Actionable Objective Optimization for Suspicious Behavior Detection on Large Bipartite Graphs

Tong Zhao, Matthew Malir, Meng Jiang
University of Notre Dame, Notre Dame, Indiana, 46556, USA
{tzhao2, mmalir, mjiang2}@nd.edu

***Abstract*—We have been spotting massive suspicious behaviors on bipartite graph-based applications such as social networks and e-commercial platforms. Existing detection methods estimate the suspiciousness score of source users (e.g., followers, buyers) assuming that the behavioral patterns of suspicious source users (e.g., botnet followers, bully buyers) lead to abnormal high density in the graphs. A serious issue when putting the methods into real use is the false positives – the platforms cannot automatically suspend the source users just based on their suspiciousness scores. In this work, we revisit the problem of suspicious behavior detection from the perspective of the target users (e.g., followees, sellers), and provide them an effective and *actionable* solution using big behavior data analytics. We propose a novel method called Actionable Objective Optimization in which the variables are the target users' decisions rather than source users' scores. Experimental results show that our proposed actionable method consistently outperforms the state-of-the-art methods.**

Keywords*—*suspicious behavior detection, bipartite graph, actionable solution, optimization, bully buyer

I. Introduction

Suspicious behaviors can be spotted everywhere on online applications such as social networks and e-commercial platforms where the behavior data can be represented as large bipartite graphs. These graphs consist of links between source users (e.g., followers, buyers) and target users (e.g., followees, sellers). The target users rely on the incoming links from the source users. At the same time, they could be attacked by fraudsters. Celebrities want to be followed massively so botnets could inflate their popularity and destroy the real followers' impressions. Another scaring group of suspicious source users are bully buyers on eBay and Taobao: with a tiny purchase, a buyer can make a bad review on purpose regardless of the quality of the product and won't change the rating back until gift cards or discounts are given from the seller [1]. Given the "follower-to-followee" or "buyer-to-seller" bipartite graphs, can we find suspicious links to avoid being followed by botnets and prevent sellers from being bullied?

Existing methods estimated the suspiciousness scores of source users assuming that the behaviors of suspicious source users (e.g., botnet follower, bully buyers) lead to abnormal subgraph patterns [2], [3], [4], [5], [6], [7], [8], [9]. Their research problems can be unified as below.

Problem 1 *Given an adjacency matrix of the bipartite graph* $\mathbf{A}$*, find a suspiciousness vector* $\mathbf{u}$*, where* u_i *is the suspiciousness score of the* i*-th source user, by optimizing*

$$\max_{\mathbf{u}} J(\mathbf{A}_{sub}(\mathbf{u})), \tag{1}$$

(a) A Taobao plugin that blocks buyers who have "low" ARs. Usually the threshold is above 0.9.

(b) A seller explains why a legitimate buyer (AR = 0.85) was blocked to check out items in the cart.

Fig. 1. When the platforms cannot protect sellers: If a buyer's average rating (AR) is lower than the threshold that a seller gives in the bully-blocking plugin, the buyer can put the item in his/her cart but cannot check out.

where $\mathbf{A}_{sub}(\cdot)$ *is the sub-matrix (or called "block") in* $\mathbf{A}$ *representing the behaviors of a subset of users who have high scores,* $J(\cdot)$ *is the suspiciousness metric of the block [6], [8].*

Density-based metrics have been widely used to evaluate the block's suspiciousness [2], [4], [7], [8]. Formally, the metric function is often defined as (1) the sub-matrix's density:

$$J_d(\mathbf{A}_{sub}) = \frac{e}{n_u \times n_v}, \tag{2}$$

where e is the number of existing edges in the block, n_u is the number of highly suspicious source users, n_v is the number of target users who attacked by the suspicious source users; or (2) the leading singular value of this sub-matrix [5]:

$$J_\alpha(\mathbf{A}_{sub}) = \frac{e}{\sqrt{n_u \times n_v}}. \tag{3}$$

However, none of the methods can achieve a perfect F1 score or even a three 9s in real applications. Due to false positives, the platforms cannot take effective action[1] – they do not want to put their reputation on high risk if they suspend accounts only because they have high suspiciousness scores; instead, they have to take a heavy human effort of experts to confirm the suspicious accounts are truly behaving suspicious.

The platforms cannot effectively protect their users with actionable solutions though they have a large amount of data. Fortunately, all the platforms provide blocklist-like functions: a Twitter or Instagram user can stop someone's following; an eBay or Taobao seller can stop a buyer's checking-out. Knowing that individual's power is limited, eBay sellers even built communities to share the names of bully buyers [1]. However, the blocklist cannot catch up with emerging misbehaving source users. So interestingly, the sellers look at the average

[1] You found your important e-mail in Spambox after 30-min search!

rating (AR) a buyer has previously given: they assume that if the AR is lower than a threshold, the buyer is more likely to be a bully. The sellers developed a plugin as in Figure 1(a) to block low-AR buyers; nevertheless, with the limited data that a seller can access, an ad-hoc blocklist or threshold may block too many honest buyers such that the seller's sale will be severely affected. Figure 1(b) shows that an honest buyer could not purchase from a seller because the buyer's AR is 0.852 based on his/her positive (+1) and negative (0) ratings, approximately 4.4 stars in the range from 1 to 5 stars, and the seller's threshold is 0.95, approximately 4.8 stars.

Inspired by those sellers' strategy, in this work, we revisit the problem of suspicious behavior detection. We claim that if an actionable solution is desired, the learning variables should no longer be suspiciousness scores of source users but "blocklist thresholds" of target users so that if a source user's some property cannot be accepted by the threshold, the target user can automatically block the source user's behavior (out-going link on the graph). But why should we learn the thresholds? Different target users may have different extents of being attacked: most of the cases, they were not attacked at all; for some of them, they might have been attacked frequently. The need is that we would like to find an optimal threshold for each target user so that (1) the target user can avoid being attacked by suspicious source users and (2) they can gain what they want (e.g., popularity or sales) as much as possible from honest source users.

The platforms have massive information-rich data but cannot take action; individuals can take actions but have very limited knowledge to find a proper threshold. In this paper, we will bridge the gap by optimizing individual's actions with the platform's big data. We propose a novel method called *Actionable Objective Optimization* (AOO) to find actionable knowledge for detecting suspicious behaviors.

