hat{z}_0 = z_0 + \zeta \frac{X + C + \sum_{I_j \in R_{i\cdot}} M_i \sum_{u_i \in R_{\cdot j}} Y}{X + 2C + \sum_{I_j \in R_{i\cdot}} M_i \sum_{u_i \in R_{\cdot j}} Y} \tag{12}$$

$$\hat{z}_1 = z_1 + \zeta \frac{C}{X + 2C + \sum_{I_j \in R_{i\cdot}} M_i \sum_{u_i \in R_{\cdot j}} Y} \tag{13}$$

where $X = [p - 1/(\beta p - p + 1)]$, $Y = [q - 1/(\beta q - q + 1)]$, and ζ is a positive constant called learning-rate. We follow the common setting $\zeta = 0.01$.

A smoothed estimator of the multinomial distribution is employed for θ_{kl}. Denote a smoothing parameter as α, the number of times the lth feature occurs in y_k class as n_{kl}, and the total number of instances in y_k class as n_k. We have

$$\hat{\theta}_{kl} = \frac{n_{kl} + \alpha}{n_k + v\alpha} \tag{14}$$

We follow a common practice by setting $\alpha = 1$. With (14), updating θ_{kl} is equivalent to updating n_{kl} and n_k as follows:

$$\hat{n}_{kl} = \sum_{u_i \in L_k} u_{il} + \lambda \sum_{u_i \in U} p(y_k|u_i;\theta_k) \prod_{I_j \in R_{i\cdot}} p\big(y_k|I_j;\theta_k\big)^{\frac{d}{|R_{\cdot j}|}} u_{il} \tag{15}$$

$$\hat{n}_k = |L_k| + \lambda \sum_{u_i \in U} p(y_k|u_i;\theta_k) \prod_{I_j \in R_{i\cdot}} p\big(y_k|I_j;\theta_k\big)^{\frac{d}{|R_{\cdot j}|}}. \tag{16}$$

In summary, we adopt an EM-like procedure to maximize the conditional likelihood that combines the feature-space and the user–product relation. In particular, we iteratively update

Algorithm 3 Hybrid Learning Algorithm

Input: Labeled set $L = P \cup RN$; Unlabeled set U; Weights λ, d;
Output: The probability $p(y_k|u_i;\theta_k)$ for each user in U;
1: **procedure** HYB_LEARNING(L, U, λ, d)
2: Compute n_{kl}, n_k, and thus θ_k, z_k on L, initially;
3: **while** $\max_{k,l} |\theta_{jl} - \hat{\theta}_{jl}| \geq \epsilon$ **do**
4: Compute $p(u_i|y_k;\theta_k)$ and thus $p(y_k|u_i;\theta_k)$ by Eqs. (5) and (7);
5: Update z_k, $k \in \{0, 1\}$ by Eqs. (12) and (13);
6: Update n_{kl}, n_k and thus θ_k by Eqs. (15), (16) and (14);
7: **end while**
8: **end procedure**

Algorithm 4 hPSD: The Unified Procedure

Input: Unlabeled set U; Weights λ, d; Metadata $\mathcal{M}$; The number of categories ν
Output: Sets of spam users identified from U;
1: **procedure** HPSD($U, \lambda, d, \mathcal{M}, \nu$)
2: **for** each type of spammers **do** ▷ iterative detection
3: Construct P set containing the specific type of spammers;
4: Extract a set of features for every user in P and U based on metadata $\mathcal{M}$;
5: Discretize each feature as ν categorical variables by invoking Algorithm 1;
6: Extract RN by invoking Algorithm 2, and let $L = P \cup RN$;
7: Compute $p(y_k|u_i;\theta_k)$ for each user in U by invoking Algorithm 3;
8: Assign $\widehat{y}_k^i$ to each u_i according to $\widehat{y}_k^i = \max_k p(y_k|u_i;\theta_k)$;
9: Output the list of users whose $\widehat{y}_k^i = 1$ and remove them from U;
10: **end for**
11: **end procedure**

the parameters z_k and θ_k by differentiating the Lagrange function of our objective function. The hybrid learning procedure is summarized in Algorithm 3.

D. Unified Procedure

Here, we integrate the above three-pronged technical details as a unified procedure of the hPSD model, which is consistent with Fig. 1. Algorithm 4 shows the pseudo-codes of intact hPSD process. In general, hPSD is an iterative process that benefits from the PU-learning framework, that is, it detects one type of spammers in each iteration as shown by the "for" loop in Algorithm 4. The *metadata* $\mathcal{M}$ may include ratings, timestamps, review text, etc., and it is used for extracting the feature set (see line 2). Since the metadata and the feature set are data-specific, we shall give the further explanation in Section V. Then, lines 5–7 correspond to three key steps: 1) feature discretization; 2) RN set extraction; and 3) the hybrid learning, as discussed above. Finally, we assign a predicted label $\widehat{y}_k^i$ to every user in U according to the class probability obtained by the hybrid learning. If a user is identified as the spammer with respect to such type guided by P, this user is added to the output list and removed from U. In [34] and [35], a similar strategy is used in the context of multiclass text classification, that is, the positive examples correctly classified are maintained after finishing each round of classification with respect to a P set. As a result, hPSD can gradually identify different types of spammers from the unlabeled set U.

V. Experimental Results

We now evaluate the performance of the proposed model for solving the spammer detection problem. We consider two scenarios: 1) to identify the injected attackers from movie datasets and 2) to detect the hidden spammers from real-world e-commerce data. Due to the available anomaly labels of injected attackers (scenario 1), we can compare our hPSD with various baseline methods and explore the impact of factors inside hPSD, both in terms of external measures. Whereas in scenario 2, we aim to demonstrate the effectiveness of our hPSD when it is applied to real e-commerce platforms.

