Three Birds With One Stone: User Intention Understanding and Influential Neighbor Disclosure for Injection Attack Detection

Zhihai Yang, Qindong Sun, and Zhaoli Liu

***Abstract*— Recommender system, as a data-driven way to help customers locate products that match their interests, is increasingly critical for providing competitive customer suggestions in many web services. However, recommender systems are highly vulnerable to malicious injection attacks due to their fundamental vulnerabilities and openness. With the endless emergence of new attacks, how to provide a feasible way for defending different malicious threats against online recommendations is still an under-explored issue. In this paper, we explore a new way to defend malicious injection attacks through user intention understanding and influential neighbour disclosure. Specifically, we propose a detection approach, termed *TBOS* (Three Birds with One Stone), to deal with different malicious threats. In *TBOS*, we first develop the discrimination of attack target by combining global influence evaluation and risk attitude estimation of users. In order to make *TBOS* controllable, second, we propose to incorporate an optimal denoising mechanism to remove disturbed information before detection. To enhance the representativeness and predictability of detection model, finally, we propose to leverage a behavioral label propagation mechanism based on constructed label space for the determination of malicious injection behaviors. Extensive experiments on both synthetic and real data demonstrate that *TBOS* outperforms all baselines in different cases. Particularly, the detection performance of *TBOS* can achieve an improvement of 6.08% FAR (false alarm rate) for optimal-injection attacks, an improvement of 3.83% FAR in average for co-visitation injection attacks, as well as an improvement of 2.3% for profile injection attacks over benchmarks in terms of FAR while keeping the highest DR (detection rate). Additional experiments on real-world data show that *TBOS* brings an improvement with the advantage of 6.5% FAR in average compared with baselines.**

***Index Terms*— Injection attack, attack detection, behavior representation, performance analysis.**

Manuscript received May 12, 2021; revised December 19, 2021; accepted January 16, 2022. Date of publication January 26, 2022; date of current version February 8, 2022. This work was supported in part by the National Natural Science Foundation of China under Grant 62172331 and Grant 62102310, in part by the Youth Innovation Team Construction of Shaanxi Provincial Department of Education under Grant 21JP081, in part by the China Postdoctoral Science Foundation under Grant 2020M683689XB, in part by the Natural Science Founds of Shaanxi under Grant 2021JQ-486, and in part by the Youth Innovation Team of Shaanxi Universities under Grant 2019-38. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Issa Traore. *(Corresponding author: Qindong Sun.)*

Zhihai Yang and Zhaoli Liu are with the School of Computer Science and Engineering, Xi'an University of Technology, Xi'an 710048, China, and also with the Shaanxi Key Laboratory for Network Computing and Security Technology, Xi'an 710048, China (e-mail: zhyang_xjtu@sina.com).

Qindong Sun is with the School of Cyber Science and Engineering, Xi'an Jiaotong University, Xi'an 710049, China, and also with the Shaanxi Key Laboratory for Network Computing and Security Technology, Xi'an 710048, China (e-mail: qdongsun@xjtu.edu.cn).

Digital Object Identifier 10.1109/TIFS.2022.3146769

I. Introduction

ONLINE recommender systems are designed to help deal with the information explosion problem and have become a crucial component of various e-commerce services such as Amazon and TripAdvisor, which recommend a user items (e.g., books on Amazon and hotels on TripAdvisor) that match the user's preference [1]. In particular, Collaborative Filtering Recommendations (CFRs), such as neighborhood-based, graph-based, association-rule-based CFRs, have been investigated during the past two decades [1], [2]. An underlying assumption of CFRs is that two users will share common interests in the future if they have expressed similar interests in the past. Nevertheless, CFRs are highly vulnerable to profile injection attacks (a.k.a., *shilling* attacks) [3], fake co-visitation injection attacks [4], poisoning attacks [2], etc., due to their openness and fundamental vulnerabilities [5], [6]. According to reports, *a prediction shift of around 1.5 points on a five-point scale can be manipulated under a 3% fake profile attacks* [7]. Attackers either inject a sufficient number of well-designed fake profiles into a recommender system and empirically rate higher scores (termed *push* or *promotion* attacks [2], [8]) or lower scores (called *nuke* or *demotion* attacks [2], [8]) towards targeted items, or inject fake co-visitations into the system to spoof CFRs [4] to manipulate recommendations or reduce the quality of recommendation as the attackers desire, which causes great damage to the public, also shaking the confidence of both customers and businesses in the virtual market. Therefore, the demand for protecting users' personal benefits is becoming more pressing.

A. Challenge and Motivation

To defend such threats, many past research methodologies hold much value [8], [9], [10]. Although these recent methods are more practical than earlier ones, several outstanding challenges still remain to be addressed as follows, (1) the conventional Target Item Analysis (TIA) [10], [11] which leverages the number of users who have rated an item to determine target items, is uncontrollable and inelastic especially facing with small-scale attacks; (2) an additional difficulty is that simply eliminating *cold* users or items [7] is insufficient to reduce the impact of disturbed data on detection performance. In reality, it is difficult to balance the dimensionality reduction of sparse rating matrix and the retention of crucial information (e.g., well-designed fake profiles we need to detect); and (3) another long-standing but unresolved issue is that the uncertainty of labels or the identification of labels limits the improvement of detection performance. How to make full use of definite and indefinite label data is still an open issue.

These challenges or difficulties inspire us: (1) intuitively, detecting injection attacks without target analysis is like *dredging for a needle in the sea*. It is intractable to discriminate suspicious targets by purely relying on counting the number of users or items. Therefore, how to design a divide-and-conquer mechanism for accurate detection is worth investigating; and (2) further enhancing the representation of rating or visitation behavior extracted from raw data is undesirable. From previous studies, the advantage of graph mining based detection methods is comparatively significant. To this end, both side information and divide-and-conquer strategies are needed.

B. Solution and Contribution

In the light of these challenges or opportunities, this paper investigates a new detection approach termed *TBOS* (i.e., Three Birds with One Stone, where *three birds* denote different attacks and *one stone* means our proposed approach), to identify malicious users or items, which expects to deal with different attacks as many as possible. To enhance the Discrimination of Attack Target (DAT), first of all, we explore a new mechanism for determining suspicious items through a weighted combination of global influence evaluation and risk attitude estimation while retaining the advantage of TIA (see Section IV-B). To adaptively and automatically filter out disturbed data before detection, second, we incorporate a denoising strategy based on influence maximization for obtaining an optimal subset (i.e., remove as many genuine users as possible and retain all concerned attackers) from original data (see Section IV-C). By considering the influence of both definite and indefinite labels on accurate detection, we construct a label space for both definite and indefinite (or termed *gray*) behaviors, and assign a valid label for both of behaviors through label propagation (see Section IV-D.2). Finally, we develop a label-prediction-based detection method to identify fake users or items based on DAT and the optimal subset. Extensive experiments on both synthetic and real-world data demonstrate the effectiveness of the proposed approach.

In summary, this paper makes the following contributions:

1) We develop an evaluation method for the determination of suspicious targets, and propose to combine the global influence and risk attitude of users (or items) for revealing the indirect influence and inherent intention of users.
2) We propose to incorporate an optimal subset selection mechanism based on influence maximization for adaptively filtering out disturbed data before detection.
3) We propose to exploit both definite and indefinite label data to enhance the representation of rating or visitation behavior, and propose a new detection approach to deal with profile injection attacks, optimal-injection attacks, and co-visitation injection attacks. Comparative experiments on synthetic and real data demonstrate that the proposed *TBOS* outperforms all baselines almost.

II. Related Work

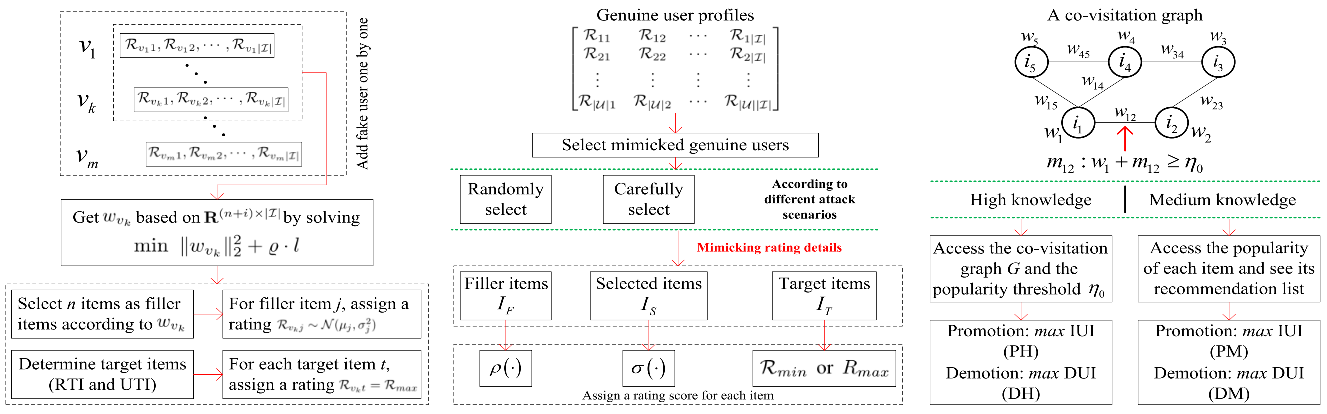
Due to the endless emergence of malicious threats against recommender systems, investigating detection approaches for defending these threats has attracted much attention from both academic and industry in the past two decades. Previous efforts provide promising results in terms of the determination of target items, the elimination of noise data, and the representation of rating (or co-visitation) behaviors. In what follows, we introduce related researches focused on the above aspects.

To achieve the purpose of attack, attackers generally focus on a target through group behaviors. For instance, attackers consistently push or nuke a target item into the recommendation list of concerned users in both profile injection attacks [3] and optimal injection attacks [2]. In co-visitation injection attacks [4], attackers elaborately inject fake co-visitations between *anchor* items and target items in order to make the target items appear in the recommendation list of anchor items. Therefore, spotting target items is a crucial task for attack detection. Previous efforts provide promising results and feasible directions for determining target items. Zhou *et al.* [11] first focused on TIA. They investigated the distribution of items in order to capture suspicious target items. After that, Yang *et al.* [10] investigated the advantage and disadvantage of TIA for anomaly detection facing with diverse injection attacks and proposed a step-by-step mechanism to make up for the deficiency of TIA.

Despite promising results based on TIA, the existence of disturbed information (e.g., unpopular items and inactive users [10]) also affects the discrimination of abnormal behavior, which results in high false alarm rates. To address this issue, investigating step-by-step detection frameworks to filter out *noise* data (compared with attack profiles, authentic profiles can be considered as noise data) has attracted much attention. First, Hurley *et al.* [12] proposed a PCA-based detection approach to eliminate a part of disturbed user profiles. Then, Zhou *et al.* [11] also developed a two-step detection method to identify shilling attacks. After that, Zhang *et al.* [7] presented a unified framework to detect spam users. Recently, Yang *et al.* [9] investigated a stepwise detection approach to spot spam users in both synthetic data and real-world data.

In addition to target item analysis and the elimination of noise data, designing a classification or prediction model for identifying concerned attackers is desirable. Naturally, the representation of rating or co-visitation behavior is the first issue to be solved. Nevertheless, directly extracting rating features from original data is difficult due to the sparsity of original data. Adaptable and representative behavior features are still scarce. Recently, Yang *et al.* [9] constructed an association graph based on original rating data and investigated a probabilistic inference model to identify anomalies.

A careful reading of the literature suggests that determining target items, eliminating disturbed data, and exploiting stepwise mechanisms, have considerable potential for accurate detection. However, (a) purely counting the number of users who have rated an item for determining the target item is uncontrollable and inelastic under different attack situations; (b) eliminating noise data according to disturbed information such as unpopular items and inactive users is unintelligent


v_1
$\mathcal{R}_{v_1 1}, \mathcal{R}_{v_1 2}, \cdots, \mathcal{R}_{v_1 |\mathcal{I}|}$
v_k
$\mathcal{R}_{v_k 1}, \mathcal{R}_{v_k 2}, \cdots, \mathcal{R}_{v_k |\mathcal{I}|}$
v_m
$\mathcal{R}_{v_m 1}, \mathcal{R}_{v_m 2}, \cdots, \mathcal{R}_{v_m |\mathcal{I}|}$
Add fake user one by one
Get w_{v_k} based on $\mathbf{R}^{(n+i)\times|\mathcal{I}|}$ by solving $\min \|w_{v_k}\|_2^2 + \varrho \cdot l$
Select n items as filler items according to w_{v_k}
For filler item j, assign a rating $\mathcal{R}_{v_k j} \sim \mathcal{N}(\mu_j, \sigma_j^2)$
Determine target items (RTI and UTI)
For each target item t, assign a rating $\mathcal{R}_{v_k t} = \mathcal{R}_{max}$
Genuine user profiles
Select mimicked genuine users
Randomly select
Carefully select
According to different attack scenarios
Mimicking rating details
Filler items I_F
Selected items I_S
Target items I_T
$\rho(\cdot)$
$\sigma(\cdot)$
$\mathcal{R}_{min}$ or $\mathcal{R}_{max}$
Assign a rating score for each item
A co-visitation graph
$m_{12} : w_1 + m_{12} \geq \eta_0$
High knowledge
Medium knowledge
Access the co-visitation graph G and the popularity threshold η_0
Access the popularity of each item and see its recommendation list
Promotion: *max* IUI (PH) Demotion: *max* DUI (DH)
Promotion: *max* IUI (PM) Demotion: *max* DUI (DM)

(a) Optimal-injection attacks (b) Profile injection attacks (c) Co-visitation injection attacks

Fig. 1. The basic structure of representative injection attacks, including optimal-injection attacks, profile injection attacks, and co-visitation injection attacks.

in reality; and (c) the reliability of label space in the final determination of concerned attacks lacks further examination. This work, differing from existing studies: (1) aims to globally discriminate target items according to both the influential neighbor disclosure of users or items and the estimation of their risk attitude; (2) adaptively filters out noise data through optimal subset selection based on influence maximization; (3) definitively deals with both *gray* (indefinite) labels and identified labels of injection profiles; and (4) explores a new detection framework that is applicable to diverse attacks.

III. Threat Model

A. Representative Threats

1) Optimal-Injection Attacks: Graph-based Recommendations (GBRs) [2] are favored in practice, which perform a random walk with a stationary probability on a graph to provide recommendations. Nevertheless, GBRs are vulnerable to poisoning attacks [2]. For convenience, in this paper, the poisoning attacks are called as *optimal-injection* attacks (see Figure 1(a)). To achieve the goal of optimal-injection attacks, attackers inject some fake users to the system by promoting a target item t to as many normal users as possible. Suppose the fraction of normal users whose top-K recommendations include t after the attack is $h(t)$ (termed the *hit ratio* of t). The attackers' goal is to maximize $h(t)$. Concretely, each attacker gives a high rating score to t and well-crafted ratings to certain selected items (called *filler items* [2]). A crucial task for the attackers is to determine the filler items and their ratings. Thus, item t hits user u if t is among the recommendation list of u, otherwise, t does not hit u.