First, AOO learns an effective threshold for each target user, such that a source user whose property (e.g., average rating) is below this threshold will be blocked by the target user. Again, instead of source user's suspiciousness scores, the variables in our AOO are seller's blocklist thresholds: the idea of making a blocklist threshold has been proved to be actionable in the plugins. The new research problem becomes:

Problem 2 *Given the matrix* $\mathbf{A}$*, find the optimal blocklist threshold vector* $\mathbf{v}$*, where* v_j *is the threshold given by the* j*-th target user, by optimizing:*

$$\max_{\mathbf{v}} J(\mathbf{A}_{sub}(\mathbf{v})), \tag{4}$$

where (1) if $u_i < v_j$*, which means the* i*-th source user's property (e.g., average rating)* u_i *is below the* j*-th target user 's blocklist threshold* v_j*, the behavior entry* A_{ij} *is blocked; (2)* $\mathbf{A}_{sub}(\cdot)$ *is the sub-matrix defined by* $\mathbf{v}$*.*

Second, AOO adopts the two objectives, the sub-matrix's density in Eq.(2) and the sub-matrix's singular value in Eq.(3). These objectives have been demonstrated to be effective with theoretical guarantee [5], [8]. When the variables are changed from the source user's $\mathbf{u}$ to the target user's $\mathbf{v}$, the optimization process becomes complicated, as shown in Section III.

We conducted experiments on both synthetic data and real-world data. Results show that the proposed Actionable Objective Optimization outperforms existing methods on detecting suspicious behaviors. If the property is average rating, sellers will no longer have risk of being attacked by potentially suspicious buyers who have given low average ratings to other sellers before; if the sellers are more cautious, say, they set up another threshold on the number of ratings, newly registered users would probably be considered as potential suspiciousness. The proposed method is general to deal with any kind of source user's properties though we only give a concrete design for the average rating in this paper. Our method can be transferred to the cyber-security domain. For example, given network visit data, AOO can optimize the security level of each individual computer's firewall to block network attacks from botnets; given mobile app data, AOO can prevent suspicious apps from having access to camera, photos, contact, etc., with optimal settings.

We summarize our main contributions as follows.

- We revisit the problem of suspicious behavior detection. We point out the importance of bridging individual's actions with platform's big data for actionable solutions to suspicious behavior detection.
- We propose a novel Actionable Objective Optimization method to block suspicious source users by learning optimal actionable variables (i.e., blocklist thresholds) with large bipartite graph data.
- Experiments demonstrate that the proposed method offers effective and actionable knowledge that individuals can easily use to make decisions.

The rest of this paper is organized as follows. Section II reviews related work on suspicious behavior detection. Section III defines actionable variables and presents the Actionable Objective Optimization method. Experimental results are given in Section IV following with discussion in Section V. Section VI concludes the paper.

II. Related Work

Data-driven approaches have received great success in the field of suspicious behavior detection [8], [10], [11] and behavior modeling [12]: these methods identify unexpectedly dense regions of the bipartite graph, which are potentially harder to evade than review text-based methods on buyer-seller relationships, as creating fake reviews/ratings unavoidably generates edges in the graph [13], [14], [15], [16], [17]. However, platforms can hardly take actions like suspending all positive (suspicious) accounts given by the methods due to false positives. Our proposed work is unique at the point that we suggest individual target users to make actions themselves – blocking suspicious source users with a blocklist threshold. We optimize blocklist settings for the target users instead of a set of predictions. This will easily lead to practice for individuals and hold exemption for the platform. Here we review and compare with suspicious behavior detection methods that may be classified by their suspiciousness assumptions.

Unexpected spectral patterns. Global graph mining methods model the entire graph to find fraud based on singular value decomposition (SVD), latent factor models and belief propagation (BP). SpokEn [2] considered the "spokes" pattern produced by pairs of eigenvectors of graphs, and was later

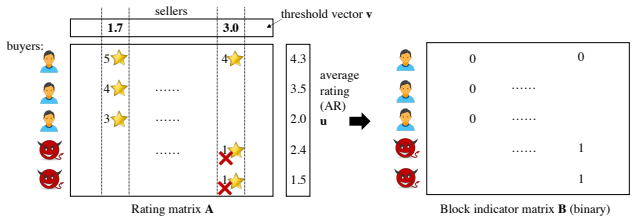


Fig. 2. From rating matrix $\mathbf{A}$, to buyer's average rating vector $\mathbf{u}$, to (learn) seller's threshold vector $\mathbf{v}$, and to have the rating-blocked indicator matrix $\mathbf{B}$.

generalized for fraud detection. FBOX [5] focuses on mini-scale attacks missed by spectral techniques. BP has been used for fraud classification on eBay [18], link farming on Twitter [19], and fake software review detection [20]. All of these methods have been successful in finding fraud but none of them is perfect: attacks can easily evade the detection methods ("camouflage") and honest users may be predicted as fraudsters. For example, FBOX [5] is able to discern suspicious accounts with 0.93 precision; LOCKINFER gives 0.95 accuracy on synthetic datasets; CATCHSYNC [6] generates 0.96 accuracy with no camouflage and 0.79 when there is camouflage, and its accuracy is 0.81 in catching zombie followers on Twitter. Moreover, these were evaluated on only a small portion of labeled ground-truth data. We argue that they are far from being applied onto a real platform requiring professionals' investigation on user identity. Our proposed idea is to prevent users from suspicious behaviors by themselves: in order to effectively block the suspicious behaviors, we recommend optimal blocklist settings.

Unexpected high density in subgraphs. Finding dense subgraphs has been studied from a wide array of perspectives such as mining frequent subgraph patterns [21], [22], detecting communities [23], [24], [25], and finding quasi-cliques [26], [27], [28], [29]. [30] shows that the average degree of subgraph can be maximized with approximation guarantees. [31] optimizes the density of adjacency matrix of subgraph with quality guarantees. [8] adopts both node degree and edge density to model suspiciousness of subgraph and further increases accuracy in binary adjacency matrix of bipartite graph. Our method can work on weighted adjacency matrix of "buyer-to-seller" bipartite graphs which differs from these methods in the data setting. Also, we optimize actionable blocklists to block suspicious behaviors.