TABLE II
Characteristics of Movie Datasets

Dataset	#User	#Movie	#Rating	Density
MovieLens_u2.base	943	1,682	80,000	5.04%
Netflix_s1	3,000	6,237	655,908	3.51%

TABLE III
Categorization of Shilling Attackers

Selection \ Rating	RFM	AFM
Random	Random Attack (Ran)	Average Attack (Avg)
Popular	Random-over-Popular (RoP)	Average-over-Popular (AoP)
Combination	Bandwagon (BanRan)	Bandwagon (BanAvg)

A. Detect Shilling Attackers in Movie Data

In this experiment, we use two movie datasets shown in Table II, where `u2.base` is a random split of MovieLens 100K dataset and `s1` was obtained by randomly sampling 3000 users but with movies rated less than 20 times deleted from the Netflix dataset.

1) Anomaly Injection: We take shilling attackers [11] as malicious users to be detected. In other words, we try to use the hPSD model to solve the shilling attack detection problem. The profile of a shilling attacker is in essence a rating record on various items, and it usually contains ratings on three types of items: 1) target items; 2) filler items; and 3) nonvoted items. The injected attackers are assumed to be with the *push* intent, that is, assigning the highest score to target items. Filler items can make a shilling profile look normal, yet exert profound impacts on other users.

In [27] and [36], the shilling attackers are summarized into several types according to different methods for selecting filler items and assigning ratings to them. There are three ways to select filler items: 1) random selection; 2) selection over popular items; and 3) the combination of them. The combination is to choose a set of popular items and give them the maximum allowed ratings, which is also called the bandwagon attack [36]. There are two rating assignment models for filler items: 1) random-filler model (RFM) and 2) average-filler model (AFM). So the shilling attackers can finally be categorized into six types, as shown in Table III. In this case, we inject about 10% hybrid shilling profiles into `u2.base` and `s1`, and assume original users in both datasets are normal. In particular, 120 and 300 attackers shared equally among six types are injected into `u2.base` and `s1`, respectively.

2) Feature Construction: In [17]–[19], a number of features are defined for distinguishing shilling attackers against normal users. However, some of them are disjointed. For example, WDMA and WDA [19] are the same as RDMA [17] except on the normalization method. We construct ten features for the shilling attack detection problem. Among them, seven features have been used several times in the previous work, including Entropy, DegSim, LengthVar, RDMA, FMTD, GFMV, and

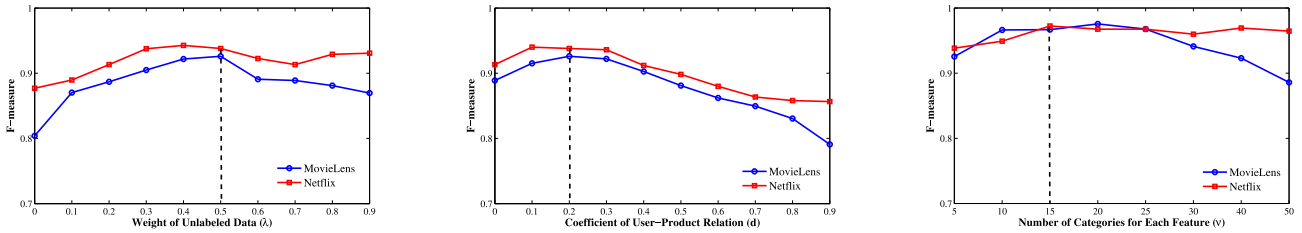


Fig. 2. Impact of parameters on the hPSD model. (a) λ. (b) d. (c) ν.

TMF. Furthermore, we define three new features: 1) popularity rank (PopRank); 2) average distance with other users (DistAvg); and 3) category entropy (CatEnt). In particular, PopRank is designed for filling popular items, which is an effective way to construct powerful attacks [26]. It measures the popularity over items rated by a user, and is defined as $\sum_{I_j \in R_i.} |R_{.j}|/|R_{i.}|$.

As summarized in [28], there are two ways for shilling attackers to maximize their predicted values. First, to construct a profile that is highly correlated to a fraction of the users (e.g., k nearest neighbors) and therefore affect them significantly. This rule can be reflected by the feature DegSim. Second, to construct a profile that is moderately correlated to a large number of users in order to affect them. In allusion to this rule, we define DistAvg as $(\sum_{j=1}^{n} 1 - \text{PCC}_{ij})/n$, where PCC_{ij} is the Pearson correlation coefficient between user u_i and u_j.

A basic assumption behind the definition of CatEnt is that a normal user is likely to have fixed interests on movies or products within limited categories, while an attacker selects filler items at random or by a common criteria, such as the popular tag, which may result in items scattered among many categories. Both MovieLens and Netflix have provided the category classification for every movie. We exploit this category classification to define the new feature CatEnt as $-\sum_{g=1}^{G} S_{ig}/S \log_2 S_{ig}/S$, where S_{ig} is the number of u_i's rated movies within the category g ($1 \le g \le G$) and $S = \sum_{g=1}^{G} S_{ig}$. It is possible that an enthusiast also has a wide range of interests, which implies none feature can precisely differentiate spammers from normal users.

3) Baselines and Evaluation Metrics: Three supervised classification models, that is, C4.5, SVM, and Naïve Bayes (NB) are first selected. The existing detectors [19], based on supervised classification, assume the continuous feature satisfies a normal distribution. However, with feature discretization, each feature satisfies the multinomial distribution which leads to different detectors. In the experiments, C4.5*, SVM*, and NB* denote the supervised detectors with feature discretization, which differ from C4.5, SVM, and NB based on the continuous feature modeling. To construct the training set, we sample 50 and 150 normal users from the rest of the dataset other than `u2.base` and `s1`. We additionally inject the same number of attackers into the training set which are equally split between random and average attackers, that is, $|L| \approx 10\%|U|$. Six supervised detectors are implemented by WEKA with above-mentioned ten features.