Given a bipartite graph $\mathcal{G} = (\mathcal{U}, \mathcal{I}, \mathcal{E})$, where $\mathcal{U}$, $\mathcal{I}$, and $\mathcal{E}$ represent the set of users, items, and edges, respectively, suppose w_v denotes the vector of weights of edges that connect attacker v and all items, the weight of the edge $e_{vi} \in \mathcal{E}$ between v and item i can be modeled as a continuous variable w_{vi}, instead of an integer rating score. The loss function of all normal users over the stationary probability distribution can be defined as, $l = \sum_{u\in S} \sum_{i\in L_u} g(p_{ui} - p_{ut})$, where $g(x) = \frac{1}{1+\exp(-x/b)}$ and b is a parameter termed *width*. $\mathcal{S}$ represents the set of normal users who have not rated the target item yet. p_{ui} denotes the i-th probability of stationary distribution of random walk that starts from user u. Since normal users often rate a small number of items, the number of filler items for each attacker is at most n to avoid detection, which essentially constrains the values of w_v. Thus, the approximate optimization problem can be finally formulated as, min $\|w_v\|_2^2 + \varrho \cdot l$, $s.\ t.\ w_{vi} \in [0, \mathcal{R}_{max}]$, where ϱ balances the regularization term and the loss function. $\mathcal{R}_{max}$ denotes the maximum rating in the system.

In *optimal-injection* attacks, attackers are generated one by one. For an attacker v_k, we first solve the approximate optimization problem with the current rating matrix in order to obtain w_{v_k}. Then, we assign the maximum rating score $\mathcal{R}_{max}$ to t for *promotion* attacks. Note that, demotion is a special case of promotion [2]. After that, n items with the largest weights are selected as filler items. Accordingly, each filler item is assigned a discretized rating score by sampling a number from a normal distribution. More implementation details of *optimal-injection* attacks will be introduced in section V-A.

2) Profile Injection Attacks: Profile injection attacks to neighborhood-based CFRs have been received much attention in the last two decades [3], [7], [9], [10], [11]. Attackers aim to make a *target* item be recommended to as many genuine (normal) users as possible through a well-designed attack model. In profile injection attacks (as shown in Figure 1(b)), an attacker's rating profiles can be generally divided into three parts, including *selected items* I_S, *filler items* I_F, and *target items* [3]. The selected items are used to ensure the similarity between attackers and mimicked genuine users by designing corresponding ratings. Accordingly, the filler items are exploited in order to make the effect of attack controllable. Concretely, given I_S, I_F, and a target item i_t, attackers can carefully design corresponding ratings for I_S and I_F respectively using functions σ and ρ, where function $\sigma(\cdot)$ to I_S and function $\rho(\cdot)$ to I_F are solved using the normal distribution of an item i, i.e., $\mathcal{N}(\bar{\mathcal{R}}_i, \bar{\sigma}_i^2)$, or the global normal distribution, i.e., $\mathcal{N}(\bar{\mathcal{R}}, \bar{\sigma}^2)$. The attackers empirically give i_t the maximum rating score $\mathcal{R}_{max}$ (for *push* attacks) or the

TABLE I
NOTATION AND DESCRIPTION

Notation	Description
$\mathcal{U}$	set of users
$\mathcal{U}^s$	set of selected users
$\mathcal{I}$	set of items
$\mathcal{I}^s$	set of selected items
$\mathcal{A}_{x,y}$	set of items that users x and y have co-rated
$\delta_x(y)$	influence strength from user y to user x
$\mathbb{N}(x,y)$	common neighbors of both users x and y
Δ^f	final influence strength matrix
$\mathcal{V}(i)$	suspiciousness value of item i

minimum rating score $\mathcal{R}_{min}$ (for *nuke* attacks). This paper exploits five representative profile injection attacks including:

- Average Over Popular items attack (AOP): Attackers choose filler items according to their overall popularity among the genuine user base. I_F contains the top $x\%$ most *popular* items, $\rho(i) \sim \mathcal{N}(\bar{\mathcal{R}}_i, \bar{\sigma}_i^2)$, $I_S = \emptyset$ [12];
- Reverse bandwagon attack: Attack profiles are generated based on giving low ratings to least popular items and target item. I_S contains a set of *unpopular* items which have been rated by a small number of users [8], $\sigma(i) = \mathcal{R}_{min}$ or R_{max}, and $\rho(i) \sim \mathcal{N}(\bar{\mathcal{R}}, \bar{\sigma}^2)$;
- Bandwagon (average) attack: Attackers generate profiles with high ratings to well-known popular items and the highest possible rating to the target item. Selected items are chosen from popular items. Each selected item i is assigned a rating via $\sigma(i) = R_{max}$ or R_{min} (push or nuke attacks). Filler items are randomly selected from the system. For each filler item i, $\rho(i) \sim \mathcal{N}(\bar{\mathcal{R}}, \bar{\sigma}^2)$;
- Power User Attack with the highest Aggregate Similarity score (PUA-AS): Power users are the users who have rated a large number of items like *active* users. We selected a part of power users according to aggregate similarity scores between users in descending order. The rating profiles of selected power users are directly used to construct attack profiles [13];
- Power Item Attack with the highest Number of Ratings (PIA-NR): Power items are the items with the highest number of ratings, which can influence a group of items like *popular* items. We select top-N items according to the number of ratings rated on the items in descending order [14]. For each item i, $\sigma(i) \sim \mathcal{N}(\bar{\mathcal{R}}_i\ \bar{\sigma}_i^2)$.

3) Co-Visitation Injection Attacks: Co-visitation injection attacks focus on co-visitation recommender systems which are based on co-visitation graphs [4]. Attackers inject well-designed co-visitations into association-rule-based CFRs to make recommendations as the attackers desire (see Figure 1(c)). Due to the intention of attackers (i.e., promotion or demotion) and the background knowledge of attackers (or termed *bounded* resources), we only exploited four different attack scenarios including Promotion attack with *High knowledge* (PH), Promotion attack with *Medium knowledge* (PM), Demotion attack with *High knowledge* (DH), and Demotion attack with *Medium knowledge* (DM) [4]. Due to space limit, in this paper, we only implemented the above four co-visitation injection attacks in our experiments.

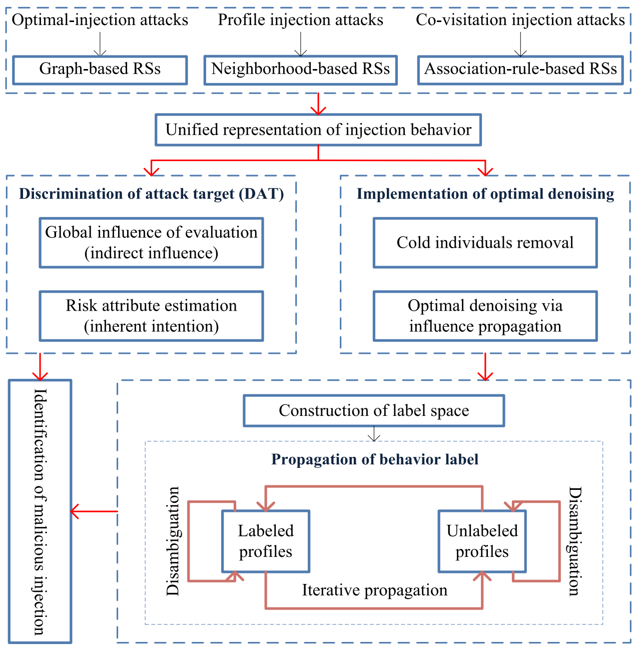

Optimal-injection attacks
Profile injection attacks
Co-visitation injection attacks
Graph-based RSs
Neighborhood-based RSs
Association-rule-based RSs
Unified representation of injection behavior
Discrimination of attack target (DAT)
Global influence of evaluation (indirect influence)
Risk attribute estimation (inherent intention)
Implementation of optimal denoising
Cold individuals removal
Optimal denoising via influence propagation
Identification of malicious injection
Construction of label space
Propagation of behavior label
Labeled profiles
Unlabeled profiles
Iterative propagation
Disambiguation
Disambiguation

Fig. 2. The basic framework of the proposed approach.

Take the PH for example, suppose i_1 is a selected *anchor* item and L_1 is the top-k recommendation list of i_1, attackers need to inject m_{12} fake co-visitations between items i_1 and i_2 in order to make i_2 appear in L_1 according to their background knowledge. For high knowledge, the attackers can access the total co-visitation graph and the popularity threshold η_0. To this end, m_{12} satisfies two conditions, (1) $s_{12} > s_{1k_1}$ and (2) $w_2 + m_{12} \geq \eta_0$, where s_{12} is the similarity between i_1 and i_2, and s_{1k_1} is the similarity between i_1 and the k-th ranked item k_1 in L_1 after the attack. Formally, $s_{12} = (w_{12} + m_{12})/f(w_1 + m_{12}, w_2 + m_{12})$, $s_{1k_1} = w_{1k_1}/f(w_1 + m_{12}, w_{k_1})$, where w_2 is the popularity of i_2 before the attack. w_{12} denotes the number of co-visitations between i_1 and i_2. f is the normalization factor. Ultimately, the goal of attackers is to maximize the increased probability of top-k user impression ($IUI = \sum_{i \in J_{i_t} - I_{i_t}} p_i$, for promotion attacks) or the decreased probability of top-k user impression ($DUI = \sum_{i \in I_{i_t} - J_{i_t}} p_i$, for demotion attacks), where $p_i = \frac{w_i}{w_1 + w_2 + \cdots + w_N}$, N and w_i represent the total number of items and the popularity of item i in the past, respectively. I_{i_t} is a set of items that a target item i_t is originally among the top-k ($k < N$) recommendation list in these items. After the attack, this set of items is enlarged or reduced to be J_{i_t} [4].

IV. Proposed Approach

A. Problem Statement

We formulate our detection problem as follows: Suppose the rating dataset $\mathcal{D}^r$ (for optimal-injection attacks and profile injection attacks) which contains both genuine and attack profiles is a set of triple $\langle u_i, i_j, \mathcal{R}_{ij} \rangle$, where u_i denotes a user,

i_j is an item, and $\mathcal{R}_{ij}$ represents the rating by u_i towards i_j. Let $\mathcal{U}$ and $\mathcal{I}$ represent the set of all users and items, respectively. The goal of our detection approach is to find out as many fake users as possible from $\mathcal{D}^r$ and filter out as many authentic users as possible simultaneously. Accordingly, for co-visitation injection attacks, suppose the co-visitation dataset $\mathcal{D}^v$ which contains both fake and authentic co-visitations is a set of triple $\langle v_i, v_j, m_{ij}\rangle$, where m_{ij} is the number of times that items v_i and v_j were co-visited. Let $\mathcal{I}^v$ represents the set of all items. The goal of our detection approach is to find out as many fake co-visitations as possible from $\mathcal{D}^v$ and filter out as many authentic co-visitations as possible simultaneously. Specially, we investigate a divide-and-conquer strategy to deal with the detection task. Figure 2 shows a basic framework of the proposed approach. Note that, Table I provides the notation we used throughout the paper.

For that purposes, we first convert $\mathcal{D}^r$ into a user-item bipartite graph $\mathcal{G}^r = (\mathcal{U}, \mathcal{I}, \mathcal{E})$, where $\mathcal{U}$ and $\mathcal{I}$ denote respectively the set of $|\mathcal{U}|$ users and $|\mathcal{I}|$ items, and $\mathcal{E} \subseteq \mathcal{U} \times \mathcal{I}$ is the set of edges ($e_{ui} \in \mathcal{E}$, $u \in \mathcal{U}$, and $i \in \mathcal{I}$). Regarding co-visitation injection attacks, $\mathcal{D}^v$ can be directly represented as a graph $\mathcal{G}^v = (\mathcal{V}, \mathcal{E}, \mathcal{W})$, where $\mathcal{V}$ (i.e., $\mathcal{I}^v$), $\mathcal{E}$, and $\mathcal{W}$ respectively are the set of $|\mathcal{V}|$ nodes (items), $|\mathcal{E}|$ co-visited edges, and $|\mathcal{W}|$ edge weights. Therein, $e_{ij} \in \mathcal{E}$, $v_i, v_j \in \mathcal{V}$, and $w_{ij} = m_{ij}$ ($w_{ij} \in \mathcal{W}$). Second, we select a set of suspicious items $\mathcal{I}^t$ as attack targets by both evaluating the global influence for each node (i.e., each user in $\mathcal{G}^r$, each item in $\mathcal{G}^v$) and estimating the risk attitude of rating (or co-visitation) intention for each user towards an item (or each item towards a co-visited item). Third, we filter out as many *disturbed* users (authentic users) as possible and obtain a set of selected users $\mathcal{U}^s \subset \mathcal{U}$ in $\mathcal{G}^r$ by exploiting an optimal denoising mechanism. Accordingly, we also can filter out as many *disturbed* items as possible and obtain a set of selected items $\mathcal{I}^s \subset \mathcal{I}^v$ in $\mathcal{G}^v$ through a similar way. Based on the selected users or items (i.e., $\mathcal{U}^s$ for profile injection attacks and optimal-injection attacks, $\mathcal{I}^s$ for co-visitation injection attacks) and $\mathcal{I}^t$, finally, fake injection ratings or co-visitations can be jointly identified.

B. Discrimination of Attack Target

To roughly discriminate target items, traditional solutions usually analyze the distribution of items (i.e., TIA). For each item i, concretely, they first calculate the number of users who have rated i with $\mathcal{R}_{max}$. Then, most of items can be filtered out using an empirical threshold of the number. Suspicious target items can be roughly determined according to the threshold. Note that, TIA can be fast implemented and be effectively used to capture target items especially facing with large attack sizes [10]. Intuitively, the more attackers join, the easier it is to identify the target items. However, some *popular* items (who were rated by a lot of users) is easily mistaken for attack targets, finally leading to high false alarm rates [10]. Moreover, the determination of empirical threshold is partly limited facing with different datasets. Thus, investigating an adaptive mechanism for DAT is naturally desirable. In this paper, we incorporate both the global influence evaluation of rating behavior and the risk attitude estimation of rating intention to comprehensively discriminate attack targets.