Unexpected high density in time-series. Typically there are two kinds of representation on density in time-series. One is dense subgraphs in evolving graphs [32]. COPYCATCH [4] uses local search heuristics to find Δt-bipartite cores in which users consistently likes the same Facebook pages at the same short time interval. The other is dense subtensors in high-order tensors of a time dimension [33], [7], [34] or tensor streams [9]. [35] considers fraud detection methods that are robust to camouflage attacks. [36] adopts a Bayesian model to find early spikes of outlier ratings in time series. All these methods focus on the time-series domain, observing changes in the behavior from system access logs rather than graph data.

III. ACTIONABLE OBJECTIVE OPTIMIZATION

In this section, we will introduce our new optimization method. In order to present the method easily for understanding, we use the language of the bully-buyer scenarios (e.g., buyers, sellers, bullies) instead of the language of general graph data (e.g., source users, target users, suspicious source users).

TABLE I. SYMBOLS AND THEIR DESCRIPTIONS.

Symbol	Description
n	The number of buyers
m	The number of sellers
$\mathbf{A} \in [0,1]^{n\times m}$	Rating score matrix
$\mathbf{I} \in \{0,1\}^{n\times m}$	Rating indicator matrix
$\mathbf{u} \in [0,1]^n$	Buyer's average ratings
$\mathbf{v} \in [0,1]^m$	Seller's blocklist thresholds
$\mathbf{B} \in \{0,1\}^{n\times m}$	Rating-blocked indicator matrix
$\mathbf{c}^{(u)} \in \mathbb{N}^n$	Count of a buyer's ratings being blocked
$\mathbf{c}^{(v)} \in \mathbb{N}^m$	Count of ratings a seller blocks
$\mathbf{s}^{(u)} \in \{0,1\}^n$	Bully buyer indicator vector
$\mathbf{s}^{(v)} \in \{0,1\}^m$	Bullied seller indicator vector
$\beta^{(u)}$	The minimum count of a bully buyer's ratings being blocked
$\beta^{(v)}$	The minimum count of ratings that a bullied seller blocks
$\mathbf{A}_{sub} \in [0,1]^{n_u\times n_v}$	Suspicious rating sub-matrix
n_u	Number of bully buyers predicted by collective decisions
n_v	Number of bullied sellers predicted by collective decisions
e	Number of valid ratings in $\mathbf{A}_{sub}$
$\mathbf{1}_n \in \{1\}^n$	All-one vector of size n
$g(\cdot)$	The logistic function
$d(\cdot)$	Density of a matrix
$\sigma(\cdot)$	Leading singular value of a matrix
$J(\cdot)$	The objective function

Suppose we have n buyers and m sellers. We denote $\mathbf{A} \in [0,1]^{n\times m}$ as the rating-data matrix in which A_{ij} is the rating score given by the i-th buyer to the j-th seller, where $i = 1,2,\ldots,n$ and $j = 1,2,\ldots,m$. We assume that the rating score from a buyer to a seller has been normalized into a range from 0 to 1: For a rating x from 1-star to 5-stars, it can be min-max normalized as $(x-1)/4$. $\mathbf{A}$ is a sparse matrix because most of the entries are unavailable. To avoid the confusion between rating scores of 0 and unavailable entries, we have another sparse matrix $\mathbf{I} \in \{0,1\}^{n\times m}$ as indicator in which I_{ij} is 1 if the rating is available and 0 if not.

A. Actionable Variables and Objectives

In order to solve Problem 2, our idea is to define actionable variables and optimize effective objectives. Here every seller can investigate the average rating of each buyer who proposes to buy some product from the seller. If the average rating is below a threshold, the seller assumes that the buyer is more likely to be a bully and the seller can use the blocklist feature to stop the buyer's purchase. This "blocklist threshold" is actionable. We denote $\mathbf{v} \in [0,1]^m$ as the vector of blocklist threshold values. And we denote $\mathbf{u} \in [0,1]^n$ as the vector of buyer's average rating scores:

$$u_i = \frac{\sum_{j=1}^{m} I_{ij} A_{ij}}{\sum_{j=1}^{m} I_{ij}}. \tag{5}$$

We define $\mathbf{B} \in \{0,1\}^{n\times m}$ as the rating-blocked indicator matrix, in which $B_{ij} = 1$ if the rating A_{ij} is blocked and $B_{ij} = 0$ otherwise. Formally, the entries in $\mathbf{B}$ are defined as

$$B_{ij} = \begin{cases} 1, & \text{if } I_{ij} = 1 \text{ and } u_i < v_j; \\ 0, & \text{otherwise.} \end{cases} \tag{6}$$

Figure 2 shows how we have the matrix $\mathbf{B}$. Basically we expect the sellers who were bullied to have a sensitive (high) threshold (like the 3.0 in the figure, which means if the buyer's AR is below 3.0, his/her rating to the seller would be blocked)

and expect unbullied sellers to have low thresholds (i.e., 1.7) for not losing honest buyer's ratings.

We denote $\mathbf{c}^{(u)} \in \mathbb{N}^n$ as the vector of the count of buyer's ratings that are blocked; and we denote $\mathbf{c}^{(v)} \in \mathbb{N}^m$ as the vector of the count of ratings a seller blocks. Formally, the vectors can be defined as follows:

$$\mathbf{c}^{(u)} = \mathbf{B} \cdot \mathbf{1}_m, \quad (7)$$
$$\mathbf{c}^{(v)} = \mathbf{B}^{\mathrm{T}} \cdot \mathbf{1}_n. \quad (8)$$

We denote $\mathbf{s}^{(u)} \in \{0,1\}^n$ as the binary vector of a buyer being a bully; and we denote $\mathbf{s}^{(v)} \in \{0,1\}^m$ as the binary vector of a seller being bullied. Formally, the i-th entry in $\mathbf{s}^{(u)}$ and the j-th entry in $\mathbf{s}^{(v)}$ are defined as follows:

$$s_i^{(u)} = \begin{cases} 1, & \text{if } c_i^{(u)} \geq \beta^{(u)}; \\ 0, & \text{otherwise.} \end{cases} \quad (9)$$

$$s_j^{(v)} = \begin{cases} 1, & \text{if } c_j^{(v)} \geq \beta^{(v)}; \\ 0, & \text{otherwise.} \end{cases} \quad (10)$$

where $\beta^{(u)}$ is the minimum count of a bully buyer's ratings being blocked, and $\beta^{(v)}$ is the minimum count of ratings that a bullied seller blocks. We assume the minimum counts $\beta^{(u)}$ and $\beta^{(v)}$ are given.