We further add two unsupervised detection methods, PCA [26] and MDS [27] as baselines, which are implemented in MATLAB. The input of both the unsupervised methods is the user-item rating matrix. Two parameters are the key to success of these detectors, that is, the number of identified attackers r of PCA and the number of clusters K of MDS. To set r, we assume PCA knows the number of injected attackers, though impossible in practice. Since K of MDS is tunable, the best results are reported. In total, we use eight shilling attack detectors as the baseline methods in comparison to our hPSD. Then, we inject the average and random attackers successively as P set, and let $|P| = 50$ and 150 for MovieLens and Netflix, respectively.

Since the ground-truth is known, we adopt the standard metrics, such as recall (R), precision (P), and F-measure (F). Specifically, all metrics are computed on the spammer class

$$R = \frac{\text{TP}}{\text{TP} + \text{FN}}, \quad P = \frac{\text{TP}}{\text{TP} + \text{FP}}, \quad F = \frac{2\text{PR}}{P + R} \tag{17}$$

where TP is the number of truly identified attackers, FN is the number of missed attackers, and FP is the number of wrongly identified attackers.

4) Parameter Analysis: In the first experiment, we study the impact of the parameters on the performance of our model. There are three parameters within hPSD: 1) the weight of unlabeled datasets λ; 2) the weight of user–product relation d; and 3) the number of categories for every feature ν. Fig. 2 shows the classification performance in terms of F-measure, where the filler size (FS) is set to 10% and 5% for MovieLens and Netflix, respectively. As can be seen, when λ approximately takes the median, the maximum F-measure values are obtained. It validates the important role of unlabeled datasets. However, as the increase of d, the performance continues reducing. This indicates that the impact of user–product relation is not suitable to be very large. When $\nu \in [15,\ 20]$, the F-measure achieves its maximum value on both datasets. As a result, in the following experiments, unless stated otherwise, we simply set $\lambda = 0.5$, $d = 0.2$, and $\nu = 15$.

5) Overall Comparison: In the second experiment, we compare our hPSD with eight baseline methods in terms of the quality of attacker classification. Table IV reports the effectiveness of different detection methods by varying the FS, that is, the ratio of filler items to all items. The lowest FS values of two datasets are approximately equal to the average length of user rating. We can obtain the following two observations from the results. First, hPSD shows the superior

TABLE IV
PERFORMANCE COMPARISON ON DETECTING HYBRID SHILLING ATTACKERS

Metric	Detector	MovieLens				Netflix			
		5%	10%	15%	20%	2%	5%	10%	15%
R	hPSD	0.958	**0.975**	**1**	**1**	0.963	**0.997**	**1**	**1**
	hPSD_H	**0.975**	0.958	**1**	**1**	**0.983**	0.980	**1**	**1**
	hPSD_R	0.792	0.792	0.867	0.850	0.570	0.717	0.923	0.903
	NB*	0.750	0.733	0.825	**1**	0.677	0.820	0.833	0.863
	SVM*	0.833	0.833	0.975	**1**	0.457	0.663	0.833	0.833
	C4.5*	0.883	**0.958**	0.917	**1**	0.627	0.667	0.667	0.667
	NB	0.642	0.333	0.667	0.667	0.443	0.347	0.387	0.667
	SVM	0.517	0.333	0.650	0.658	0.357	0.577	0.623	0.660
	C4.5	0.450	0.333	0.625	0.685	0.347	0.583	0.337	0.657
	PCA	0.775	0.717	0.667	0.667	0.633	0.597	0.653	0.667
	MDS	0.245	0.508	0.500	0.567	0.263	0.317	0.497	0.333
P	hPSD	0.885	0.959	0.938	0.952	0.845	0.949	0.962	0.956
	hPSD_H	0.854	0.878	0.960	0.945	0.776	0.842	0.949	0.932
	hPSD_R	0.841	0.950	0.981	0.990	0.881	0.927	0.952	0.982
	NB*	0.900	0.889	0.934	0.952	0.927	0.942	0.996	0.996
	SVM*	0.943	0.926	0.936	0.952	0.890	0.884	0.984	0.984
	C4.5*	0.726	0.723	0.821	0.857	0.908	0.794	0.939	0.957
	NB	**1**	**1**	**1**	**1**	**1**	**1**	**1**	**1**
	SVM	0.954	**1**	**1**	**1**	0.877	0.951	0.969	0.971
	C4.5	0.915	0.976	0.987	0.891	0.867	0.931	0.971	0.956
	PCA	0.775	0.717	0.667	0.667	0.633	0.597	0.653	0.667
	MDS	0.365	0.772	0.800	0.883	0.587	0.856	0.877	0.926
F	hPSD	**0.920**	**0.967**	**0.968**	**0.976**	**0.900**	**0.972**	**0.980**	**0.977**
	hPSD_H	0.911	0.916	**0.980**	0.972	0.868	0.906	0.974	0.965
	hPSD_R	0.815	0.864	0.920	0.915	0.692	0.808	0.937	0.941
	NB*	0.818	0.804	0.876	0.976	0.782	0.877	0.907	0.925
	SVM*	0.885	0.877	0.955	0.976	0.604	0.758	0.903	0.903
	C4.5*	0.797	0.824	0.866	0.923	0.742	0.725	0.780	0.786
	NB	0.782	0.500	0.800	0.800	0.614	0.515	0.558	0.800
	SVM	0.670	0.500	0.788	0.794	0.507	0.718	0.759	0.786
	C4.5	0.603	0.497	0.765	0.774	0.495	0.717	0.500	0.779
	PCA	0.775	0.717	0.667	0.667	0.633	0.597	0.653	0.667
	MDS	0.293	0.613	0.615	0.690	0.364	0.462	0.634	0.490