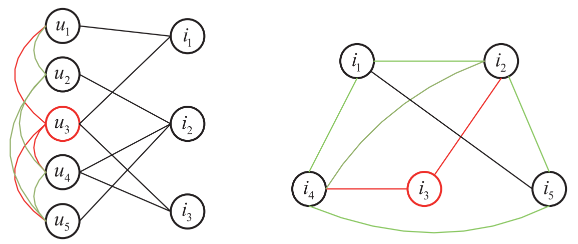


Fig. 3. An example for evaluating the influence of an individual (user or item). In the bipartite graph, the influence of user u_3 can be calculated according to the influence of its neighbors (i.e., users u_1, u_4, and u_5) in average. In the co-visitation graph, the influence of item i_3 is calculated according to the influence of items i_2 and i_4 in average.

1) Global Influence Evaluation: The goal of global influence evaluation is to find out the influential neighborhoods of user and disclose anomalous rating behaviors. Intuitively, similar and consistent injection behaviors among attackers on target items are relatively significant compared with that of genuine users [10], due to the fact that 1) attackers have a common motivation on the attack targets; and 2) the rating or visitation behaviors of genuine users are generally sparse. It is not easy to disclose the direct promotion or demotion behavior of attackers on target items. To this end, investigating the inherent relation of behaviors is desirable. Inspired from the influence strength mining in [15], [16], for profile injection attacks and optimal-injection attacks, we take both the neighbor distribution of user and item distribution into consideration in order to globally evaluate the influence strength of users. Given a user-item bipartite graph $\mathcal{G}$, the influence strength from user $y \in \mathcal{U}$ to user $x \in \mathcal{U}$ is defined in Eq. 1,

$$\delta_x(y) = \varphi \frac{|\mathbb{N}(x, y)|}{|\mathbb{N}(x)|} + (1 - \varphi)\frac{|\mathcal{A}_{x,y}|}{|\mathcal{A}_x|}, \tag{1}$$

where $\mathbb{N}(x, y)$ and $\mathbb{N}(x)$ are the common neighbors of both x and y and the neighbors of x, respectively. $\mathcal{A}_{x,y}$ represents the set of items that x and y have co-rated. $\mathcal{A}_x$ represents the set of items that x has rated. φ is a balance parameter. For co-visitation injection attacks, accordingly, $\mathbb{N}(x, y)$ denotes the common neighbors of both items x and y. $\mathcal{A}_{x,y}$ represents the sum of co-visitations between items x and y. $\mathcal{A}_x$ denotes the sum of co-visitations between item x and its neighbors.

As shown in Figure 3, given a user-item bipartite graph, associative links between users can be determined according to the *Pearson* correlation coefficient [9]. To evaluate the global influence of users and explore the propagation of influence, how to assign a stable influence value for a user is the first task. In this paper, we exploit an iterative influence propagation mechanism to obtain a stable influence value for each user. The influence strength obtained from similar neighbors only discovers the direct influence of user, but does not consider the indirect influence from neighbors. In reality, like information or virus, influence also propagates over networks, which produces different types of indirect influence. Inspired from atomic and iterative influence propagation in [16], we exploit two basic processes, i.e., *concatenation* and *aggregation*, for influence propagation. The atomic influence propagation can be instantiated as, $\delta_v(u) = \sum_{w\in\mathbb{N}(v)} \delta_v(w)\cdot\delta_w(u)$.

Suppose Δ_v represents the vector of the influence strength from all nodes (e.g., n nodes) in the network on node v, i.e., $\Delta_v = (\delta_v(u_1), \delta_v(u_2), \cdots, \delta_v(u_n))$. We use superscript to denote the propagation step, i.e., Δ^0 denotes the initial influence strength and Δ^1 denotes the influence strength after the atomic propagation. Then the atomic influence propagation can be represented as the matrix multiplication, $\Delta_v^1 = \Delta_v^0 \cdot \mathbf{T}$, where $\mathbf{T}$ is the transition matrix, $\mathbf{T} = (\Delta_{v_1}; \Delta_{v_2}; \cdots; \Delta_{v_n})$ and $\mathbf{T}(v,u) = \delta_v(u)$. Note that, $\delta_v(u)$ is calculated via Eq. 1.

Influence can propagate iteratively to collect the contribution of influence on longer paths [15], [16]. Therefore, the atomic influence propagation can be performed iteratively to propagate direct influence over the entire network. The influence strength on k-length paths can be calculated by k steps of atomic propagations. According to the matrix multiplication of atomic influence propagation, the influence vector after k-step atomic propagation is calculated by $\Delta_v^k = \Delta_v^{k-1}\cdot\mathbf{T} = \Delta_v^0\cdot\mathbf{T}^k$. Note that, Δ^k denotes the influence strength vector on k-length paths. Suppose the final influence between two nodes after k-step iterative propagation is defined as Δ^{f_k}, we collect all contributions of the influence on paths with the length ranging from 0 to k. Δ^{f_k} can be inferred from the sequences of propagation via a weighted linear combination if the addition operation is used as the aggregation function, i.e., $\Delta^{f_k} = \sum_{i=0}^{k} \beta_i \Delta^i$, where β_i represents the weight for the influence on i-length paths. Due to the decrease of indirect influence with the in crease of propagation length, β_i should decrease with the increase of iteration step i. Here, we incorporate conservative atomic influence propagation [16] to address this issue, which can be formulated as $\Delta^{f_t} = (1-\beta)\cdot\Delta^0 + \beta\cdot\Delta^{f_{t-1}}\cdot\mathbf{T}$.

Based on the above iterative propagation processes, the conservative influence propagation can be modeled as $\Delta^{f_t} = (1-\beta)\cdot\Delta^0\cdot\sum_{i=0}^{t-1}(\beta^i\cdot\mathbf{T}^i)+\Delta^0\cdot\beta^t\cdot\mathbf{T}^t$. Due to $\Delta^t = \Delta^0\cdot\mathbf{T}^t$, Δ^{f_t} can be reorganized as $\Delta^{f_t} = (1-\beta)\cdot\sum_{i=0}^{t-1}(\beta^i\cdot\mathbf{T}^i)+\beta^t\cdot\mathbf{T}^t$, where $\beta (0 \le \beta \le 1)$ is a damping factor. β^t decreases if t increases, which makes the effect of influence on longer paths smaller [16]. The goal of assigning different weights to the influences on different-length paths in the conservative influence propagation can be achieved. It is worth emphasizing that we use two different types of influence strength functions to evaluate the influence between nodes (users or items). To be different from $\delta_x(y)$, we exploit the following definition to evaluate the initial influence,

$$\mathbb{I}(v \to u) = \frac{\sum_{i\in\mathbb{N}(u)} N_i/|\mathbb{N}(u)|}{\sum_{j\in\mathbb{N}(v)} N_j/|\mathbb{N}(v)|}, \tag{2}$$

where $\mathbb{N}(v)$ denotes the set of neighbors of user v. N_i is the number of items rated by user i. Accordingly, the initial influence in a co-visitation graph is formally calculated by,

$$\mathbb{I}(v \to u) = \frac{\sum_{i\in\mathbb{N}(u)} m_{u,i}/|\mathbb{N}(u)|}{\sum_{j\in\mathbb{N}(v)} m_{v,j}/|\mathbb{N}(v)|}, \tag{3}$$

where $m_{u,i}$ is the number of visitations between u and i.

2) Risk Attitude Estimation for Attack Targets: As aforementioned, to determine attack targets, we leverage the influential neighbour disclosure to indirectly evaluate users' rating behaviors. Both neighbour association and individual rating distribution are taken into account for evaluating the neighbor influence strength. Understanding the inherent rating intention of users, however, is remained. Intuitively, the user satisfaction of rating an item may directly reflect the rating intention of user (or termed *rating psychology*). To evaluate this rating psychology, here, we incorporate the user risk preferences studied in the Prospect Theory [17], [18]. They stated that users usually have to take the risk for their own choices when making decisions under uncertainty.

In general, users have different evaluations for gains and losses. The decision-making process of users between probabilistic alternatives involving risks can be described as (1) all potential outcomes can be divided into gains or losses by setting a reference point that they consider as indifferent. In other words, the outcomes less than the reference point are considered as losses while the outcomes that are greater are considered as gains; and (2) users have different prospect value functions for each outcome and its probabilistic estimation. It is noteworthy that the prospect value function is a product of the value function and the weighting function [17]. The value function measures the subjective value of the gain or loss while the weighting function can be considered as a subjective distortion of the original probabilities.

Concretely, given a reference point $\bar{\mathcal{R}}_i$ which denotes users' expectation to a given item i, the gain or loss (outcome) denoted by x_i, can be defined as $x_i = (\mathcal{R}_{max} - \mathcal{R}_{min})\cdot\tanh(\mathcal{R}_{ui} - \bar{\mathcal{R}}_i)$, where $\mathcal{R}_{ui}$ represents the rating score that user u has rated on item i. Note that, u is one of the users that directly link to item i in the bipartite graph $\mathcal{G}$. The hyperbolic tangent function $tanh(\cdot)$ is used to normalize $(\mathcal{R}_{ui} - \bar{\mathcal{R}}_i)$ into $[-1, 1]$. Specifically, x_i is positive for gains and negative for losses. According to the outcome x_i, the value function can be defined as, $v(x_i) = \lambda x_i^{\alpha_1}$ if $x_i \ge 0$, $v(x_i) = -(-x_i)^{\alpha_2}$ otherwise, where $0 < \alpha_1, \alpha_2, \gamma < 1$. The value function is concave for gains, convex for losses, and steeper for losses than for gains [17], [18]. Furthermore, to evaluate the decision weight for all possible outcomes, the weighting function can be defined as, $\varpi(p_i) = p_i^{\beta_1}/(p_i^{\beta_1} + (1-p_i)^{\beta_1})^{\frac{1}{\beta_1}}$ if $x_i \ge 0$, $\varpi(p_i) = p_i^{\beta_2}/(p_i^{\beta_2} + (1-p_i)^{\beta_2})^{\frac{1}{\beta_2}}$ otherwise, where parameters β_1 and $\beta_2 (0 < \beta_1, \beta_2 < 1)$ control the shape of the weighting function. p_i is the probability of the gain or loss depending on x_i, which can be calculated by $p_i = Pr(x = i) = \int_{i-0.5}^{i+0.5} \varsigma\sigma x^{\varsigma-1}\exp(-\sigma x^{\varsigma})dx$, where ς and σ represent the shape and scale parameters, respectively.

Based on the value function and the weighting function, the prospect value for an item i can be calculated as below,

$$\mathcal{V}(i) = \begin{cases} \dfrac{\sum_{k=1}^{n_s} v(x_k)\varpi(p_k)}{\sum_{u\in\Gamma_i} \mathcal{O}(\mathcal{R}_{ui} = \mathcal{R}_{max})}, & if\ \mathcal{O}(\mathcal{R}_{ui} = \mathcal{R}_{max}) > 0 \\ \sum_{k=1}^{n_s} \psi\cdot v(x_k)\varpi(p_k), & if\ \mathcal{O}(\mathcal{R}_{ui} = \mathcal{R}_{max}) = 0 \end{cases} \tag{4}$$

```
Algorithm 1 Determination of Suspicious Targets
Require:
    A bipartite graph $\mathcal{G} = (\mathcal{U}, \mathcal{V}, \mathcal{E})$ and parameter $\alpha$;
Ensure:
    Suspicious items $\mathcal{I}^{sus}$;
1: $\mathcal{S}^i = \varnothing$ and $t = 1$;
2: Calculate the transition matrix $\mathbf{T}$ according to Eq. 1, Eq. 2, and Eq. 3;
3: repeat
4:     Calculate $\Delta^{f_t}$ in the $t$-th round and $t = t + 1$;
5: until $\|\Delta^{f_t} - \Delta^{f_{t-1}}\| < 10^{-5}$
6: Calculate the prospect value for each item using Eq. 4;
7: for each item $i \in \mathcal{V}$ do
8:     Calculate the suspiciousness value $s(i)$ using Eq. 5;
9:     $\mathcal{S}^i \leftarrow (s(i), i)$;
10: end for
11: Rank $\mathcal{S}^i$ in ascending order according to the suspiciousness values and obtain a ranked $\mathcal{S}'^i$;
12: Filter out cold items from $\mathcal{S}'^i$ and select top-I items as the suspected target items $\mathcal{I}^{sus}$;
13: return $\mathcal{I}^{sus}$;
```

where ψ and n_s denote the penalty factor and the number of states, respectively. $\mathcal{O}(\mathcal{R}_{ui} = \mathcal{R}_{max})$ is set as 1 if user u has rated item i with $\mathcal{R}_{max}$ score, 0 otherwise. Γ_i represents the set of users who have rated item i.

Based on the final influence strength matrix Δ^f and the prospect value $\mathcal{V}$, we calculate the suspiciousness value for each item by combining both of them. The balanced combination of neighbour influence (indirect) and rating intention (inherent) is formally defined as follows,

$$s(i) = \underbrace{\alpha \frac{\|\Delta^{f^{(1)}} - \Delta^{f^{(2)}}\|}{|\mathcal{I}|}}_{\textbf{indirect influence}} + \underbrace{(1-\alpha)\frac{\mathcal{V}(i)}{|\mathcal{I}|}}_{\textbf{inherent intention}}, \tag{5}$$

where $\Delta^{f^{(1)}}$ denotes the influence of each user (in a user-item bipartite graph) or item (in a co-visitation graph) based on Δ^f. $\Delta^{f^{(2)}}$ denotes the influence of each neighbor of a user in $\Delta^{f^{(1)}}$. In other words, we calculate the difference of influence between a user and its neighbors to evaluate the suspiciousness value for an item. For instance, in Figure 3(a), the suspiciousness value of item i_1 is calculated based on the influence of linked users (i.e., u_1 and u_3). Intuitively, the influence between attackers is relatively stable and compact during the iterative influence propagation due to their consistent attack behavior. The smaller the indirect influence value, the more suspicious. The smaller the value of inherent rating intention, the more suspicious. Thus, items with small suspiciousness values are considered as target items.

To demonstrate the effectiveness of DAT, we have implemented additional experiments as shown in Figure 4. Here, we first calculated a suspiciousness value for each item using TIA and DAT, respectively. Then, all candidate items have been ranked (i.e., in descending order for TIA, in ascending order for DAT) according to the corresponding suspiciousness values. Finally, we selected *top-I* items as target items from the ranked items and calculated DRs and FARs. Naturally, a user is considered as a concerned attacker if he rated at least one of the target items with $\mathcal{R}_{min}$ (for demotion attacks, $\mathcal{R}_{max}$ for promotion attacks) score. Figure 4(a) and Figure 4(c) demonstrate the effectiveness of DAT compared with TIA in terms of FARs. Note that, to be fair, we tried our best to obtain the highest DR and only analyze FARs. In our experiments, *top-I* was respectively set as 20 and 280 in the Bandwagon (average) and PIA-NR attacks. Accordingly, *top-I* was set to 5 and 70 on TripAdvisor and LibraryThing, respectively. We can see that DAT significantly outperforms TIA in different cases. Additionally, Figure 4(b) and Figure 4(d) also show the impact of different *top-I*s on detection performance. We can observe that selecting a reasonable top-I that corresponds to the highest DR and a small FAR using DAT is relatively easy and effective compared with using TIA. Algorithm 1 shows the basic process of discriminating suspicious targets. In Algorithm 1, $\mathcal{G}$ is constructed based on the original rating data. Note that, we empirically filtered out a part of *cold* items based on $\mathcal{S}'^i$. Undoubtedly, few ordinary items are mistaken for suspicious items. Nevertheless, the stages of optimal denoising and label prediction may partly make up for this misjudgment.