The number of buyers that are predicted to be bullies from the collective decisions is

$$n_u = \mathbf{1}_n^{\mathrm{T}} \cdot \mathbf{s}^{(u)}. \quad (11)$$

The number of sellers that are predicted to be bullied is

$$n_v = \mathbf{1}_m^{\mathrm{T}} \cdot \mathbf{s}^{(v)}. \quad (12)$$

The number of ratings that are given by the set of bully buyers to the set of the bullied sellers is

$$e = {\mathbf{s}^{(u)}}^{\mathrm{T}} \cdot \mathbf{B} \cdot \mathbf{s}^{(v)}. \quad (13)$$

Based on the objectives in Eq.(2) and Eq.(3), we have two variants of actionable objective optimizations:

$$J_d(\mathbf{v}) = \frac{e}{n_u \times n_v} = \frac{{\mathbf{s}^{(u)}}^{\mathrm{T}} \cdot \mathbf{B} \cdot \mathbf{s}^{(v)}}{\big(\mathbf{1}_n^{\mathrm{T}} \cdot \mathbf{s}^{(u)}\big)\big(\mathbf{1}_m^{\mathrm{T}} \cdot \mathbf{s}^{(v)}\big)}, \quad (14)$$

$$J_\alpha(\mathbf{v}) = \frac{e}{{n_u}^{\frac{1}{2}} \times {n_v}^{\frac{1}{2}}} = \frac{{\mathbf{s}^{(u)}}^{\mathrm{T}} \cdot \mathbf{B} \cdot \mathbf{s}^{(v)}}{\big(\mathbf{1}_n^{\mathrm{T}} \cdot \mathbf{s}^{(u)}\big)^{\frac{1}{2}}\big(\mathbf{1}_m^{\mathrm{T}} \cdot \mathbf{s}^{(v)}\big)^{\frac{1}{2}}}. \quad (15)$$

B. Optimization

It is easy to simplify the partial derivatives of J_d and J_α with respect to $\mathbf{v}$ as follows. For each $k \in \{1, 2, \ldots, m\}$,

$$\frac{\partial J_d}{\partial v_k} = \frac{1}{n_u n_v}\frac{\partial e}{\partial v_k} - \frac{e}{{n_u}^2 n_v}\frac{\partial n_u}{\partial v_k} - \frac{e}{n_u {n_v}^2}\frac{\partial n_v}{\partial v_k}, \quad (16)$$

$$\frac{\partial J_\alpha}{\partial v_k} = \frac{1}{{n_u}^{\frac{1}{2}} {n_v}^{\frac{1}{2}}}\frac{\partial e}{\partial v_k} - \frac{e}{2{n_u}^{\frac{3}{2}} {n_v}^{\frac{1}{2}}}\frac{\partial n_u}{\partial v_k} - \frac{e}{2{n_u}^{\frac{1}{2}} {n_v}^{\frac{3}{2}}}\frac{\partial n_v}{\partial v_k}. \quad (17)$$

Then we look for the partial derivatives $\frac{\partial n_u}{\partial v_k}$, $\frac{\partial n_v}{\partial v_k}$ and $\frac{\partial e}{\partial v_k}$.

To use Matrix Calculus to optimize the variables $\mathbf{v}$, we approximate the matrix $\mathbf{B}$ and the vectors $\mathbf{s}^{(u)}$, $\mathbf{s}^{(v)}$ as follows:

$$\mathbf{B} = \mathbf{I} \odot g(\mathbf{1}_n \cdot \mathbf{v}^{\mathrm{T}} - \mathbf{u} \cdot \mathbf{1}_m^{\mathrm{T}}), \quad (18)$$
$$\mathbf{s}^{(u)} = g(\mathbf{c}^{(u)} - \beta^{(u)}\mathbf{1}_n), \quad (19)$$
$$\mathbf{s}^{(v)} = g(\mathbf{c}^{(v)} - \beta^{(v)}\mathbf{1}_m), \quad (20)$$

in which $g(x)$ is the logistic function:

$$g(x) = \frac{1}{1 + e^{-\alpha x}}, \quad (21)$$

where α is the steepness of the "S"-shape curve. We will use the derivative of the logistic function:

$$\frac{\partial g(x)}{\partial x} = \alpha g(x) g(1 - x). \quad (22)$$

Therefore, n_u, n_v and e can be written as below.

$$n_u = \sum_{i=1}^{n} s_i^{(u)} = \sum_{i=1}^{n} g\Big(\sum_{j=1}^{m} B_{ij} - \beta^{(u)}\Big), \quad (23)$$

$$n_v = \sum_{j=1}^{m} s_j^{(v)} = \sum_{j=1}^{m} g\Big(\sum_{i=1}^{n} B_{ij} - \beta^{(v)}\Big), \quad (24)$$

$$e = \sum_{i=1}^{n}\sum_{j=1}^{m} s_i^{(u)} B_{ij} s_j^{(v)}. \quad (25)$$

Then we have

$$\frac{\partial n_u}{\partial B_{ij}} = \alpha g\Big(\sum_{j=1}^{m} B_{ij} - \beta^{(u)}\Big)\Big(1 - g\Big(\sum_{j=1}^{m} B_{ij} - \beta^{(u)}\Big)\Big) = \alpha s_i^{(u)}\big(1 - s_i^{(u)}\big), \quad (26)$$

$$\frac{\partial n_v}{\partial B_{ij}} = \alpha g\Big(\sum_{i=1}^{n} B_{ij} - \beta^{(v)}\Big)\Big(1 - g\Big(\sum_{i=1}^{n} B_{ij} - \beta^{(v)}\Big)\Big) = \alpha s_j^{(v)}\big(1 - s_j^{(v)}\big), \quad (27)$$

$$\frac{\partial e}{\partial B_{ij}} = \alpha s_i^{(u)}(1 - s_i^{(u)}) \sum_{q=1}^{m} B_{iq} s_q^{(v)} + \alpha s_j^{(v)}(1 - s_j^{(v)}) \sum_{p=1}^{n} B_{pj} s_p^{(u)} + s_i^{(u)} s_j^{(v)}. \quad (28)$$

Given

$$B_{ij} = I_{ij} g(v_j - u_i), \quad (29)$$

we obtain the partial derivative of B_{ij} with respect to v_k:

$$\frac{\partial B_{ij}}{\partial v_k} = \begin{cases} \alpha B_{ik}\big(1 - g(v_k - u_i)\big), & \text{if } k = j; \\ 0, & \text{otherwise.} \end{cases} \quad (30)$$