Note: (1) Marking $\star$ denotes features are discretized and modeled by multinominal distribution.
(2) hPSD_H denotes to inject average and random attackers into P set simultaneously.
(3) hPSD_R denotes to extract RN set by randomly selecting users from U.

performance over the other methods. Meanwhile, the recall of hPSD always attains the highest position, indicating that nearly all the injected attackers have been identified by hPSD. By contrast, the supervised approaches only detect a small amount of attackers, which is indicated by higher P values but lower R values. Second, C4.5*, SVM*, and NB* are superior to C4.5, SVM, and NB, respectively. This largely owes to the proposed feature discretization method that can fuse the statistical distribution of each feature on unlabeled data. Third, the hPSD outperforms hPSD_H in most cases and always prevails over hPSD_R. So, the RN extraction and the iterative detection phases using two classifiers are the necessary settings for hPSD. In what follows, we will show further validation for the two important settings.

An hPSD "wrongly" classified a number of normal users as attackers. Hence, we are interested in whether the behavior of these wrongly identified attackers is abnormal. We select MovieLens with FS $= 10\%$ as a case study, where FP $= 4, 6$ as building P with average and random attackers, respectively. Thus, 77 suspected target movies are extracted, that is, at least one 5-star rating is given by a wrongly identified attacker. By plotting the average scores of 77 movies, as shown in Fig. 3, we find that the average rating is generally low. Also, roughly 73% movies are disliked by most users, for example, the average score is lower than 4. As the movie name is provided, we look up the reputation of the suspected target movies from other sources, and find most of them are unpopular. For example, the movie "Shadow Conspiracy" (ID: 984) only received 8.9% positive votes (i.e., 4- and 5-star ratings) in IMDb,[3] and

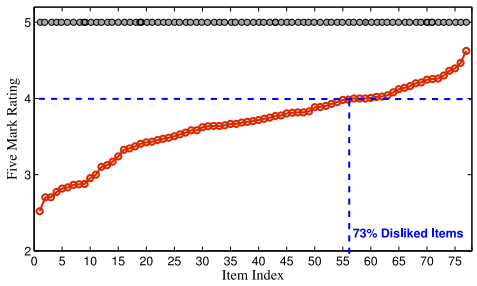
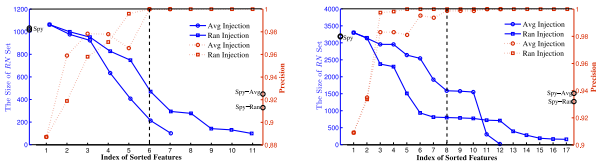


Fig. 3. Abnormal ratings of wrongly identified attackers.



(a) (b)

Fig. 4. Precision of the RN extraction. (a) `MovieLens`, FS $= 10\%$. (b) `Netflix`, FS $= 5\%$.

its average rating was 2-star explained as "poor" in empire-online.[4] The abnormal behavior that the wrongly identified attackers rate 5-star to unpopular movies implies that these users are likely to have a push intent.

6) Inside PU-Learning: Here, we investigate two important factors inside hPSD: 1) the quality of RN set and 2) the necessity of using two classifiers. Two cases are selected for validation: 1) MovieLens with FS $= 10\%$ and 2) Netflix with FS $= 5\%$.

Quality of RN: We validated the effectiveness of the RN set extraction heuristic in Section IV-B. For each feature, we record the number of truly negative instances in RN (denoted as |TRN|). As a result, the precision for measuring the quality of RN is defined as |TRN|/|RN|. Obviously, the initial precision is the ratio of normal users to all users, which is roughly 90%. The famous `Spy` algorithm [37] for RN extraction is adopted as baseline, where half of instances are randomly sampled from P as spy instances.

Fig. 4 reports the results on two movie datasets. Clearly, `Spy` performs poorly in extracting a big RN set containing many positive instances. This is because NB* equipped in `Spy` tends to excessively classify users as legitimate users. By contrast, our proposed method can flexibly adjust the scale of RN, and quickly filter positive instances (i.e., |TRN|/|RN| soars to 1). We further observe that as the |RN| values of MovieLens and Netflix are reduced to about 400 and 1000, respectively, no positive instance is included in RN. Since we only extract an RN set with a scale near to set P, the extracted RN set achieves an accuracy of 100% in our experiments.

Necessity of Using Two Classifiers: hPSD essentially trains multiple classifiers based on the different P, each of which is used to detect one type of attackers. Hence, we hope multiple

[3]http://www.imdb.com/title/tt0120107/?ref_=fn_al_tt_1

[4]http://www.empireonline.com/reviews/review.asp?FID=2447

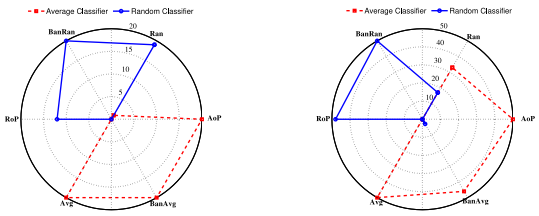


Fig. 5. Complementarity between average and random classifiers. (a) MovieLens, FS = 10%. (b) Netflix, FS = 5%.

TABLE V
CHARACTERISTICS OF AMAZON DATASET

Dataset	#User	#Product	#Rating	Density
Amazon	9,424	19,185	469,393	0.26%

classifiers are complementary. To demonstrate this, we investigate the impacts of the average and random classifiers on detecting hybrid shilling attackers.

Fig. 5 shows the number of truly identified attackers of six types in case of two classifiers. It clearly demonstrates that the average and random classifiers inside hPSD exhibit strong complementarity. The missed attackers are mainly from RoP, and the AoP attackers can only be captured by the average classifier. Meanwhile, the random classifier can identify a part of AFM attackers, but the average classifier is difficult to detect RFM attackers. These indicate that selecting popular items as filler items and assigning them with random ratings do bring the challenge to detectors.