C. Denoising Solution

1) Conventional Denoising Solution: Previous researches mainly focus on step-by-step detection mechanisms [9], [10], [11], [19]. For stepwise detection approaches, filtering out disturbed information before detection is partly controllable. Given an original rating dataset, both the distributions of items and users are generally subjected to the *Power-law* distribution [19]. Concretely, most of items in the system are rated by a few users (e.g., one or two users), which are called *unpopular* items. A few items are rated by a lot of users, which are termed *popular* items. Accordingly, most of users rate a few items (e.g., one or two items) and a few users rate a lot items in the system, which are called *inactive* users and *active* users, respectively. Attackers rate some items (e.g., target items and filler items) in order to ensure the effect of attack. Intuitively, inactive users are insufficient to be considered as attackers due to their limited contribution for the attack. Similarly, unpopular items are also insufficient to be target items or filler items. Therefore, these inactive users (or unpopular items) can be filtered out in advance by exploiting an empirical threshold of user inactivity (or item unpopularity). The positive is that the use of empirical threshold for filtering out disturbed data is convenient and tractable in reality.

Nevertheless, purely relying on the empirical threshold to eliminate disturbed information is inelastic and unintelligent. How to globally determine disturbed users and items without strong prior knowledge and intelligently filter out noise data is a challenging issue to be solved. Thus, exploring a mechanism for selecting an optimal subset (at least contains all concerned attack profiles) from the original data especially under noisy environments will be discussed in what follows.

2) Optimal Denoising via Influence Propagation: Due to the limitation of conventional denoising solution, this paper

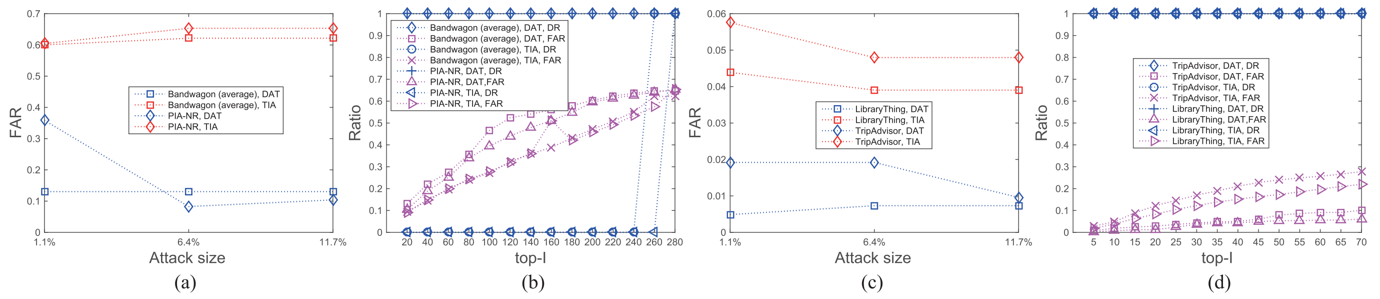


Fig. 4. A comparison between TIA and DAT for determining target items in different cases, where the filler size is 7.3% in profile injection attacks (the Bandwagon (average) and PIA-NR attacks are taken for example) and the filler size is 0.8% in optimal-injection attacks (attacks on LibraryThing and TripAdvisor are taken for example). (a) and (c) show the false alar rates (FAR) respectively using TIA and DAT in different attacks with different attack sizes, while the detection rates (DR) keep the highest. (b) and (d) show the DRs and FARs respectively using TIA and DAT with different *top-I*s in different attacks.

aims to explore a new denoising mechanism for accurate detection. Concretely, given a set of users $\mathcal{U} = \{u_1, u_2, \cdots, u_n\}$ including genuine users and attackers, where n denotes the number of users, the goal of denoising is to find a subset of at most k users $\mathcal{U}^s$ (i.e., $|\mathcal{U}^s| \leq k$, $\mathcal{U}^s \subseteq \mathcal{U}$) by maximizing an objective function f. The basic denoising task can be formulated as the following optimization problem,

$$\arg\max_{\mathcal{U}^s \subseteq \mathcal{U}} f(\mathcal{U}^s), \quad s.t.\ |\mathcal{U}^s| \leq k, \tag{6}$$

where f is monotone if for any $\mathcal{U}^s \subseteq \mathcal{U}$, $f(\mathcal{U}^s) \leq f(\mathcal{U})$. In reality, however, the exact objective value $f(\mathcal{U}^s)$ can not be obtained but rather only a *noisy* one $\mathcal{F}(\mathcal{U}^s)$. Generally, given a set $\mathcal{U}^s = \varnothing$, $\mathcal{U}^s$ can be conducted by repeatedly implementing k times through, a) obtaining an optimal node (user) by $u^* = \arg\max_{u \in \mathcal{U} \setminus \mathcal{U}^s} \mathcal{F}(\mathcal{U}^s \cup \{u\})$; and b) $\mathcal{U}^s = \mathcal{U}^s \cup \{u^*\}$.

Suppose A_i and p represent respectively the set of vertices that are activated in the i-th round and an independent probability (termed *propagation probability*), the optimal solution $\arg\max \mathcal{F}(\mathcal{U}^s)$ can be obtained by iteratively implementing the following influence maximization process r times:

1) Given a user-user graph $\mathcal{G}_u = (\mathcal{U}, \mathcal{E})$, for any $e_{uv} \in \mathcal{E}$ such that user $u \in A_i$ and user v is not yet activated. In addition, v is activated by u in the next iteration round.
2) A refined graph $\mathcal{G}'_u$ is generated by removing each edge from $\mathcal{G}_u$ with p. Particularly, the probability on either from u to v or from v to u is the same p.
3) A random set of vertices reachable from $\mathcal{U}^s$ in $\mathcal{G}'_u$ is obtained by a random precess of influence cascade [20], which is denoted as $\mathcal{R}_{\mathcal{G}'}(\mathcal{U}^s)$. We calculate the size of $\mathcal{R}_{\mathcal{G}'}(\{u\})$ for all vertices ($u \in \mathcal{U}$). For every $u \in \mathcal{U} \setminus \mathcal{U}^s$, the additional number s_u of vertices in $\mathcal{G}'_u$ which are influenced by selecting u into $\mathcal{U}^s$ is $|\mathcal{R}_{\mathcal{G}'}(\{u\})|$ if u dose not belong to $\mathcal{R}_{\mathcal{G}'}(\mathcal{U}^s)$, or 0 if $u \in \mathcal{R}_{\mathcal{G}'}(\mathcal{U}^s)$.
4) The average s_u is calculated for all $u \in \mathcal{U} \setminus \mathcal{U}^s$ after r times. Finally, the best candidate vertex u with the best average s_u is obtained by solving $\arg\max_{u \in \mathcal{U} \setminus \mathcal{U}^s}(\{u\})$.

Despite the simplicity and *greedy* nature in Eq. 6, it may be insufficient under the noisy environments. Importantly, some potential domination relationships between solutions may be ignored in the process of optimal subset selection. Given

Algorithm 2 Optimal Denoising for User Selection

```
Require:
    All users $\mathcal{U}$, parameters $\zeta$, $T_1$, $T_2$, and $k$.
Ensure:
    A set of well-selected users $\mathcal{U}^s$;
 1: Given an initial boolean vector $\mathbf{c} \in \{0\}^n$, solution space $\mathcal{C} = \{\mathbf{c}\}$, and influence $\mathcal{F}^u$ for all users;
 2: repeat
 3:   Randomly select a solution from $\mathcal{C}$ and conduct a new solution $\mathbf{c}'$ by randomly flipping each bit of an archived solution selected from $\mathcal{C}$ with the probability $1/n$;
 4:   Constrain $\mathbf{c}'$ such that $\mathbf{c}_i = 1$ and $u_i \in \mathcal{U}^f$, where $\mathcal{U}^f = \{u_i \in \mathcal{U} | \mathcal{F}^{u_i} \leq \zeta\}, (i = 1, 2, \cdots, n)$;
 5:   Add $\mathbf{c}'$ into $\mathcal{C}$ if $\mathbf{c}'$ is not dominated by any previously archived solution $\tilde{\mathbf{c}} \in \mathcal{C}$;
 6:   Randomly select two solutions from a constrained solution space $\mathcal{C}^s = \{\tilde{\mathbf{c}} \in \mathcal{C} | |\tilde{\mathbf{c}}| = |\mathbf{c}'|\}$ uniformly without replacement under $T_2$ iterations, and choose a better solution $\dot{\mathbf{c}}$ according to their $\mathcal{F}$ values and update $\mathcal{C} = \mathcal{C} \cup \dot{\mathbf{c}}$ and $\mathcal{C}^s = \mathcal{C}^s \setminus \dot{\mathbf{c}}$;
 7: until exceeds $T_1$ iterations
 8: Obtain the optimal solution $\mathbf{c}^*$ based on the final solution space $\mathcal{C}$, $\mathbf{c}^* = \arg\max_{\mathbf{c} \in \mathcal{C}, |\mathbf{c}| \leq k} \mathcal{F}(\mathbf{c})$;
 9: Select a subset of users $\mathcal{U}^s$ with at most $k$ items from $\mathcal{U}$ according to $\mathbf{c}^*$, $\mathcal{U}^s = \{u_i \in \mathcal{U} | \mathbf{c}^*_i = 1\}, (i = 1, 2, \cdots, n)$;
10: return $\mathcal{U}^s$;
```

a boolean vector $\mathbf{c} \in \{0,1\}^n$ (a subset $\mathcal{U}^s$ of $\mathcal{U}$), where $\mathbf{c}_i = 1$ if $u_i \in \mathcal{U}^s$ and $\mathbf{c}_i = 0$ otherwise, Eq. 6 can be formally reformulated as a bi-objective maximization problem, $\arg\max_{\mathbf{c} \in \{0,1\}^n}(f_1(\mathbf{c}), f_2(\mathbf{c}))$, where $f_1(\mathbf{c}) = -\infty$ if $|\mathbf{c}| \geq 2k$, $f_1(\mathbf{c}) = \mathcal{F}(\mathbf{c})$ otherwise. $f_2(\mathbf{c}) = -|\mathbf{c}|$ [21]. The bi-objective optimization problem aims to maximize the original objective and simultaneously minimize the subset size. Nevertheless, in noisy environments, a worse solution may appear to have a better $\mathcal{F}$ value and survive to replace the true better solution as [21] mentioned. Therefore, in order to keep the solutions with close $\mathcal{F}$ values rather than only one with the best $\mathcal{F}$ value in a candidate solution space, here, we incorporate an improved *Pareto* Optimization method for the optimal subset selection [22].

Concretely, given a user-user graph, a noisy objective function $\mathcal{F}$ based on the influence propagation of users, and a boolean vector $\mathbf{c} = \{0, 1\}^n$ for solutions, the goal of optimal denoising is to gradually remove worse solutions and select optimal solutions with close $\mathcal{F}$ values under limited iterations. In other words, we aim to filter out as many disturbed users (genuine users) as possible and keep all concerned attackers. Suppose the set of current solutions is denoted as $\mathcal{C} = \{\mathbf{c}\}$, we iteratively implement the selection of optimal solution according to the influence of users. Algorithm 2 describes the basic process of optimal denoising. Notably, in our detection stage, we considered the advantage of conventional denoising solution during the process of optimal denoising. Despite the well-selected users via the optimal denoising, few disturbed users (genuine users) may still be contained in the selected users, which finally leads to small FARs. Thus, further identifying malicious users and filtering out genuine users for accurate detection are necessary to be focused.

D. Identification of Malicious Injections

1) Construction of Label Space: The labeling information of user rating behavior is very critical for anomaly detection. However, there is no any useful labeling information that can be used for supervised learning in original data. Thus, how to extract useful labeling information from the original data is the first task. In the supervised label learning [23], let $\mathcal{X} = \mathbb{R}^d$ be the d-dimension instance space, a ground-truth label y_i for instance $\mathbf{x}_i$ is very crucial for supervised learning. Nevertheless, assigning a precise label for an instance (individual) is intractable in reality. Here, we try to construct an associate candidate label set for an individual to provide multifaceted behavior labels instead of assigning a ground-truth label. To construct the label space of individuals (users or items), we propose to exploit three labeling rules, including the activity of individual, link information, and attack intention.

First, to incorporate the activity of individual for rating or co-visitation behavior labeling, the activity $\mathfrak{A}_u$ of user u, which is defined as the number of items rated by u, is calculated. As discussed in section IV-C, *inactive* users are impossible to be considered as attackers. Given a threshold ξ_1 of activity of user, we can roughly divide the original users into two types, i.e., a user u is labeled (e.g., ℓ_1) as a genuine user if $\mathfrak{A}_u < \xi_1$ or an attacker otherwise. But inevitably, some of genuine users may be mistakenly labeled as attackers in case of $\mathfrak{A} \geq \xi_1$. Comparatively, genuine users are easy to be labeled when $\mathfrak{A} < \xi_1$. Accordingly, in co-visitation injection attacks, the number $\mathcal{C}_i$ of co-visitations of an item i, which is defined as the sum of co-visitations between item i and its neighbours, is easy to be calculated. We also use a threshold ξ_2 of co-visitation of node to label all nodes, i.e., an item i is labeled as a *fake* node if $\mathcal{C}_i \geq \xi_2$, or a normal node otherwise.

Second, to utilize the linkage information of graph, we extract labeling information for each individual from the perspective of connective structure of graph. Given a graph, we calculate the degree $\mathfrak{D}_i$ for node i. $\mathfrak{D}_i$ is defined as the number of linked neighbors of node i. The association between attackers is intuitively more significant than that of genuine users, which can be partly evaluated via the construction of association graph [9]. Concretely, a user i is labeled (e.g., ℓ_2) as an attacker if $\mathfrak{D}_i \geq \mu_1$ (a threshold) or a genuine user otherwise. Undoubtedly, the accuracy of labeling users is partly limited according to the setting of degree threshold. In a co-visitation graph, each node (item) can also be assigned a label through a similar way.