Now we can have the partial derivatives as below.

$$\frac{\partial n_u}{\partial v_k} = \sum_{i=1}^{n}\sum_{j=1}^{m} \frac{\partial n_u}{\partial B_{ij}} \cdot \frac{\partial B_{ij}}{\partial v_k} = \alpha^2 \sum_{i=1}^{n} s_i^{(u)}\big(1 - s_i^{(u)}\big) B_{ik}\big(1 - g(v_k - u_i)\big) \quad (31)$$

$$\frac{\partial n_v}{\partial v_k} = \sum_{i=1}^{n}\sum_{j=1}^{m} \frac{\partial n_v}{\partial B_{ij}} \cdot \frac{\partial B_{ij}}{\partial v_k} = \alpha^2 s_k^{(v)}\big(1 - s_k^{(v)}\big) \sum_{i=1}^{n} B_{ik}\big(1 - g(v_k - u_i)\big) \quad (32)$$

$$\frac{\partial e}{\partial v_k} = \sum_{i=1}^{n}\sum_{j=1}^{m}\frac{\partial e}{\partial B_{ij}}\cdot\frac{\partial B_{ij}}{\partial v_k}$$
$$= \alpha\sum_{i=1}^{n} s_i^{(u)}\big(1-s_i^{(u)}\big)\sum_{q=1}^{m} B_{iq}s_q^{(v)} \quad (33)$$
$$+n\cdot\alpha s_k^{(v)}\big(1-s_k^{(v)}\big)\sum_{p=1}^{n} B_{pk}s_p^{(u)} + s_k^{(v)}\sum_{i=1}^{n} s_i^{(u)}.$$

We learn the threshold vector $\mathbf{v}$ by multiple iterations of optimization. In real practice, we observe that the block density (J_d) works much better than the leading singular value (J_α) as the metric function. The reason is that the bully attacks create dense blocks but not necessarily high degrees from each attacker. Therefore, we only report the performance of our optimization method based on the density metric.

C. Complexity Analysis

Here we analyze the computational complexity step by step. From Eq.(18) we know the complexity of computing $\mathbf{B}$ is $O(nm)$, where n and m are the numbers of buyers and sellers, respectively. We are aware of the high sparsity of $\mathbf{B}$: the density of $\mathbf{B}$ is not bigger than the density of the rating matrix $\mathbf{A}$. If we denote n_r as the number of ratings in the data, from Eq.(31), Eq.(32), and Eq.(33), we know the complexities of computing $\frac{\partial n_u}{\partial \mathbf{v}}$, $\frac{\partial n_v}{\partial \mathbf{v}}$, and $\frac{\partial e}{\partial \mathbf{v}}$ are $O(n_r)$, $O(n_r)$, and $O(mn_r+n_r)$, respectively. Therefore, if we combine all the above together, the total complexity is $O((mn_r+mn+3n_r)t)$, that can be simplified as the *quadratic* time $O(mn_rt)$, where t is the number of iterations. The algorithm is applicable for big real data in this era of high computability.

IV. Experiments

In this section, we conduct experiments on both synthetic data and real-world data sets to answer the following questions:

- **Q1. (Actionable effectiveness)** Does the proposed AOO method provide effective, actionable solutions? How does it perform compared with the state-of-the-art under different settings in terms of whole graph density, rating score distribution, and number of attack source user groups?
- **Q2. (Discovery of suspiciousness)** What can AOO detect when applied to a public Amazon product review dataset?
- **Q3. (Efficiency)** Does AOO work efficiently on a large size of data? How does the time cost increase when the dataset goes bigger?

A. Experiments on Actionable Effectiveness

1) Experimental settings: Here we present synthetic data, baseline methods and their invariants, and parameter settings we use in the experiments.

Synthetic datasets. We generate a random rating matrix of n honest buyers, m sellers, and density d. The rating scores are generated by a Gaussian distribution $\mathcal{N}(\mu,\sigma^2)$. The default setting is $n=2,000$, $m=2,000$, $d=0.01$, $\mu=4$, and $\sigma=0.5$. So the mean value of the number of ratings a buyer gives is $m\times d=20$. Note that when the scores were min-max normalized into the range $[0,1]$, the mean value ($\mu=4$) becomes 0.8. Then we inject n_g bully-group attacks into the synthetic data. Each group attack creates a block (i.e., matrix attached to the former one) of n_b bully buyers and m_b sellers. The number of attacks (ratings) given by a bully buyer follows a Gaussian distribution $\mathcal{N}(\mu_a,{\sigma_a}^2)$. The bully's rating scores are generated by a Gaussian distribution $\mathcal{N}(\mu_b,{\sigma_b}^2)$. We will investigate method performances under different settings:

- Bully group number: $n_g=10$ in default; we will investigate $n_g\in\{1,2,\ldots,20\}$;
- Bully density: $\mu_a=20$ and $\sigma_a=3$ in default; we will investigate $\mu_a\in\{5,10,15,20,25,30,35\}$;
- Bully greediness: $\mu_b=1$ (1-star) and $\sigma_b=0.5$ in default; we will investigate $\mu_b\in[1,1.3]$. What does $\mu_b=1.3$ mean? Among 10 bully attacks as giving low rating scores, the bullies can have 1 normal score (4-stars) as camouflage. Then the average rating score is 1.3.

Evaluation metrics. We evaluate the performance of our proposed method and all baseline methods on *accuracy*, *precision*, *recall* and *F1 score* based on the standard confusion matrix. Positives are the ratings given by injected bully buyers and negatives are the ratings by honest buyers. Accuracy is calculated as the number of true positives (TPs) and true negatives (TNs) divided by the total number of ratings. Precision is the number of TPs by the positive predictions and recall is the number of TPs by the ground-truth positives. F1 score is the harmonic mean of precision and recall. It tends to be close to the smaller one. For all of the four metrics, a higher score indicates a better performance.