B. Detect Hidden Attackers in Amazon Data

In this section, we apply hPSD to a large proprietary dataset for detecting hidden spam users. This dataset is provided by Amazon China[5] containing review records from September 2000 to December 2011. To get rid of niche products and cold-start users, we extract the products that were rated at least 15 times, and then pick out users who have rated over 20 times. Consequently, an experimental dataset is acquired with the basic information listed in Table V.

1) Attacker Types: In our experiments, we consider two types of attackers, that is, duplicate spammers and promoters, that are ubiquitous on real e-commerce platforms [38]. The duplicate spammers post a series of nearly identical reviews that usually have weak relationship with target products. Promoters are more sophisticated and always have the potential marketing goals. Many tricky promoters behave almost the same as legitimate users and can actively circumvent the detection from webmasters by making fake purchase record. However, reviewing the same products with other identified attackers might expose themselves to our hybrid learning method.

2) Feature Construction: Every record of Amazon dataset consists of the following attributes: 1) User_ID; 2) Product_ID (ASIN); 3) review title; 4) star rating, and 5) posting date. Via the lens of rich information, we build eight features as attack behavior signals. Among them, five features are reformed based on opinion spam behavior indicators [9], [10]: 1) the difference of first and last rating dates; 2) the number of days when posting ratings; 3) the maximal number of ratings posted in one day; 4) the average number of ratings per day; and 5) the entropy (i.e., variance) of the number of ratings per day. In addition, PopRank and CatEnt defined in Section V-A2 will also be used here. Finally, we define a novel indicator to measure the similarity among the products' ASIN rated by a user as $\sum_{I_p,I_q\in R_i.} \mathrm{LCS}_{p,q}/|\mathrm{ASIN}|$, where $\mathrm{LCS}_{p,q}$ represents the longest common subsequence of two ASIN strings, and ASIN in Amazon is a character string with fixed-length 10. Although it is claimed that ASIN is randomly assigned to a product [39], we still observe the deceptive spammer tends to rate a series of products with very similar ASIN. It can probably be boiled down to two reasons: 1) robots might sequentially scan ASIN to produce a great deal of identical ad words and 2) congeneric products with the close release time might have similar random ASIN strings.

3) Spotting Duplicate Spammers: Due to available review titles, it is very easy for us to manually label the set P of duplicate spammers. We first select 82 users who post a large number of identical yet long review titles, and create six copies of each labeled user to alleviate the imbalance between two classes. Therefore, the P set contains 492 instances and the corresponding RN set contains 484 instances.

We then run hPSD to obtain 977 users who are classified in the spam class. Together with 82 manually labeled attackers, we finally identify 1059 (11.2%) duplicate spammers. After scanning the review titles of each users and identifying some identical review titles, a majority of users (892 users) are found posting at least three identical review titles of which the length exceeds six characters, and can obviously be identified as the duplicate spammers. The behavior of these users fit with the definition of a duplicate spammer. However, 167 unexpected users are also identified as the duplicated spammers. We sampled several users and carefully examined their homepages in Amazon. Two following observations are noteworthy. First, some disquised duplicate spammers are discoverable. They can hardly be identified as duplicate spammers due to the varying yet short review titles they post. However, the fact that they usually recommend a series of similar products discloses themselves as spammers. A user with nickname "zhanglong1112"[6] is a typical spammer on electronic products, such as cameras, mike, earphones, etc. Second, some normal users prefer writing similar review titles and posting lots of reviews in the same day. By carefully checking their homepages, we can judge they are not attackers according the frequent purchase and the useful votes. For instance, "jjwinwin119"[7] and "hly71801"[8] are typical normal users who operate in this manner.

4) Spotting Promoters: To label promoters manually is more arduous than to label duplicate spammers. A strict

[5]http://www.amazon.cn

[6]http://www.amazon.cn/gp/cdp/member-reviews/A3RH0RXLEOIT34?ie=UTF8&display=public&page=2&sort_by=MostRecentReview

[7]http://www.amazon.cn/gp/cdp/member-reviews/A1BXAY6Y4UL8Q8?ie=UTF8&display=public&page=2&sort_by=MostRecentReview

[8]http://www.amazon.cn/gp/cdp/member-reviews/A1M2UBKJU21NVA/ref=cm_cr_pr_auth_rev?ie=UTF8&sort_by=MostRecentReview

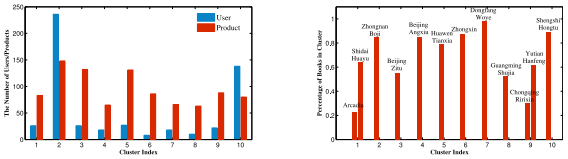


Fig. 6. Characteristics of sampled clusters. (a) Scale of users/products. (b) Dominant publishing company.

rule is presented to label a user as a spammer: over 80% of 5-star reviews of this user are given to products having strong semantic relationships with each other. We then empirically judge the strong semantic relationship, such as books/audio/video with the same issuer, cosmetics/cellphones in the same brand, etc. With such rule, we manually labeled 50 promoters and then constructed the P set of 300 instances by creating six copies. By removing duplicate spammers identified early, hPSD extracted an RN set including 283 instances and classified 979 users to the spam class. Consequently, 1029 (10.9%) promoters were identified in total.

Abnormal Behavior of Identified Spammers: A bipartite network with 1029 user nodes and 4053 product nodes is built to represent the rating behavior between the users and products. The Graclus[9] tool is utilized to partition the bipartite network into 30 clusters, with the normalized cut as the objective function. For each cluster, we count the number of products falling in every category (e.g., books, electronics, etc.) and thus find that 21 clusters primarily contain books, 7 clusters cover electronic products, and the remaining 2 clusters are about cosmetics. So, it seems that the book cluster is the most severely attacked category by the promoters. This is probably because Amazon mainly sells books. However, we tentatively interpret this as the driving sales for books has a much stronger economic incentive as compared with other products. Namely, a publishing company generally releases vast books on Amazon and thus is likely to hire promoters to promote its own books.