Third, to incorporate the intention of attack for enhancing the labeling space of user, we exploit the determined targets to distinguish attackers and genuine users. Here, a user is labeled (e.g., ℓ_3) as an attacker if he has rated at least one of the target items with $\mathcal{R}_{min}$ score (for demotion attacks) or $\mathcal{R}_{max}$ score (for promotion attacks), or a genuine user otherwise.

In summary, we construct a label vector for an individual i (i.e., $\mathcal{L}_i = \{\ell_1, \ell_2, \ell_3\}$) according to the above labeling rules. Specifically, a user is an attacker if $\ell_i = 1$ or a genuine user if $\ell_i = 0$ ($i = \{1, 2, 3\}$). Despite the limited validity of labeling information, comprehensively assessing the labeling space of individual for attack detection is desirable to be further investigated. In order to make full use of the advantage of labeling information, in this paper, we employ a partial label learning framework, which will be introduced in what follows.

2) Behaviour Label Propagation: As aforementioned, a part of user profiles are ambiguous between attack and authentic profiles due to the limited information of ground-truth labels in original data. Here, we only provide a set of candidate labels for each individual. To reduce the disambiguation of candidate labels and simultaneously carry rich labeling information extracted from unlabel user profiles, we employ an iterative label propagation mechanism to address the concerned task. Given a set of labeled user profiles $(\mathbf{D}^l, \mathcal{L})$ where $\mathbf{D}^l = \mathbb{R}^{m\times d}$ denotes a d-dimensional feature space and $\mathcal{L} = \mathbb{R}^{m\times 3}$ denotes the label space (m and d are the number of individuals and features, respectively), and a set of unlabeled user profiles $\mathbf{D}^u = \mathbb{R}^{n\times d}$, we can construct a weighted bipartite graph $\mathcal{G}^w = (\mathcal{V}^l, \mathcal{V}^u, \mathcal{E})$ between $\mathbf{D}^l$ and $\mathbf{D}^u$, where $\mathcal{V}^l$ and $\mathcal{V}^u$ respectively correspond to individuals in $\mathbf{D}^l$ and $\mathbf{D}^u$. $\mathcal{E}$ is the set of edges between $\mathcal{V}^l$ and $\mathcal{V}^u$, which is generated according to the neighbors between individuals in $\mathcal{V}^l$ and $\mathcal{V}^u$. For instance, to deal with the label propagation from definite rating (or co-visitation) behaviors in $\mathbf{D}^l$ to indefinite rating (or co-visitation) behaviors in $\mathbf{D}^u$, we first determine κ-nearest neighbors $\mathbb{N}(\mathbf{x}_i)$ in $\mathbf{D}^l$ for the attribute vector of user i (i.e., $\mathbf{x}_i \in \mathbf{D}^u$). Then, we obtain a weight vector $\hat{\mathbf{w}}_i$ for $\mathbf{x}_i$ by solving the following optimization problem,

$$\begin{aligned} &\min \|\mathbf{x}_i - \sum_{z=1}^{\kappa} w_{i,j_z} \cdot \mathbf{x}_{j_z}\| \\ &s.t.\ w_{i,j_z} \geq 0, \quad (j_z \in \mathbb{N}(\mathbf{x}_i), i \leq z \leq \kappa) \end{aligned} \tag{7}$$

where $w_{i,j_z} \geq 0$ if $(i, j_z) \in \mathcal{E}$ or $w_{i,j_z} = 0$ otherwise. A matrix $\mathbf{W} = [w_{i,j_z}]_{t\times s}$ (from $\mathbf{D}^l$ to $\mathbf{D}^u$) can be normalized by $\tilde{\mathbf{W}} = \mathbf{D}^{-1}\mathbf{W}$, where $\mathbf{D} = diag[d_1, d_2, \cdots, d_t]$ is a diagonal matrix with $d_i = \sum_{j_z=1}^{s} w_{i,j_z}$, and $\mathbf{H} = \tilde{\mathbf{W}}$.

In addition to the label propagation from $\mathbf{D}^l$ to $\mathbf{D}^u$, the completeness of labels can be further improved through an iterative label propagation procedure [24]. Concretely, the other label propagations including from $\mathbf{D}^u$ to $\mathbf{D}^l$, from $\mathbf{D}^l$

to $\mathbf{D}^l$, and from $\mathbf{D}^u$ to $\mathbf{D}^u$ can also be formulated via a similar way. In particular, labels in the propagations from $\mathbf{D}^l$ to $\mathbf{D}^l$ and from $\mathbf{D}^u$ to $\mathbf{D}^u$ can be disambiguated. In this paper, we respectively utilize $\mathbf{J}$ (from $\mathbf{D}^l$ to $\mathbf{D}^l$), $\mathbf{K}$ (from $\mathbf{D}^u$ to $\mathbf{D}^u$), and $\mathbf{L}$ (from $\mathbf{D}^u$ to $\mathbf{D}^l$) to denote the weight matrices. Inspired from [24], given two labeling confidence matrices $\mathbf{C}^l = [c^l_{i,c}]_{m\times 3}$ and $\mathbf{C}^u = [c^u_{i,c}]_{n\times 3}$ for $\mathbf{D}^l$ and $\mathbf{D}^u$, where $c^l_{i,c} = 1/|\mathcal{L}_i|$ if $\ell_c \in \mathcal{L}_i$, 0 otherwise; $c^u_{i,c} = 1/3$, consequently, according to $\mathbf{H}$, $\mathbf{J}$, $\mathbf{K}$, $\mathbf{L}$, $\mathbf{C}^l$ and $\mathbf{C}^u$, the label propagation procedure can be updated in T iterations by implementing the following steps, (1) $\mathbf{C}^u = \varepsilon \cdot \mathbf{H} \cdot \mathbf{C}^l + (1-\varepsilon)\cdot \mathbf{C}^u$; (2) $\tilde{\mathbf{C}}^l = \varepsilon \cdot \mathbf{J} \cdot \mathbf{C}^l + (1-\varepsilon)\cdot \mathbf{C}^l$; (3) $\mathbf{C}^u = \mu \cdot \mathbf{K} \cdot \mathbf{C}^u + (1-\mu)\cdot \mathbf{C}^u$; and (4) $\tilde{\mathbf{C}}^l = \mu \cdot \mathbf{L} \cdot \mathbf{C}^u + (1-\mu)\cdot \mathbf{C}^l$. Therein, ε and μ are used to balance the importance of labeled and unlabeled data. Note that, $\tilde{\mathbf{C}}^l = [\tilde{c}^l_{i,c}]_{m\times 3}$ and $\tilde{\mathbf{C}}^u = [\tilde{c}^u_{i,c}]_{n\times 3}$ need to be re-scaled so that the probability of labels outside of the candidate label set being the ground-truth labels of labeled data is cleared and simultaneously the sum of the labeling probability of a user (node) is normalized into one. To re-scale $\tilde{\mathbf{C}}^l$ to $\mathbf{C}^l$, $c^l_{i,c} = \tilde{c}^l_{i,c}/\sum_{\ell_k\in\mathcal{L}_i} \tilde{c}^l_{i,k}$ if $\ell_c \in \mathcal{L}_i$, 0 otherwise; Accordingly, to re-scale $\tilde{\mathbf{C}}^u$ to $\mathbf{C}^u$, $c^u_{i,c} = \tilde{c}^u_{i,c}/\sum_{\ell_k\in\mathcal{L}_i} \tilde{c}^u_{i,k}$ if $\ell_c \in \mathcal{L}_i$, 0 otherwise. Based on the updated $\mathbf{C}^l$ and $\mathbf{C}^u$, the multi-label individual profiles $(\mathbf{D}^l_i, \mathcal{L})$ can be disambiguated into single-label individual profiles $(\mathbf{D}^l_i, \hat{\ell}_i)$. Moreover, $\mathbf{D}^u_i$ can be assigned a valid label $\hat{\ell}_i$ by solving the optimization problem, i.e., $\hat{\ell}_i = \arg\max_{\ell_c\in\mathcal{L}} \frac{n_c}{\tilde{n}_c}\cdot c^l_{i,c}$, where $n_c = \sum_{i=1}^{m} c^l_{i,c}$ denotes the prior distribution of ℓ_c in the initial labeling confidence matrix $\mathbf{C}^l$, and $\tilde{n}_c = n_c + \sum_{i=m+1}^{m+n} c^u_{i,c}$ is the class mass of ℓ_c in the final labeling confidence matrices $\mathbf{C}^l$ and $\mathbf{C}^u$.

Ultimately, both $\mathbf{D}^l$ and $\mathbf{D}^u$ have been converted to single label data. Therefore, we can train a label classification model based on the single label data in order to accomplish the prediction of injection behaviors.

3) Determination of Malicious Injection Behaviors: Given a label prediction model $\mathcal{F}^l$ learned from the single label data, a subset of users $\mathcal{U}^s$ selected by the optimal denoising strategy, and the determined target items $\mathcal{I}^{sus}$ decided by DAT, we aim to identify the concerned attackers in $\mathcal{U}^s$ using $\mathcal{F}^l$ based on $\mathcal{I}^{sus}$. In $\mathcal{U}^s$, the class label of an unseen instance (a feature vector of a user) $\mathbf{x}^*$ can be predicted according to the minimum reconstruction error criterion on the disambiguated training examples [24], namely, the disambiguated label data $\{(\mathbf{D}^l_i, \hat{\ell}_i) | 1 \le i \le m\}$ and the disambiguated unlabeled data $\{(\mathbf{D}^u_i, \hat{\ell}_i) | m+1 \le i \le m+n\}$. Two weight vectors $\phi^* = [\phi^*_{i_1}, \phi^*_{i_2}, \cdots, \phi^*_{i_k}]$ $(i_a \in \mathbb{N}_l(\mathbf{x}^*), 1 \le a \le k)$ and $\psi^* = [\psi^*_{i_1}, \psi^*_{i_2}, \cdots, \psi^*_{i_k}]$ $(i_b \in \mathbb{N}_u(\mathbf{x}^*), 1 \le b \le k)$ can be determined w.r.t. $\mathbf{x}^*$ and its κ-neighbors in $\mathbf{D}^l$ and $\mathbf{D}^u$ by solving the optimization problem as defined in Eq. 7, where $\mathbb{N}_l(\mathbf{x}^*)$ and $\mathbb{N}_u(\mathbf{x}^*)$ identified separately denote the neighbors of $\mathbf{x}^*$ in $\mathbf{D}^l$ and $\mathbf{D}^u$, respectively. Based on ϕ^* and ψ^*, the label of $\mathbf{x}^*$ is predicted by solving the optimization problem,

$$\ell^* = \arg\min_{\ell_c\in\mathcal{L}} \|\mathbf{x}^* - \rho \cdot \sum_{a=1}^{k} \mathcal{O}(\hat{\ell}_{i_a} = \ell_c)\cdot \phi^*_{i_a} \cdot \mathbf{x}^*_{i_a} - (1-\rho)\cdot \sum_{b=1}^{k} \mathcal{O}(\hat{\ell}_{i_b} = \ell_c)\cdot \psi^*_{i_b} \cdot \mathbf{x}^*_{i_b}\|, \quad (8)$$

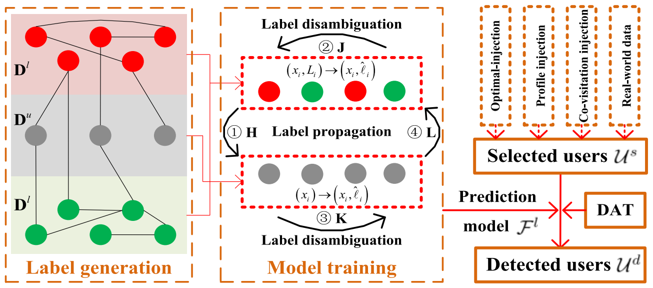


Fig. 5. A basic framework of determining malicious injections, which consists of label generation, model training, and the determination of malicious injections.

where ρ is used to balance the weight between the reconstruction errors on $\mathbf{D}^l$ and $\mathbf{D}^u$. $\mathcal{O}(\hat{\ell}_{i_a} = \ell_c)$ is 1 if $\hat{\ell}_{i_a} = \ell_c$, 0 otherwise. Note that, $\mathbf{x}_i$ can be represented through a truncated singular value decomposition [25]. For instance, $\mathbf{D}^l = \mathbf{U}_d \sum_d \mathbf{V}^T_d$, where $\mathbf{U}_d$ and $\mathbf{V}_d$ denote $m\times d$ orthonormal matrices corresponding to the selected singular values. $\sum_d$ is the diagonal matrix formed from the top-d singular values. The i-th row of $\mathbf{X}_d = \mathbf{U}_d \sum_d^{1/2}$ can be represented as the feature vector $\mathbf{x}_i$.

Figure 5 shows the basic process of identifying malicious users (or items). We first construct a label space for both definite labels (for all definite attackers and a part of definite genuine users, $\mathbf{D}^l$) and indefinite labels (for a part of "genuine" users, $\mathbf{D}^u$). Then, we train a label prediction model according to the iterative label propagation. Finally, the concerned nodes can be detected based on DAT and $\mathcal{U}^s$. In this paper, a node u $(u \in \mathcal{U}^s)$ is considered as a *sybil* node (i.e., $\mathcal{U}^d \leftarrow u$) if its predicted label $\ell^*_u = 1$, or a genuine node if $\ell^*_u = 0$. In profile injection attacks and optimal-injection attacks, particularly, a user u $(u \in \mathcal{U}^s)$ is considered as an attacker if $\ell^*_u = 1$ and u has rated item i $(i \in \mathcal{I}^{sus})$. In co-visitation injection attacks, accordingly, an item i is considered as a *fake* item if $\ell^*_i = 1$ and i has been co-visited at least with one target item i $(i \in \mathcal{I}^{sus})$.

V. Performance Evaluation and Analysis

A. Experimental Data and Setting

1) Real-World Data: To conduct attack data for profile injection attacks and optimal-injection attacks, we used four real-world datasets including Amazon,[1] LibraryThing,[2] TripAdvisor,[3] and MovieLens-100K (MovieLens).[4] Note that, the LibraryThing dataset contains 73,882 users, 337,561 items, and 979,053 reviews. The TripAdvisor dataset contains 12,773 users, 1,759 items, and 235,793 reviews. 943 users have rated 1,682 items in the MovieLens datasets.

2) Synthetic Data: For optimal-injection attacks, based on the LibraryThing, TripAdvisor, and MovieLens datasets, we implemented corresponding attack scenarios as introduced in section III to construct attack profiles using different attack

[1]http://jmcauley.ucsd.edu/data/amazon
[2]http://jmcauley.ucsd.edu/data
[3]http://sifaka.cs.uiuc.edu/~wang296/Data
[4]http://grouplens.org/datasets/movielens

sizes {0.5%, 0.8%, 1.6%} and filler sizes {0.8%, 2.5%}. With regard to the selection of the benchmark datasets, the above three datasets, which are widely used for evaluating recommender systems in the data mining community, are generally considered as authentic data for constructing injection attacks [2]–[4]. Target items in optimal-injection attacks are determined through two different ways, including Random Target Items (RTI) and Unpopular Target Items (UTI). Accordingly, we obtained 36 experimental datasets, including 3 different datasets, 3 different attack sizes, 2 different filler sizes, and 2 different ways of determining target items. Note that, demotion is a special case of promotion [2]. Therefore, we only analyze the promotion optimal-injection attacks in our experiments.