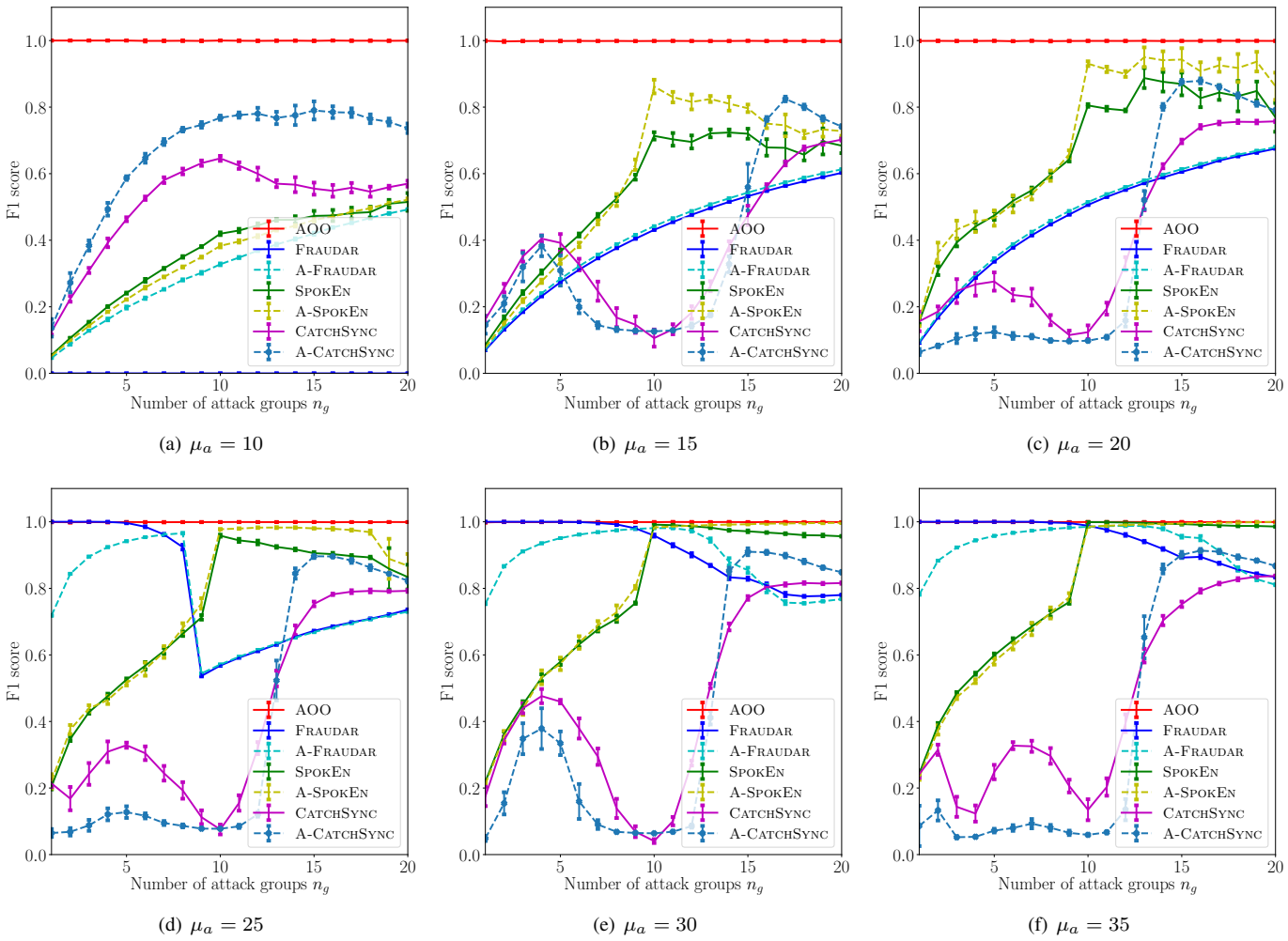
Baseline methods. We compare our proposed method with three state-of-the-art methods and their "actionable" versions:

- SpokEn [2]: it uses eigenvectors of the rating matrix to find anomalous patterns in spectral subspaces;
- CatchSync [6]: it captures the synchronized behavior patterns of bad actors in social networks and rating behaviors. This work was selected as "Bests of SIGKDD 2014" and invited to ACM TKDD 2016;
- Fraudar [8]: it provides theoretical bounds to catching fraudsters in matrix-shape behavior data. This work won ACM SIGKDD 2016 Best Research Paper Award.

Though all the above methods have been well recognized, none of them provides actionable solutions to protect individual users. Therefore, based on their results, we implement an "actionable version" for each of them. The idea is to learn an optimized seller's threshold vector $\mathbf{v}$ via maximizing the accuracy of blocking bully buyers that were detected by the method. Thus, we have three *actionable* baselines A-SpokEn, A-CatchSync, and A-Fraudar.

AOO initialization. We initialize the optimization in AOO with a threshold vector $\mathbf{v}$ generated from $\mathcal{N}(\mu_v,{\sigma_v}^2)$, where $\sigma_v=0.1$, μ_v will be set as 1 and discussed in $[1,1.5]$.

2) Evaluation on overall effectiveness: Table II presents the performance of Actionable Objective Optimization (AOO) method and all baselines for different bully densities $\mu_a\in\{10,20,25,30\}$. AOO consistently achieves almost perfect performance: it shows at least .999 on accuracy and .998 on F1 score for all cases. The standard deviation is very small (smaller than .001) which means AOO has stable performances. The too perfect performance is potentially caused



(a) $\mu_a = 10$ (b) $\mu_a = 15$ (c) $\mu_a = 20$

(d) $\mu_a = 25$ (e) $\mu_a = 30$ (f) $\mu_a = 35$

Fig. 3. Our AOO model consistently outperforms baselines with almost perfect F1 scores for different bully group numbers n_g and bully densities μ_a. The baseline methods include both the original state-of-the-art but also their "actionable" versions we created.

by the imperfect simulation of real-life ratings by Gaussian distribution.

The baseline methods have different behaviors in different settings. First of all, SPOKEN performs well though not perfect (F1 scores of .958 and .992) when the bully buyers generate dense attacks, say, the average number of low ratings given from each bully is at least 25. However, when the bully density μ_a is lower (i.e., 20, 15), the F1 score drops significantly to .805 and .420. The reason is that eigenvectors would not be able to capture the strange patterns when the bad actors perform the same or even less extensive connections to the sellers. Interestingly, the "actionable" version A-SPOKEN generates higher precision than SPOKEN when μ_a is 20 or 25 because the thresholding would not mix honest buyers and bullies when they generated a similar number of ratings.

Second, CATCHSYNC performs poorly when the attacks become denser: The feature of synchronicity is defined as the similarity of sellers the buyers give ratings to, however, when the bullies attack more kinds of sellers, the synchronicity will become less significant. This is complementary with the detection methods that only focus on isolated dense blocks.