To verify our interpretation, we selected 10 out of 21 book clusters, and searched the publishing company of each offending book on Amazon. Fig. 6(a) shows the number of users and books in each cluster, and Fig. 6(b) depicts the percentage of books released by the same publishing company. It is striking to see that so many users in each cluster assigned nearly all reviews to books published by the same company. We, therefore, believe the identified attackers do contain a high proportion of promoters. In turn, many publishing companies are under suspicion of employing promoters on Amazon.

Impact of Hybrid Learning: In this experiment, we validate the effectiveness of hybrid learning. We run hPSD with and without joint learning based on the same P set. The detection results of hPSD with and without hybrid learning are summarized as a two-way table (Table VI). A kappa statistic of 54.8% indicates that two methods have achieved a certain consensus in attack identification. Specifically, hPSD aided by hybrid learning additionally identifies 568 promoters but

[9]http://www.cs.utexas.edu/users/dml/Software/graclus.html

TABLE VI
STATISTICS OF WITH AND WITHOUT HYBRID LEARNING

		Without Hybrid Learning		Σ_{row}
		Normal	Promoter	
With Hybrid Learning	Normal	8411	34	8445
	Promoter	**568**	411	*979*
Σ_{col}		8979	*445*	$\kappa = 0.548$

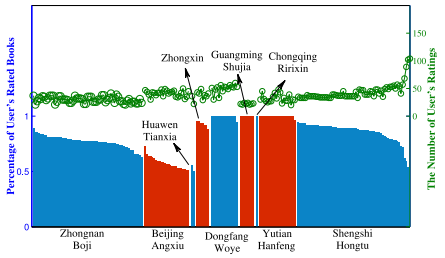


Fig. 7. Abnormal behavior of promoters identified by joint learning.

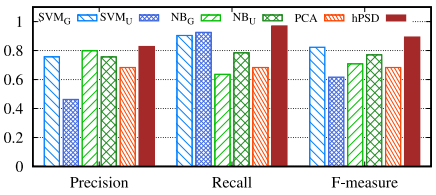


Fig. 8. Performance of compared methods on Amazon.

only misses 34 promoters, compared with hPSD without using hybrid learning. Hence, we are interested in whether these 568 newly identified users are truly abnormal. In the above experiment shown in Fig. 6, we have manually searched 750 books published by 12 deceptive companies. As a result, we are able to compute the percentage of books that are released by each deceptive issuer and rated by these newly identified promoters. We filter the user with the percentage less than 50% and finally obtain 204 users concentrating on 9 issuers. Fig. 7 reports the results. As can be seen, the users generally write reviews for about 50 books, among which a majority of books are released by the same deceptive issuer. From this perspective, these newly identified promoters tend to be abnormal, which in turn verifies the effectiveness of hybrid learning. As we examine the homepages of the above 204 promoters, we find that most users are not filtered by Amazon, while some of their reviews have been deleted. This also provides the evidence of their spam behavior. Meanwhile, most suspicious promoters remaining in Amazon have a few purchase records, which generally makes them to look like normal users. For instance, according to the homepage of "xiaokeshuichong,"[10] all reviews (e.g., 12 reviews in total) are posted to glorify books released by "Guangmingshujia," and two purchase records have been confirmed by Amazon.

Comparison With Other Methods: Among 1029 promoters identified by hPSD, we manually label 887 malicious users. This human labeling is *conservative*, that is, the labeled promoters are definitely spammers and not vice versa. As a result, we obtain an experimental dataset with 887 promoters and

[10]http://www.amazon.cn/gp/cdp/member-reviews/A2EAUSOFYDQO1Z/ref=cm_cr_pr_auth_rev?ie=UTF8&sort_by=MostRecentReview

7478 legitimate users. We consider two types of features: 1) graph features, as suggested in [12], include PageRank scores, degree, k-core, graph coloring, connected components, and triangle count and 2) user features, as defined in Section V-B2. Then, we select two classifiers SVM and NB, and combine them with both graph and user features to obtain four supervised detection methods: 1) SVM_G; 2) SVM_U; 3) NB_G; and 4) NB_U. SVM_G denotes the SVM classifier with graph features, which is analogical to other three notations. In addition, we select one unsupervised detector PCA [26] for comparison. We conduct tenfold cross validation for our hPSD and four supervised detection methods, while the unsupervised PCA is run on the whole data only once. Fig. 8 shows the comparison results in terms of P, R, and F. As can be seen, hPSD achieves the best performances among six compared methods, which is followed in descending order by SVM_G, NB_U, NB_G, PCA, and SVM_U. Moreover, SVM_G performs fairly good (i.e., only 8% less than hPSD on F-measure), which implies the importance of the relation in spam detection.

VI. Conclusion

This paper proposes a novel and principled hybrid learning model to combine both the feature space and user–product relations for spammer detection. In particular, the proposed hPSD model employs PU-learning to iteratively detect multiple types of spammers. It also enables semisupervised learning that makes full use of both labeled and unlabeled datasets to construct the classifier. Experimental results on movie data with shilling injection show that our hPSD outperforms several state-of-the-art baseline methods. Consequently, the hPSD model is utilized for the detection of hidden spammers in a real-life Amazon dataset. A set of spammers and their underlying employers (e.g., book publishers) are successfully discovered and validated. These demonstrate the effectiveness and practical value of hPSD in real-life applications. To make the applicability of hPSD more widespread, in the future, we will enhance the hybrid learning model to incorporate more heterogeneous data, such as topics of review text.