Regarding co-visitation injection attacks, we first generated an *Erdos-Renyi* (ER) random graph [4]. Then, we conducted co-visited edges and weights using corresponding co-visitation injection attack models with different background knowledge and different numbers of nodes, and finally inserted them into the original *ER* graph to construct a final co-visitation graph. Thus, we obtained 20 co-visitation graphs including 4 different attack models and 5 different numbers of nodes.

To generate attack data for profile injection attacks, based on the MovieLens-100K dataset, we implemented five representative shilling attack scenarios using different attack sizes {1.1%, 6.4%} and filler sizes {7.3%, 16.4%}. Each set of attack profiles is combined with the authentic data to construct the final experimental data. Thus, we have 20 datasets including 5 different attacks, 2 different attack sizes, and 2 different filler sizes. Note that, we only detect nuke shilling attacks and push attacks can be detected in the analogous manner.

It is worth emphasizing that all generated users in profile injection attacks and optimal-injection attacks have been labeled as attackers, while some of other users have been labeled as genuine users. Similarly, nodes (items) used to construct malicious co-visitations in co-visitation injection attacks have been labeled as anomalous nodes. Some of other nodes have been labeled as genuine nodes. Comparative experiments on the above data will be analyzed and discussed in what follows.

3) Evaluation Metric: To measure the performance of the presented methods, we uniformly exploit the Detection Rate (*DR*) and the False Alarm Rate (*FAR*) [3]. Concretely, *DR* is defined as the number of detected attackers divided by the number of all attackers in the system. *FAR* is defined as the number of genuine (authentic) users that are predicted as attackers divided by the number of genuine users. *DR* and *FAR* can be described as $DR = \frac{|\mathcal{U}^f \cap \mathcal{U}^a|}{|\mathcal{U}^a|}$ and $FAR = \frac{|\mathcal{U}^f \cap \mathcal{U}^g|}{|\mathcal{U}^g|}$, where $\mathcal{U}^f$, $\mathcal{U}^a$, and $\mathcal{U}^g$ represent the set of the detected users, all attackers, and all genuine users, respectively.

The ideal goal of detection is to obtain the highest DR and the lowest FAR. However, DR and FAR may have misjudgment and false alarm in reality, resulting in some authentic user profiles being wrongly excluded from the recommendation. It is noteworthy that this exclusion may affect the relevant recommendation in some cases. In other words, we may sacrifice some of main objectives of the recommendation while obtaining a specific detection objective.

B. Performance Analysis

1) Baselines: To demonstrate the effectiveness of the proposed TBOS, we selectively introduce the following representative methods as baselines for performance comparison:

- DeR-TIA [11]: DeR-TIA is a tow-step detection model, which uses an improved metric based on Degree of Similarity with top neighbors (DegSim) and Rating Devation from Mean Agreement (RDMA). TIA is designed to roughly determine target items.
- PCA-based [12]: The PCA-based detection model calculates the principal components of the covariance matrix of the ratings dataset, which actually identifies an attack cluster as a set of profiles with low pairwise covariance.
- CBS [7]: CBS is a label-propagation-based detection model, which calculates the spam probability of a user by incorporating all of the spam probability of its adjacent items, as well as the connections between the user and corresponding items.
- TIA-UPot [10]: Based on suspicious target items determined by TIA, the first-order (unary) potential of Conditional Random Field (CRF) is used to define the likelihood of a label motivated from a node being assigned to suspicious targets. TIA-UPot exploits the unary potential (UPot) to represent the smooth boundary of dense rating (or co-visitation) behaviors.
- TIA-PCRF [10]: TIA-PCRF leverages the second-order (pairwise) potential of CRF (PCRF) to represent the smooth boundary of dense rating (or co-visitation) behavior. TIA is adopted to determine target items.
- Ref-TIA [9]: Ref-TIA refines the links of source-target association graph via recursive propagation as well as incorporates the graphic representation of coupled networks, which combines the refinement of association links (Ref) and TIA.
- Ref-CLP [9]: Based on the refinement of source-target association graph, the Coupled Link Prediction (CLP) of target network is introduced to deal with the link redetermination of target network.
- Det-DAT: Det-DAT without the labelling scheme is designed to independently capture injection profiles, which is used to demonstrate the effectiveness of DAT.
- SubS-DAT: Based on suspicious targets determined by DAT, the optimal Subset Selection (SubS) is adopted to verify the effectiveness of the optimal denoising strategy.

It should be noted that it is difficult to fully verify the superiority of detection performance of TBOS compared with all baselines through experiments. The selection of benchmarks and the fairness of different attack experiments applicable to these methods are usually not easy to take into account.

2) Optimal-Injection Attack Detection: A list of experiments has been implemented as shown in Table II. Specially, five different benchmarks including DeR-TIA, PCA-based, CBS, Det-DAT, and SubS-DAT have been exploited in order to demonstrate the effectiveness of TBOS. Note that, Det-DAT and SubS-DAT are exploited without the labelling scheme to respectively demonstrate the effectiveness of DAT and the effectiveness of both the optimal denoising strategy and DAT.

TABLE II

DETECTION RESULTS OF THE PRESENTED METHODS UNDER OPTIMAL-INJECTION ATTACKS BASED ON THREE DIFFERENT DATASETS

Data	FS	Method	Metric	Attack size					
				Random target items			Unpopular target items		
				0.5%	0.8%	1.6%	0.5%	0.8%	1.6%
MovieLens	0.8%	DeR-TIA	DR	0.3333	1.0	1.0	1.0	1.0	1.0
			FAR	0.4675	0.2650	0.2075	0.4825	0.4150	0.2400
		PCA-based	DR	0.2000	0.2000	0.4000	0.2000	0.2000	0.4000
			FAR	0.2450	0.2525	0.2325	0.2450	0.2675	0.2325
		CBS	DR	1.0	1.0	1.0	1.0	1.0	1.0
			FAR	0.4758	0.4758	0.4758	0.4572	0.4572	0.4572
		Det-DAT	DR	1.0	1.0	1.0	1.0	1.0	1.0
			FAR	0.3333	0.1800	0.1800	0.3925	0.1800	0.1800
		SubS-DAT	DR	1.0	1.0	1.0	1.0	1.0	1.0
			FAR	0.1785	0.0325	0.0250	0.1785	0.0325	0.0275
		TBOS	DR	**1.0**	**1.0**	**1.0**	**1.0**	**1.0**	**1.0**
			FAR	**0.0375**	**0**	**0**	**0.0375**	**0**	**0**
	2.5%	DeR-TIA	DR	0.3333	1.0	0.4000	1.0	1.0	1.0
			FAR	0.4825	0.2650	0.2475	0.4825	0.2700	0.2400
		PCA-based	DR	0.2000	0.2000	0.4000	0.2000	0.2000	0.4000
			FAR	0.2450	0.2675	0.2350	0.2450	0.2675	0.2375
		CBS	DR	1.0	1.0	1.0	1.0	1.0	1.0
			FAR	0.4758	0.4758	0.4758	0.4572	0.4572	0.4572
		Det-DAT	DR	1.0	1.0	1.0	1.0	1.0	1.0
			FAR	0.3333	0.1800	0.1800	0.3925	0.1800	0.1800
		SubS-DAT	DR	1.0	1.0	1.0	1.0	1.0	1.0
			FAR	0.1785	0.0325	0.0250	0.1785	0.0325	0.0250
		TBOS	DR	**1.0**	**1.0**	**1.0**	**1.0**	**1.0**	**1.0**
			FAR	**0.0375**	**0**	**0**	**0.0375**	**0**	**0**
LibraryThing	0.8%	DeR-TIA	DR	0.2000	0.2000	0.4000	0.3333	0.2000	0.2000
			FAR	0.0365	0.0121	0.0463	0.0414	0.0048	0.0463
		PCA-based	DR	0.2000	0.2000	0.4000	0.3333	0.2000	0.2000
			FAR	0.0439	0.0439	0.0073	0.0414	0.0439	0.0024
		CBS	DR	1.0	1.0	1.0	1.0	1.0	1.0
			FAR	0.4365	0.4365	0.4365	0.4414	0.4463	0.4414
		Det-DAT	DR	1.0	1.0	1.0	1.0	1.0	1.0
			FAR	0.0414	0.1268	0.1317	0.0439	0.1268	0.1268
		SubS-DAT	DR	1.0	1.0	1.0	1.0	1.0	1.0
			FAR	0.0146	0.0243	0.0243	0.0146	0.0243	0.0243
		TBOS	DR	**1.0**	**1.0**	**1.0**	**1.0**	**1.0**	**1.0**
			FAR	**0**	**0**	**0**	**0**	**0**	**0**
	2.5%	DeR-TIA	DR	0.2000	0.2000	0.4000	0.3333	0.2000	0.2000
			FAR	0.0390	0.0390	0.0390	0.0390	0.0439	0.0365
		PCA-based	DR	0.2000	0.2000	0.4000	0.3333	0.2000	0.2000
			FAR	0.0390	0.0390	0.0390	0.0390	0.0439	0.0268
		CBS	DR	1.0	1.0	1.0	1.0	1.0	1.0
			FAR	0.5487	0.5487	0.5487	0.5536	0.5463	0.5561
		Det-DAT	DR	1.0	1.0	1.0	1.0	1.0	1.0
			FAR	0.0292	0.1268	0.1317	0.1097	0.1243	0.1243
		SubS-DAT	DR	1.0	1.0	1.0	1.0	1.0	1.0
			FAR	0.0097	0.0292	0.0243	0.0365	0.0243	0.0243
		TBOS	DR	**1.0**	**1.0**	**1.0**	**1.0**	**1.0**	**1.0**
			FAR	**0**	**0**	**0**	**0**	**0**	**0**
TripAdvisor	0.8%	DeR-TIA	DR	1.0	1.0	1.0	1.0	1.0	1.0
			FAR	0.4095	0.4095	0.3023	0.3023	0.3023	0.3023
		PCA-based	DR	1.0	1.0	1.0	1.0	1.0	1.0
			FAR	0.4095	0.4095	0.4095	0.3271	0.3271	0.4095
		CBS	DR	1.0	1.0	1.0	1.0	1.0	1.0
			FAR	0.4458	0.4458	0.4458	0.4458	0.4458	0.4458
		Det-DAT	DR	1.0	1.0	1.0	1.0	1.0	1.0
			FAR	0.1582	0.0287	0.2110	0.1582	0.1582	0.2110
		SubS-DAT	DR	1.0	1.0	1.0	1.0	1.0	1.0
			FAR	0.0335	0.0095	0.1486	0.1055	0.1031	0.1510
		TBOS	DR	**1.0**	**1.0**	**1.0**	**1.0**	**1.0**	**1.0**
			FAR	**0**	**0**	**0**	**0**	**0**	**0**
	2.5%	DeR-TIA	DR	1.0	1.0	1.0	1.0	1.0	1.0
			FAR	0.4095	0.3023	0.3023	0.4095	0.3023	0.3023
		PCA-based	DR	1.0	1.0	1.0	1.0	1.0	1.0
			FAR	0.4095	0.4095	0.4095	0.3271	0.3271	0.4095
		CBS	DR	1.0	1.0	1.0	1.0	1.0	1.0
			FAR	0.4577	0.4577	0.4577	0.4577	0.4577	0.4577
		Det-DAT	DR	1.0	1.0	1.0	1.0	1.0	1.0
			FAR	0.1271	0.1247	0.1247	0.1271	0.1247	0.2086
		SubS-DAT	DR	1.0	1.0	1.0	1.0	1.0	1.0
			FAR	0.0575	0.0575	0.0575	0.0599	0.0575	0.1271
		TBOS	DR	**1.0**	**1.0**	**1.0**	**1.0**	**1.0**	**1.0**
			FAR	**0**	**0**	**0**	**0**	**0**	**0**

To be fair, we have tried our best to make the DRs of all methods as close as possible to the highest. Then, we directly compare their FARs with TBOS's FAR. In Table II, we can observe that TBOS significantly outperforms all baselines almost in all cases with an advantage of 6.08% FAR in average. Particularly, TBOS shows a perfect detection level in some cases, i.e., the highest DR and the lowest FAR.

The second observation is that, the effectiveness of DAT, optimal subset selection, and Semi-Supervised Label Learning (SSLL), can be successfully demonstrated by comparing with the detection results of Det-DAT, SubS-DAT, and TBOS. First, purely relying on DAT for detecting attack profiles is more effective than that of TIA. Second, based on DAT, the advantage of optimal subset selection is significant. Moreover, based on the remaining users obtained by SubS-DAT, distinguishing anomalous users from the remained users can be further improved by SSLL.

With regard to DeR-TIA, the highest DR can be obtained in some cases. However, the FARs are unacceptable compared with TBOS. The reason may be that a) characterizing users' rating behavior is difficult, which directly affects the clustering effect in its second stage facing with optimal-injection attacks; and b) only relying on an empirical threshold of count is not easy to adapt to different situations. Note that, the empirical threshold of count was set as 6 in the experiments. Similarly, PCA-based also shows limited detection results compared with TBOS. Regarding CBS, its detection results seem stable almost in all cases. Nevertheless, the FARs are unacceptable. The reason may be that it is difficult to construct a weighted edge between nodes in a bipartite graph facing with optimal injection behaviors. The optimal construction of attack profiles may lead to the concealment of attack behaviors. In addition, the similar equivalence of bipartite graphs converted from the original rating matrix may affect its detection results, or possibly due to the fact that similar attack profiles are injected into the original rating data [7]. Note that, both parameters p_u and p_i for the user and product spam probability vectors used in CBS were set as 1.

3) Co-Visitation Injection Attack Detection: To evaluate the detection performance of TBOS under co-visitation injection attacks, we also have implemented a list of experiments as shown in Figure 6. Here, we exploited six different benchmarks including Kmeans-based, EM-based, DensityCluster-based, CBS, Det-DAT, and SubS-DAT to verify the effectiveness of TBOS. Note that, default parameters of *Weka* have been only used in Kmeans-based, EM-based, and DensityCluster-based. For these three baselines, nodes of a co-visitation graph are divided into two clusters using each clustering-based method. Intuitively, the cluster which has relatively few nodes is considered as suspicious nodes. In reality, the number of nodes which are concerned by attackers is significantly less than that of authentic nodes due to the limited cost of attacks. In Figure 6, we can see that TBOS significantly outperforms all baselines. By comparing all situations that the presented methods have the highest DR, we can observe that optimistic FARs of TBOS can be obtained, which shows an advantage of 3.83% FAR in average compared with all baselines. The other observation is that the advantages of DAT, SubS, and SSLL are significant. By keeping the highest DR, the FARs can be gradually reduced through observing the detected results of Det-DAT, SubS-DAT, and TBOS. These results also suggest that the effectiveness of step-by-step detection mechanism is positive.