Third, FRAUDAR shows very similar behaviors as SPOKEN does. It performs very well with F1 score of .958 when the bully density μ_a is as high as 30; and the actionable version A-FRAUDAR generates an even high F1 score, .981. Unfortunately, it doesn't perform better than SPOKEN when the bully density is smaller. The reason is that FRAUDAR considers the contrast between density of normal user's ratings and density of bully buyer's ratings. When the background density is 20 and the bully density is smaller than or close to it, FRAUDAR could not catch the suspicious behaviors.

Overall, our proposed AOO method has three advantages so that it outperforms the three baselines and their actionable versions. First, our method considers not only the rating behaviors but also the rating scores. It calculated buyer's average rating as the explicit clue for blocking bullies. Second, our method learns a vector of seller's thresholds by jointly utilizing the decisions of all the sellers. More importantly, the threshold vector enables an actionable solution to prevent the buyers from being bullied. Third, our method optimizes (maximizes) the block density by updating seller's thresholds. A higher block density indicates more suspicious group attacks.

3) Effectiveness evaluation on bully group number (number of attack groups) n_g: Figure 3 presents the performance of each method on different numbers of bully groups. Each sub-

TABLE II. OUR AOO MODEL OUTPERFORMS THE AWARD-WINNING WORK FRAUDAR [8], OTHER BASELINES, AND THEIR ACTIONABLE VERSIONS ON SYNTHETIC DATASETS OF DIFFERENT VALUES OF μ_a.

	Accuracy	Precision	Recall	F1 score
	Mean value of #attacks per bully $\mu_a = 10$			
SPOKEN	.4466 ± .0093	.2655 ± .0033	1. ± 0.	.4196 ± .0041
A-SPOKEN	.3557 ± .0128	.2369 ± .0036	1. ± 0.	.3831 ± .0047
CATCHSYNC	.8556 ± .0029	.6338 ± .0072	.6586 ± .0093	.6460 ± .0075
A-CATCHSYNC	.9153 ± .0020	.8490 ± .0042	.7014 ± .0096	.7682 ± .0065
FRAUDAR	.0042 ± .0005	0. ± 0.	0. ± 0.	0. ± 0.
A-FRAUDAR	.2420 ± .0008	.1991 ± .0015	.9234 ± .0112	.3276 ± .0027
AOO	**1. ± 0.**	1 ± 0.	1. ± 0.	**1. ± 0.**
	Mean value of #attacks per bully $\mu_a = 20$			
SPOKEN	.8385 ± .0042	.6737 ± .0057	1. ± 0.	.8050 ± .0041
A-SPOKEN	.9502 ± .0055	.8701 ± .0126	1. ± 0.	.9305 ± .0072
CATCHSYNC	.6825 ± .0038	.7756 ± .0655	.0674 ± .0118	.1237 ± .0204
A-CATCHSYNC	.6673 ± .0004	.5095 ± .0049	.0542 ± .0013	.0980 ± .0021
FRAUDAR	.3493 ± .0021	.3388 ± .0007	1. ± 0.	.5061 ± .0008
A-FRAUDAR	.3478 ± .0003	.3224 ± .0001	1. ± 0.	.4876 ± .0001
AOO	**.9993 ± .0005**	.9998 ± .0004	.9980 ± .0012	**.9989 ± .0007**
	Mean value of #attacks per bully $\mu_a = 25$			
SPOKEN	.9665 ± .0027	.9199 ± .0060	1. ± 0.	.9583 ± .0032
A-SPOKEN	.9824 ± .0006	.9564 ± .0013	1. ± 0.	.9777 ± .0007
CATCHSYNC	.6272 ± .0029	.8140 ± .0354	.0396 ± .0080	.0754 ± .0149
A-CATCHSYNC	.6165 ± .0004	.5040 ± .0063	.0420 ± .0011	.0776 ± .0019
FRAUDAR	.4147 ± .0024	.3966 ± .0010	1. ± 0.	.5679 ± .0010
A-FRAUDAR	.4226 ± .0009	.3998 ± .0004	1. ± 0.	.5712 ± .0004
AOO	**.9994 ± .0002**	1. ± 0.	.9985 ± .0006	**.9993 ± .0003**
	Mean value of #attacks per bully $\mu_a = 30$			
SPOKEN	.9926 ± .0017	.9831 ± .0038	1. ± 0.	.9915 ± .0019
A-SPOKEN	.9852 ± .0001	.9665 ± .0002	1. ± 0.	.9830 ± .0001
CATCHSYNC	.5785 ± .0009	.8396 ± .0500	.0208 ± .0031	.0406 ± .0058
A-CATCHSYNC	.5712 ± .0002	.4961 ± .0036	.0341 ± .0005	.0638 ± .0008
FRAUDAR	.9621 ± .0024	.9188 ± .0047	1. ± 0.	.9577 ± .0025
A-FRAUDAR	.9832 ± .3447	.9624 ± .0007	1. ± 0.	.9808 ± .3868
AOO	**.9992 ± .0006**	1. ± 0.	.9980 ± .0013	**.9990 ± .0007**

figure presents for a specific bully density μ_a. (1) Figure 3(a) shows that when the bullies perform sparse attacks ($\mu_a = 10$), our AOO method can still catch their bullying behaviors because the thresholding strategy is actionable and insensitive to the attack density. Other baselines that rely on the density signal cannot perform well. When the number of attack groups becomes bigger, their performances become better because more groups, though sparse, make the signal stronger and thus it becomes easier to detect the bullies. (2) Figure 3(b) and 3(c) show that when the attack density becomes close to the normal user's behaviors, CATCHSYNC doesn't work well but SPOKEN performs better. From Figure 3(c) and 3(d), we can see when the number of groups becomes a big number ($n_g \geq 15$), CATCHSYNC performs well because it models the synchronicity of bad actors' grouping behaviors. (3) Figure 3(d), 3(e) and 3(f) show that when the bully density becomes more than the normal user's behaviors ($\mu_a > 20$), FRAUDAR performs well when the number of attack group is small, and SPOKEN performs well when the number of groups is big. The reason is that these two methods follow different heuristics in detecting bad actors. Our AOO method (the red line) can perform consistently almost perfect on F1 score.

The error bars at almost every point for every method are small, which means that the synthetic datasets have consistent properties and all the methods have stable performances.