References

[1] C. Forman, A. Ghose, and B. Wiesenfeld, "Examining the relationship between reviews and sales: The role of reviewer identity disclosure in electronic markets," *Inf. Syst. Res.*, vol. 19, no. 3, pp. 291–313, 2008.

[2] F. Zhu and X. Zhang, "Impact of online consumer reviews on sales: The moderating role of product and consumer characteristics," *J. Market.*, vol. 74, no. 2, pp. 133–148, 2010.

[3] T.-M. Choi, H. K. Chan, and X. Yue, "Recent development in big data analytics for business operations and risk management," *IEEE Trans. Cybern.*, vol. 47, no. 1, pp. 81–92, Jan. 2017.

[4] M. Ott, C. Cardie, and J. Hancock, "Estimating the prevalence of deception in online review communities," in *Proc. 21st Int. Conf. World Wide Web*, 2012, pp. 201–210.

[5] E.-P. Lim, V.-A. Nguyen, N. Jindal, B. Liu, and H. W. Lauw, "Detecting product review spammers using rating behaviors," in *Proc. 19th ACM Int. Conf. Inf. Knowl. Manag.*, 2010, pp. 939–948.

[6] G. Wang, S. Xie, B. Liu, and P. S. Yu, "Review graph based online store review spammer detection," in *Proc. 11th IEEE Int. Conf. Data Min. (ICDM)*, 2011, pp. 1242–1247.

[7] A. Fayazi, K. Lee, J. Caverlee, and A. Squicciarini, "Uncovering crowdsourced manipulation of online reviews," in *Proc. 38th Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval*, 2015, pp. 233–242.

[8] N. Jindal and B. Liu, "Review spam detection," in *Proc. 16th Int. Conf. World Wide Web*, 2007, pp. 1189–1190.

[9] A. Mukherjee, B. Liu, and N. Glance, "Spotting fake reviewer groups in consumer reviews," in *Proc. 21st Int. Conf. World Wide Web*, 2012, pp. 191–200.

[10] A. Mukherjee *et al.*, "Spotting opinion spammers using behavioral footprints," in *Proc. 19th ACM SIGKDD Int. Conf. Knowl. Disc. Data Min.*, 2013, pp. 632–640.

[11] S. K. Lam and J. Riedl, "Shilling recommender systems for fun and profit," in *Proc. WWW*, 2004, pp. 393–402.

[12] S. Fakhraei, J. Foulds, M. Shashanka, and L. Getoor, "Collective spammer detection in evolving multi-relational social networks," in *Proc. 21st ACM SIGKDD Int. Conf. Knowl. Disc. Data Min.*, 2015, pp. 1769–1778.

[13] N. Jindal and B. Liu, "Opinion spam and analysis," in *Proc. Int. Conf. Web Search Data Min.*, 2008, pp. 219–230.

[14] G. P. C. Fung, J. X. Yu, H. Lu, and P. S. Yu, "Text classification without negative examples revisit," *IEEE Trans. Knowl. Data Eng.*, vol. 18, no. 1, pp. 6–20, Jan. 2006.

[15] J. Wu, S. Pan, X. Zhu, C. Zhang, and X. Wu, "Positive and unlabeled multi-graph learning," *IEEE Trans. Cybern.*, vol. 47, no. 4, pp. 818–829, Apr. 2017.

[16] N. Jindal and B. Liu, "Analyzing and detecting review spam," in *Proc. 7th IEEE Int. Conf. Data Min. (ICDM)*, 2007, pp. 547–552.

[17] R. Burke and B. Mobasher, C. Williams, and R. Bhaumik, "Classification features for attack detection in collaborative recommender systems," in *Proc. KDD*, 2006, pp. 542–547.

[18] P.-A. Chirita, W. Nejdl, and C. Zamfir, "Preventing shilling attacks in online recommender systems," in *Proc. WIDM*, 2005, pp. 67–74.

[19] C. Williams, "Profile injection attack detection for securing collaborative recommender systems," School Comput. Sci. Telecommun. Inf. Syst., DePaul Univ., Chicago, IL, USA, Rep. TR06-014, pp. 1–47, 2006.

[20] X. Wu, W. Fan, J. Gao, Z. Feng, and Y. Yu, "Detecting marionette microblog users for improved information credibility," *J. Comput. Sci. Technol.*, vol. 30, no. 5, pp. 1082–1096, 2015.

[21] Z. Wu, J. Wu, J. Cao, and D. Tao, "HySAD: A semi-supervised hybrid shilling attack detector for trustworthy product recommendation," in *Proc. 18th ACM SIGKDD Int. Conf. Knowl. Disc. Data Min.*, 2012, pp. 985–993.

[22] F. Li, M. Huang, Y. Yang, and X. Zhu, "Learning to identify review spam," in *Proc. IJCAI Int. Joint Conf. Artif. Intell.*, vol. 22, 2011, pp. 2488–2493.

[23] M. Ott, Y. Choi, C. Cardie, and J. T. Hancock, "Finding deceptive opinion spam by any stretch of the imagination," in *Proc. 49th Annu. Meeting Assoc. Comput. Linguist. Human Lang. Technol.*, vol. 1, 2011, pp. 309–319.

[24] R. Yu, X. He, and Y. Liu, "GLAD: Group anomaly detection in social media analysis," in *Proc. 20th ACM SIGKDD Int. Conf. Knowl. Disc. Data Min.*, 2014, pp. 372–381.

[25] Y. Wang, Z. Wu, Z. Bu, J. Cao, and D. Yang, "Discovering shilling groups in a real e-commerce platform," *Online Inf. Rev.*, vol. 40, no. 1, pp. 62–78, 2016.

[26] B. Mehta, T. Hofmann, and P. Fankhauser, "Lies and propaganda: Detecting spam users in collaborative filtering," in *Proc. 12th Int. Conf. Intell. User Interfaces (IUI)*, 2007, pp. 14–21.