4) Profile Injection Attack Detection: To examine the detection performance of TBOS under profile injection attacks, extensive experiments have been implemented in five representative shilling attacks as shown in Table III. In the experiments,

TABLE III

DETECTION RESULTS OF THE PRESENTED METHODS IN FIVE DIFFERENT PROFILE INJECTION ATTACKS WITH DIFFERENT CASES, WHERE $\mathcal{N}^a$ AND $\mathcal{N}^g$ DENOTE ATTACKERS AND GENUINE USERS, RESPECTIVELY

Attack	FS	Attack size ($\|\mathcal{N}^a\|/\|\mathcal{N}^g\|$)	Metric	Benchmarks								
				DeR-TIA	CBS	TIA-UPot	TIA-PCRF	Ref-TIA	Ref-CLP	Det-DAT	SubS-DAT	TBOS
AOP	7.3%	1.1%	DR	0	1.0	1.0	1.0	0.8000	1.0	1.0	1.0	**1.0**
		(10/943)	FAR	0.3181	0.5472	0.4104	0.1326	0.3073	0.3807	0.1108	0.0230	**0**
		6.4%	DR	0.5000	1.0	1.0	1.0	1.0	1.0	1.0	1.0	**1.0**
		(60/943)	FAR	0.2863	0.3266	0.4178	0.0742	0.4550	0.3807	0.0910	0.0230	**0**
	16.4%	1.1%	DR	0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	**1.0**
		(10/943)	FAR	0.3181	0.5483	0.4486	0.1209	0.4073	0.3785	0.1046	0.0210	**0**
		6.4%	DR	0.3000	1.0	1.0	1.0	1.0	1.0	1.0	1.0	**1.0**
		(60/943)	FAR	0.2990	0.3531	0.2842	0.0318	0.4450	0.3722	0.0910	0.0210	**0**
Reverse Band.	7.3%	1.1%	DR	0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	**1.0**
		(10/943)	FAR	0.3181	0.5472	0.3616	0.2375	0.4073	0.3775	0.1149	0.0275	**0**
		6.4%	DR	0.4166	1.0	1.0	1.0	1.0	1.0	1.0	1.0	**1.0**
		(60/943)	FAR	0.2916	0.3160	0.2333	0.0965	0.4073	0.3785	0.1056	0.0242	**0**
	16.4%	1.1%	DR	0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	**1.0**
		(10/943)	FAR	0.3181	0.5472	0.3266	0.1209	0.4083	0.3732	0.1075	0.0275	**0**
		6.4%	DR	0.1166	1.0	1.0	1.0	1.0	1.0	1.0	1.0	**1.0**
		(60/943)	FAR	0.3107	0.3340	0.1251	0.0201	0.4083	0.3669	0.0963	0.0089	**0**
Band. (Average)	7.3%	1.1%	DR	0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	**1.0**
		(10/943)	FAR	0.3573	0.4189	0.1633	0.1421	0.4073	0.3764	0.1322	0.0509	**0.0010**
		6.4%	DR	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	**1.0**
		(60/943)	FAR	0.2226	0.4189	0.1728	0.1516	0.4967	0.3775	0.1046	0.0300	**0**
	16.4%	1.1%	DR	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	**1.0**
		(10/943)	FAR	0.1611	0.4189	0.1580	0.1368	0.4083	0.3732	0.1322	0.0803	**0.0010**
		6.4%	DR	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	**1.0**
		(60/943)	FAR	0.2216	0.4189	0.1739	0.1527	0.4977	0.3658	0.1175	0.0533	**0**
PIA-NR	7.3%	1.1%	DR	0.1000	1.0	1.0	1.0	1.0	1.0	1.0	1.0	**1.0**
		(10/943)	FAR	0.3595	0.4189	0.1590	0.1378	0.4073	0.3807	0.1201	0.0023	**0.0021**
		6.4%	DR	0.8500	1.0	1.0	1.0	1.0	1.0	1.0	1.0	**1.0**
		(60/943)	FAR	0.1803	0.4189	0.1601	0.1389	0.4967	0.3807	0.1046	0.0021	**0**
	16.4%	1.1%	DR	0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	**1.0**
		(10/943)	FAR	0.3807	0.4189	0.1633	0.1399	0.4073	0.3807	0.1201	0.0023	**0.0021**
		6.4%	DR	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	**1.0**
		(60/943)	FAR	0.1399	0.4189	0.1654	0.1431	0.4967	0.3807	0.1046	0.0021	**0**
PUA-AS	7.3%	1.1%	DR	0.4000	1.0	1.0	1.0	1.0	1.0	1.0	1.0	**1.0**
		(10/943)	FAR	0.3658	0.4189	0.1601	0.1410	0.4073	0.3807	0.1046	0.0071	**0.0010**
		6.4%	DR	0.3166	1.0	1.0	1.0	1.0	1.0	1.0	1.0	**1.0**
		(60/943)	FAR	0.1558	0.4189	0.1707	0.1516	0.4988	0.3849	0.1046	0.0023	**0**
	16.4%	1.1%	DR	0.3000	1.0	1.0	1.0	1.0	1.0	1.0	1.0	**1.0**
		(10/943)	FAR	0.1495	0.4189	0.1622	0.1431	0.4073	0.3807	0.1046	0.0071	**0.0010**
		6.4%	DR	0.3166	1.0	1.0	1.0	1.0	1.0	1.0	1.0	**1.0**
		(60/943)	FAR	0.1558	0.4189	0.1717	0.1527	0.4988	0.3849	0.1046	0.0023	**0**

we exploited eight different benchmarks including DeR-TIA, CBS, TIA-UPot, TIA-PCRF, Ref-TIA, Ref-CLP, Det-DAT, and SubS-DAT. In Table III, we can see that the detection performance of TBOS outperforms all baselines almost in all cases. Intuitively, the larger the attack size, the easier it is to be detected. To be fair, we have tried our best to make each baseline get the highest DR. Therefore, we only make a comparison between FARs of all presented methods. In this way, an advantage of 2.3% FAR for TBOS can be obtained compared with all baselines.

The second observation is that few genuine users are mistakenly considered as attackers by TBOS when the attack size is small. Accordingly, the baselines are also prone to the same issue. The reason may be that a) counting the number of users who have rated an item with $\mathcal{R}_{min}$ or $\mathcal{R}_{max}$ (i.e., TIA) is insufficient when the attack size is small; and b) evaluating the influence propagation and risk attitude of users (i.e., DAT) is limited facing with small scale attacks. Note that, the effect of attack and the necessity of detection need to make a balance. In reality, attack and detection are a game process.

Furthermore, we also provide experimental results regarding the computational time of all presented methods as shown in Table IV. Note that, all experiments were implemented using MATLAB R2014a and Python 3.8.10 on a server with Intel(R) Xeon(R) W-2133 CPU 3.60 GHz, 64GB memory, and Linux operating system. We can observe that the effectiveness and efficiency of the proposed method are relatively acceptable compared with the baselines. The calculation time of TBOS is mainly consumed in global influence evaluation, risk attitude estimation, and optimal denoising. Conservatively, we improve the detection performance of TBOS by sacrificing computing consumption. Undoubtedly, our experimental results are only for reference.

TABLE IV

EFFICIENCY PERFORMANCE (CPU TIME, HOUR(h), MINUTE(m) AND SECOND(s)) OF BENCHMARKS COMPARED WITH TBOS, TAKE THE BANDWAGON (AVERAGE) ATTACK FOR EXAMPLE

Methods	# profiles	
	80,310	153,060
DeR-TIA	1h33m43s	2h22m28s
CBS	37m46s	45m42s
TIA-UPot	1h6m57s	1h23m39s
TIA-PCRF	1h8m29s	1h28m45s
Ref-TIA	35m58s	39m53s
Ref-CLP	1h4m40s	1h7m49s
Det-DAT	37m3s	53m14s
SubS-DAT	52m4s	1h21m23s
TBOS	58m13s	1h26m17s

5) Discovery on Amazon Data: To further verify the effectiveness of TBOS, we have implemented additional experiments on Amazon data as shown in Figure 7. Due to the large scale and sparsity of Amazon data, we have removed the *cold* users and items and have respectively selected a part of users and items before detection. In particular, we analyzed the Amazon data year by year. Inspired from the experiments of [7] (CBS), we finally got 2,220 users (including 28 demotion attackers and 15 promotion attackers) from 336,073

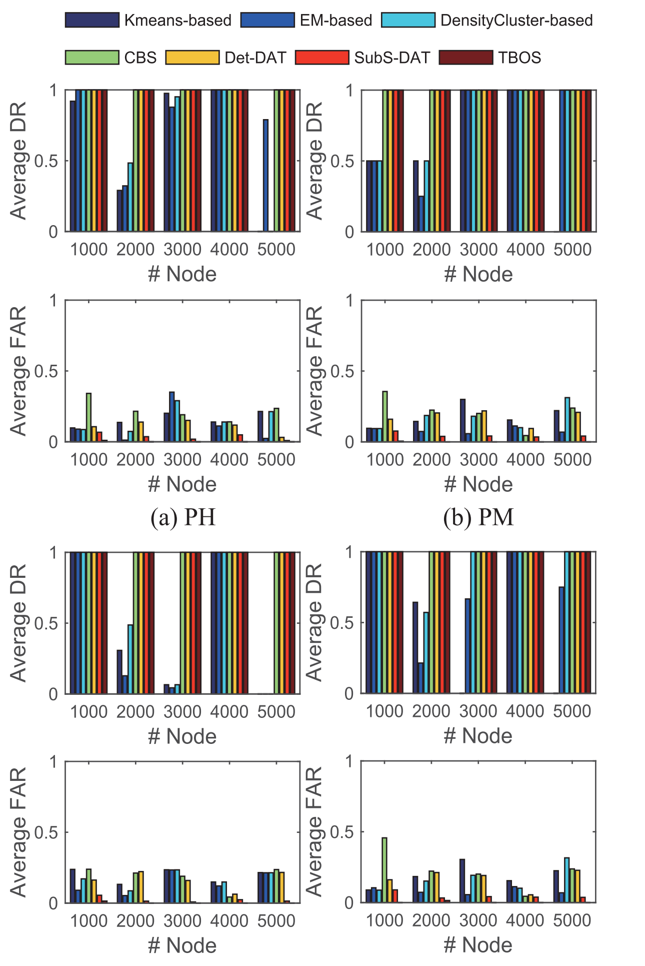


Fig. 6. Average detection results of all presented methods in different co-visitation injection attacks, where the popularity threshold ϵ_0 and top-k were set as 500 and 16, respectively.

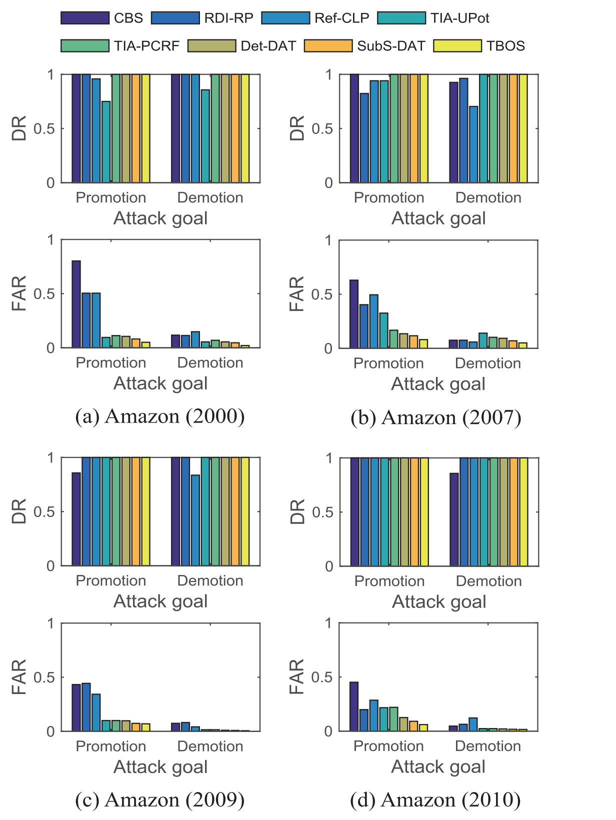


Fig. 7. Detection results of the presented benchmarks compared with the proposed approach on the Amazon data, including promotion and demotion attack scenarios.

users in Amazon (2000), 1,119 users (including 27 demotion attackers and 34 promotion attackers) from 995,481 users in Amazon (2007), 1,265 users (including 11 demotion attackers and 14 promotion attackers) from 1,478,772 users in Amazon (2009), and 893 users (including 14 demotion attackers and 18 promotion attackers) from 1,934,841 users in Amazon (2010), respectively. In the experiments, seven benchmarks including CBS, RDI-RP (removal of disturbed information and recursive propagation) [9], Ref-CLP, TIA-UPot, TIA-PCRF, Det-DAT, and SubS-DAT have been implemented to compare with TBOS in both promotion and demotion attacks. In Figure 7, we can see that the detection results of TBOS outperform all baselines almost in all cases with an advantage of 6.5% FAR in average. It is noteworthy that small FARs still exist using TBOS. This may be attributed to the limitation of determining spam users via human annotators, or possibly due to the fact that residual noise users after the optimal denoising stage affect the generation and prediction of labels.