[27] J.-S. Lee and D. Zhu, "Shilling attack detection—A new approach for a trustworthy recommender system," *INFORMS J. Comput.*, vol. 24, no. 1, pp. 117–131, 2012.

[28] B. Mehta and W. Nejdl, "Unsupervised strategies for shilling detection and robust collaborative filtering," *User Model. User Adapt. Interact.*, vol. 19, nos. 1–2, pp. 65–97, 2009.

[29] H. Liu *et al.*, "How many zombies around you?" in *Proc. IEEE 13th Int. Conf. Data Min. (ICDM)*, 2013, pp. 1133–1138.

[30] U. M. Fayyad and K. B. Irani, "Multi-interval discretization of continuous-valued attributes for classification learning," in *Proc. IJCAI*, 1993, pp. 1022–1029.

[31] J. Yuan *et al.*, "T-drive: Driving directions based on taxi trajectories," in *Proc. 18th SIGSPATIAL Int. Conf. Adv. Geograph. Inf. Syst. (GIS)*, 2010, pp. 99–108.

[32] X. Li, S. Y. Philip, B. Liu, and S.-K. Ng, "Positive unlabeled learning for data stream classification," in *Proc. 9th SIAM Int. Conf. Data Min.*, 2009, pp. 259–270.

[33] V. Plagianakos and G. Magoulas, "Stochastic gradient descent," in *Advances in Convex Analysis and Global Optimization: Honoring the Memory of C. Caratheodory (1873–1950)*, vol. 54. Dordrecht, The Netherlands: Kluwer, 2013, p. 433.

[34] B. Liu, W. S. Lee, P. S. Yu, and X. Li, "Partially supervised classification of text documents," in *Proc. 19th Int. Conf. Mach. Learn. (ICML)*, 2002, pp. 387–394.
[35] B. Liu, Y. Dai, X. Li, W. S. Lee, and P. S. Yu, "Building text classifiers using positive and unlabeled examples," in *Proc. 3rd IEEE Int. Conf. Data Min. (ICDM)*, 2003, pp. 179–186.
[36] B. Mobasher, R. Burke, and J. J. Sandvig, "Model-based collaborative filtering as a defense against profile injection attacks," in *Proc. AAAI*, Boston, MA, USA, 2006, pp. 1388–1393.
[37] B. Liu, *Web Data Mining: Exploring HyperLinks, Contents, and Usage Data*. Heidelberg, Germany: Springer, 2007.
[38] K. Lee, J. Caverlee, and S. Webb, "Uncovering social spammers: Social honeypots + machine learning," in *Proc. 33rd Int. ACM SIGIR Conf. Res. Develop. Inf. Retrieval*, 2010, pp. 435–442.
[39] P. Bausch, *Amazon Hacks*. Sebastopol, CA, USA: O'Reilly Media, 2003.

Zhiang Wu (S'14–M'17) received the Ph.D. degree in computer science from Southeast University, Nanjing, China, in 2009.

He is currently a Full Professor with the School of Information Engineering, Nanjing University of Finance and Economics, Nanjing, where he is also the Director of the Jiangsu Provincial Key Laboratory of E-Business. His current research interests include distributed computing, data mining, e-commerce intelligence, and social network analysis. He has published over 30 refereed journal and conference papers in the above areas.

Dr. Wu is a member of the ACM and a Senior Member of CCF.

Jie Cao received the Ph.D. degree from Southeast University, Nanjing, China, in 2002.

He is currently a Chief Professor and the Dean of the School of Information Engineering, Nanjing University of Finance and Economics, Nanjing. He is also a Ph.D. Advisor with the Nanjing University of Science and Technology, Nanjing. His current research interests include data mining, big data, and e-commerce intelligence.

Dr. Cao has been selected in the Program for New Century Excellent Talents in University and Awarded with Young and Mid-Aged Expert with Outstanding Contribution in Jiangsu Province. He is a member of the ACM, CCF, and IEEE Computer Society.

Yaqiong Wang received the B.S. and M.S. degrees in information systems from the School of Economics and Management, Beihang University, Beijing, China. She is currently pursuing the Ph.D. degree in information systems with the Department of Information and Decision Sciences, Carlson School of Management, University of Minnesota, Minneapolis, MN, USA.

Her research program explores how to exploit data analytics and information technologies to identify user behavior patterns and enhance business performance. Her current research interests include personalization technologies and recommender systems, social commerce, and evaluation of predictive models.

Youquan Wang received the Ph.D. degree in computer science from the Nanjing University of Science and Technology, Nanjing, China, in 2017.

He is currently a Lecturer with the Jiangsu Provincial Key Laboratory of E-Business, Nanjing University of Finance and Economics, Nanjing. His current research interests include social network analysis and data mining.

Dr. Wang is a member of CCF and ACM.

Lu Zhang received the Ph.D. degree in computer science from Southeast University, Nanjing, China, in 2012.

He is currently a Lecturer with the Jiangsu Provincial Key Laboratory of E-Business, Nanjing University of Finance and Economics, Nanjing. His current research interests include distributed computing and network security.

Dr. Zhang is a member of CCF.

Junjie Wu received the B.E. degree in civil engineering and the Ph.D. degree in management science and engineering from Tsinghua University, Beijing, China, in 2002 and 2008, respectively.

He is currently a Full Professor with the Information Systems Department, Beihang University, Beijing, China. He is also the Director of the Research Center for Data Intelligence, and the Vice Director of the Beijing Key Laboratory of Emergency Support Simulation Technologies for City Operations. His current research interests include data mining, with a special interest in social computing, urban computing, and financial computing.

Mr. Wu was a recipient of the various national academic awards in China, including the NSFC Distinguished Young Scientist, the MOE Changjiang Young Scholars, and the National Excellent Doctoral Dissertation.