C. Parameter Sensitivity Analysis

The sensitivity of parameters used in TBOS directly determines the detection performance. Therefore, reasonably determining an empirical threshold for each model parameter is a key task. To this end, we have implemented a list of experiments as shown in Figure 8, in order to analyze some crucial parameters and finally provide our suggestions for the determination of parameter threshold. It is worth emphasizing that it is very hard to fully evaluate all parameters used in the work due to space limit. Here, we only analyze six key model parameters, including α used in Eq. 5, k used to select a subset in the stage of optimal denoising, ε and μ used in the labeling propagation, ρ used in Eq. 8, and κ used to determine κ-nearest neighbours between $\mathbf{D}^l$ and $\mathbf{D}^u$. In Figure 8, we can observe that DRs and FARs are not sensitive to all parameters almost in all cases except for parameter k. For parameter k, the detection results gradually change with the increase of k ($5 < k < 150$). This is due to the fact that (1) for the optimal-injection attack on the TripAdvisor dataset, the number of attackers is 5 in the experiments. Selecting an ideal and optimal subset from the original users is limited due to the limited denoising performance when k is less than 5 or slightly larger than 5 (e.g., 10); (2) in AOP, the number of attackers is 110. Similarly, selecting all concerned attackers through the optimal denoising is impossible especially when k is smaller than 110; and (3) in PM, the number of vertexes of co-visitation graph and the number of *sybil* nodes are 1000 and 5, respectively. Choosing all sybil nodes using the optimal denoising is also difficult when k is small (e.g., $k < 50$). Apart from the above situations, we also can see that the detected results tend to be stable gradually with the increase of k. This phenomenon may imply that the denoising

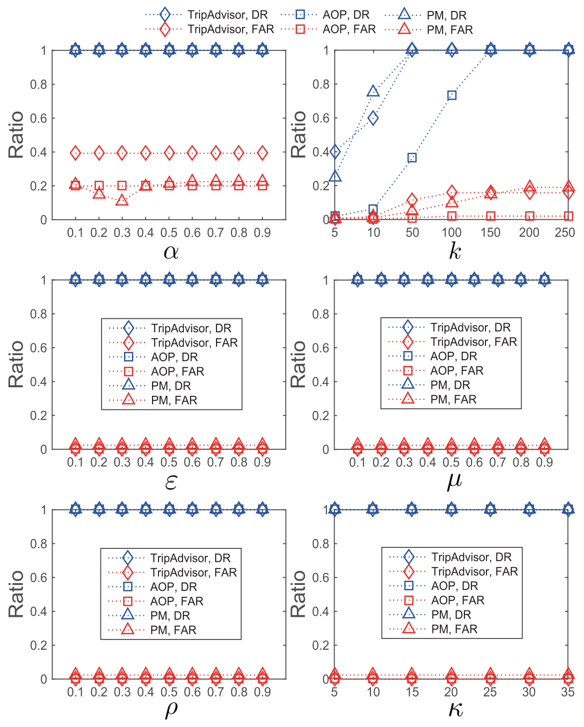


Fig. 8. Sensitivity analysis for crucial model parameters of TBOS, where the attack size is 11.7% in AOP attack, the attack size is 0.8% on the TripAdvisor dataset for optimal-injection attacks, and the attack size is 0.5% in PM.

performance is promising when the number of iterations is large enough.

It should be noted that TBOS consists of four stages, i.e., the unified representation of injection behavior, the discrimination of attack target, the implementation of optimal denoising, and identifying malicious injection based on iterative label propagation. Parameters ε, μ, ρ, and κ are used for the last stage of TBOS. In our comparative experiments, Det-DAT and SubS-DAT are used to demonstrate the effectiveness of the detection after the second and third steps, respectively. It can be seen from Table II-III and Figure 6-7 that the detection results of Det-DAT, SubS-DAT, and TBOS are gradually outstanding in almost all cases. Parameters ε, μ, ρ, and κ have little impact on the detection results with the increase of threshold value. This may be due to the fact that the range to be detected is greatly reduced after DAT and optimal denoising, or possibly due to the fact that the optimal elimination of noise data is beneficial to the stable label prediction. The identification of injection behavior is carried out through the behavior prediction model based on an iterative label propagation mechanism. A relatively small amount of disturbed information may have little impact on the detection performance. Based on the above analyses, the thresholds of α, k, ε, μ, ρ, and κ were empirically set as 0.5, 150, 0.5, 0.5, 0.5, and 0.5 in our experiments, respectively. Undoubtedly, our experimental results are only for reference.

D. Discussion and Limitation

According to all experimental results, several insights are worth analyzing and discussing as follows:

First, the effectiveness of the optimal denoising strategy and DAT can be experimentally verified by SubS-DAT and Det-DAT respectively. However, it is difficult to directly evaluate the effectiveness of behaviour label propagation. The final step of TBOS is to predict a label for each selected user using $\mathcal{F}^l$ learned from the single label data, which is supported by both label propagation and label disambiguation. The feasibility of behaviour label propagation is only verified. By comparing the results of SubS-DAT (without the labelling scheme) with that of TBOS, the effectiveness of label prediction for determining malicious injections can be quantitatively confirmed as shown in Table II-III and Figure 6-7. Due to space limitation, specifically evaluating the effectiveness of behaviour label propagation will be concerned in our future work.

Second, despite the effectiveness of TBOS compared with presented baselines, the design of TBOS empirically depends on the acquisition of priori knowledge of known attacks. This work only focuses on the detection of three kinds of injection attacks. It is also hoped that it can provide a reference basis for anomaly detection of real unlabeled data in e-commerce systems. In recent years, nevertheless, representative attacks (e.g., [26]–[28]) against recommender systems have been published one after another. Additionally, Li *et al.* [29] proposed data poisoning attacks (termed *DPoison* attacks) against matrix-factorization-based CFRs. Considering the generation of attack profiles and the unified representation of multiple injection behaviors, this paper does not detect and analyze the *DPoison* attacks. Investigating the detection of multiple types of attacks has always been a big challenge. Therefore, abnormality detection or discovering interesting findings on real data may conservatively refer to similar malicious injection behavior. Naturally, the generalization ability of the proposed detection model is limited. Moreover, facing with other unknown attacks with completely different attack mechanisms, both the effectiveness of DAT and the detection performance of TBOS may be limited. Of course, the defense against new attacks is still our future work.

VI. Conclusion and Future Work

In this paper, we propose TBOS to detect optimal-injection attacks, co-visitation injection attacks, and profile injection attacks. Extensive evaluations on different datasets demonstrate the effectiveness of TBOS compared with competing baselines. We also verify the effectiveness of discovering interesting findings on real data using the presented detection way. Based on the analysis and discussion of the experimental results, we can briefly summarize as follows: (1) DAT based on both global influence evaluation and risk attitude estimation compared with TIA is easier to adapt to different types of attacks. The parameter setting is relatively more flexible (e.g., α); (2) The optimal denoising method based on influence propagation selects the optimal solution space iteratively to obtain potential domination relationships. It is more flexible compared with the traditional denoising method, but it also pays a price for the computational time; and (3) Label propagation and label disambiguation exploit the structural attributes of unlabeled data, which is easy to design a semi-supervised detection model. It is also conducive to weaken the uncertainty of gray injection behavior in the original data.

In our future work, we will explore the detection problem to deal with recent data poisoning attacks [26], [27], [28], [30], [31], and adversarial attacks [32] against recommender systems. Naturally, facing with diverse threats, how to design a unified detection framework to defend these threats is still a very challenging issue. Additionally, investigating abnormality forensics for real data is also necessary to be concerned.

References

[1] X. Luo, M. Zhou, S. Li, and M. Shang, "An inherently nonnegative latent factor model for high-dimensional and sparse matrices from industrial applications," *IEEE Trans. Ind. Informat.*, vol. 14, no. 5, pp. 2011–2022, May 2018.

[2] M. Fang, G. Yang, N. Z. Gong, and J. Liu, "Poisoning attacks to graph-based recommender systems," in *Proc. 34th Annu. Comput. Secur. Appl. Conf. (ACSAC)*, Dec. 2018, pp. 381–392.

[3] R. Burke, B. Mobasher, C. Williams, and R. Bhaumik, "Classification features for attack detection in collaborative recommender systems," in *Proc. Int. Conf. Knowl. Discovery Data Mining*, 2006, pp. 17–20.

[4] G. Yang, N. Z. Gong, and Y. Cai, "Fake co-visitation injection attacks to recommender systems," in *Proc. Netw. Distrib. Syst. Secur. Symp. (NDSS)*, 2017, pp. 1–15.

[5] J. A. Calandrino, A. Kilzer, A. Narayanan, E. W. Felten, and V. Shmatikov, "'You might also like': Privacy risks of collaborative filtering," in *Proc. IEEE Symp. Secur. Privacy (SP)*, May 2011, pp. 231–246.

[6] X. Xing, W. Meng, D. Doozan, A. Snoeren, N. Feamster, and W. Lee, "Take this personally: Pollution attacks on personalized services," in *Proc. USENIX Security*, 2013, pp. 671–686.

[7] Y. Zhang, Y. Tan, M. Zhang, Y. Liu, T. Chua, and S. Ma, "Catch the black sheep: Unified framework for shilling attack detection based on fraudulent action propagation," in *Proc. 24th Int. Joint Conf. Artif. Intell. (IJCAI)*, 2015, pp. 2408–2414.

[8] I. Gunes, C. Kaleli, A. Bilge, and H. Polat, "Shilling attacks against recommender systems: A comprehensive survey," *Artif. Intell. Rev.*, vol. 42, no. 4, pp. 767–799, 2014.

[9] Z. Yang, Q. Sun, and Y. Zhang, "Probabilistic inference and trustworthiness evaluation of associative links toward malicious attack detection for online recommendations," *IEEE Trans. Depend. Sec. Comput.*, early access, Sep. 10, 2020, doi: 10.1109/TDSC.2020.3023114.

[10] Z. Yang, Q. Sun, Y. Zhang, and W. Wang, "Identification of malicious injection attacks in dense rating and co-visitation behaviors," *IEEE Trans. Inf. Forensics Security*, vol. 16, pp. 537–552, 2021.

[11] W. Zhou, Y. S. Koh, J. Wen, S. Alam, and G. Dobbie, "Detection of abnormal profiles on group attacks in recommender systems," in *Proc. 37th Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, vol. 1, 2014, pp. 955–958.

[12] N. Hurley, Z. Cheng, and M. Zhang, "Statistical attack detection," in *Proc. ACM Conf. Recommender Syst.*, 2009, pp. 149–156.

[13] C. E. Seminario, "Accuracy and robustness impacts of power user attacks on collaborative recommender systems," in *Proc. 7th ACM Conf. Recommender Syst.*, Oct. 2013, pp. 447–450.

[14] C. E. Seminario and D. C. Wilson, "Attacking item-based recommender systems with power items," in *Proc. 8th ACM Conf. Recommender Syst.*, 2014, pp. 57–64.

[15] L. Liu, J. Tang, J. Han, M. Jiang, and S. Yang, "Mining topic-level influence in heterogeneous networks," in *Proc. 19th ACM Int. Conf. Inf. Knowl. Manage.*, 2010, pp. 199–208.

[16] L. Liu, J. Tang, J. Han, and S. Yang, "Learning influence from heterogeneous social networks," *Data Mining Knowl. Discovery*, vol. 25, no. 3, pp. 511–544, Nov. 2012.

[17] Y. Ge, S. Xu, S. Liu, Z. Fu, F. Sun, and Y. Zhang, "Learning personalized risk preferences for recommendation," in *Proc. 43rd Int. ACM SIGIR Conf. Res. Develop. Inf. Retr. (SIGIR)*, Jul. 2020, pp. 409–418.

[18] D. Kahneman and A. Tversky, "Prospect theory: An analysis of decision under risk," *Econometrica*, vol. 47, no. 2, pp. 263–292, 1979.

[19] Z. Yang, Z. Cai, and X. Guan, "Estimating user behavior toward detecting anomalous ratings in rating systems," *Knowl.-Based Syst.*, vol. 111, pp. 144–158, Nov. 2016.

[20] W. Chen, Y. Wang, and S. Yang, "Efficient influence maximization in social networks," in *Proc. 15th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining (KDD)*, 2009, pp. 199–208.

[21] C. Qian, J. C. Shi, Y. Yu, K. Tang, and Z. H. Zhou, "Subset selection under noise," in *Proc. 31st Int. Conf. Neural Inf. Process. Syst. (NIPS)*, 2017, pp. 3563–3573.

[22] C. Qian, Y. Yu, and Z. Zhou, "Subset selection by Pareto optimization," in *Proc. 28th Conf. Neural Inf. Process. Syst. (NIPS)*, 2015, pp. 1765–1773.

[23] Z.-H. Zhou, "A brief introduction to weakly supervised learning," *Nat. Sci. Rev.*, vol. 5, no. 1, pp. 44–53, 2017.

[24] Q.-W. Wang, Y.-F. Li, and Z.-H. Zhou, "Partial label learning with unlabeled data," in *Proc. 28th Int. Joint Conf. Artif. Intell. (IJCAI)*, Aug. 2019, pp. 3755–3761.

[25] J. Zhang, Y. Dong, Y. Wang, J. Tang, and M. Ding, "ProNE: Fast and scalable network representation learning," in *Proc. 28th Int. Joint Conf. Artif. Intell.*, Aug. 2019, pp. 1–7.

[26] H. Zhang, Y. Li, B. Ding, and J. Gao, "Practical data poisoning attack against next-item recommendation," in *Proc. Web Conf. (WWW)*, Apr. 2020, pp. 2458–2464.

[27] C. Lin, S. Chen, H. Li, Y. Xiao, L. Li, and Q. Yang, "Attacking recommender systems with augmented user profiles," in *Proc. 29th ACM Int. Conf. Inf. Knowl. Manage.*, Oct. 2020, pp. 855–864.

[28] X. Zhang, J. Chen, R. Zhang, C. Wang, and L. Liu, "Attacking recommender systems with plausible profile," *IEEE Trans. Inf. Forensics Security*, vol. 16, pp. 4788–4800, 2021.

[29] B. Li, Y. Wang, A. Singh, and Y. Vorobeychik, "Data poisoning attacks on factorization-based collaborative filtering," in *Proc. 29th Conf. Neural Inf. Process. Syst. (NIPS)*, 2016, pp. 1–13.

[30] M. Fang, N. Z. Gong, and J. Liu, "Influence function based data poisoning attacks to top-N recommender systems," in *Proc. Web Conf. (WWW)*, Apr. 2020, pp. 3019–3025.

[31] J. Song *et al.*, "PoisonRec: An adaptive data poisoning framework for attacking black-box recommender systems," in *Proc. IEEE 36th Int. Conf. Data Eng. (ICDE)*, Apr. 2020, pp. 157–168.

[32] K. Christakopoulou and A. Banerjee, "Adversarial attacks on an oblivious recommender," in *Proc. 13th ACM Conf. Recommender Syst. (RecSys)*, Sep. 2019, pp. 322–330.

Zhihai Yang received the Ph.D. degree in control science and engineering from Xi'an Jiaotong University, in 2016. He is currently an Associate Professor with the School of Computer Science and Engineering, Xi'an University of Technology. His research interests include information security, recommender systems, and data mining.

Qindong Sun received the Ph.D. degree from the School of Electronic and Information Engineering, Xi'an Jiaotong University, China. He is currently a Professor with the School of Cyber Science and Engineering, Xi'an Jiaotong University. His research interests include network security, online social networks, and the Internet of Things.

Zhaoli Liu received the Ph.D. degree in cyberspace security from Xi'an Jiaotong University, in 2019. She is currently with the School of Computer Science and Engineering, Xi'an University of Technology. Her research interests include network security and online social network analysis.