4) Effectiveness evaluation on bully density μ_a: Figure 4 presents the performance of each method as μ_a changes under default setting ($n_g = 20$). We observe that SPOKEN and FRAUDAR, as well as their actionable versions, can perform well when the attack density μ_a is higher than normal (> 20); CATCHSYNC prefers a bigger number of groups but cannot handle dense attacks. Our AOO method does not require the

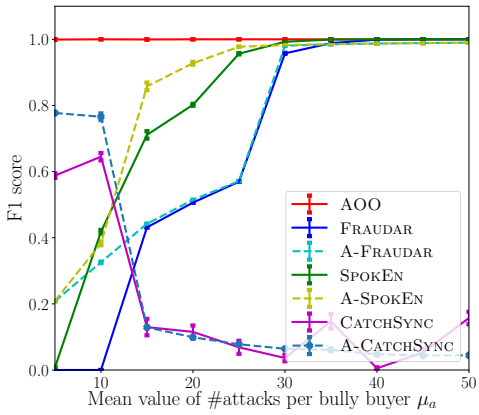


Fig. 4. Our proposed AOO model consistently generates almost-perfect F1 score, outperforming FRAUDAR [8], its actionable version A-FRAUDAR, and other baselines when the bullies do not make extensive attacks.

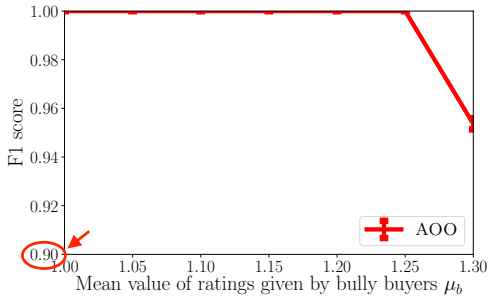


Fig. 5. AOO generates almost perfect performance when the mean value of ratings given by **bully buyers** μ_b is not higher than 1.3.

block of bully behavior to be significantly denser than the data matrix. It blocks suspicious reviews by thresholding the average rating score of buyers. So AOO consistently performs well no matter μ_a is big or small. We compare subfigures in Figure 3 and have the same conclusions as we draw above.

5) Effectiveness evaluation on bully greediness μ_b: Bully buyers give low ratings to bully the sellers. There are different levels of low ratings. For example, Amazon users can give ratings from 1 star to 5 stars. Usually, 1 star and 2 stars are both considered as negative ratings. So it is possible that some bully buyers make a portion of 2-stars negative reviews to evade detection on individual low ratings because, in this way, their average rating is not globally very low. Figure 5 presents AOO's performance under different mean values of ratings given by bully buyers μ_b. AOO generates almost perfect performance when μ_b is not higher than 1.3. The F1 score drops from .99 to .95 when μ_b becomes higher than 1.25 (i.e., a 2-stars rating along with three 1-star ratings).

6) Effectiveness evaluation on normal user's rating μ: Figure 6 presents AOO's performance under different mean values of ratings given by normal buyers. Just opposite to the above discussion on bully buyer's rating scores, if the ratings given by normal buyers are low, it will be challenging for algorithms to distinguish the two groups. Interestingly, even when μ is as small as 2.0, our AOO method can still achieve a F1 score higher than .94. It is worthwhile to note that the average rating of normal users on real-world datasets (like Amazon) is usually higher than 3.5. So we are confident that AOO can perform very well on real applications.

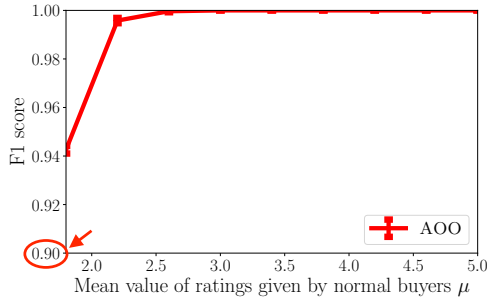
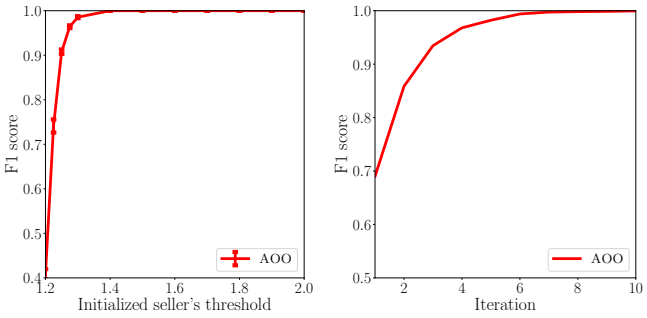


Fig. 6. AOO generates almost perfect performance when the mean value of ratings given by **normal users** μ is not smaller than 2.0.



(a) Insensitive to initialization (b) Convergence after iterations

Fig. 7. AOO's two important properties: insensitivity and convergence.

7) Evaluation on initialization insensitivity and iterative process: Figure 7(a) presents the F1 scores given by the proposed AOO method when we set different initialized values for the seller's threshold vector. When the value is above 1.25-stars, the F1 score is consistently higher than .9; when the value is above 1.4, the F1 score achieves at least .99. There is a huge range of the initialized value, $[1.4, 5.0]$, that can generate almost perfect performances. So we observe that the AOO method is insensitive to the parameter. Figure 7(b) shows how the performance converges to the perfect after a few iterations. Here at the 7^{th} iteration, the F1 score is above .99.

B. Experiments on Suspiciousness Discovery

Amazon data description. We crawled a small product review dataset from Amazon in the year of 2015, while making sure each user/product had made/received at least 20 reviews. The dataset has 4,552 users (considered as "buyers") and 6,347 products (considered as "sellers"). It has 231,600 ratings along with reviews in text. The density of the rating matrix is 0.008.

Figure 8 presents the distribution of the buyer's average rating. The rating scale is from 1-star to 5-stars. The average score of all the ratings is 4.2. From the distribution, we can see that the most frequent bucket is 4–4.5. But we do have a number of users who have relatively very low average ratings. We have 3.4% users whose average rating is below 3. The absolute number of those users is 155.

Unfortunately, we do not have any ground truth or label that indicates bully users in the Amazon data: (1) Most of the bullying behaviors happen on consumer-to-consumer (C2C) business models such as eBay and Taobao. (2) Even for eBay and Taobao, without serious complaints from customers or without careful investigation by experts, we can hardly conclude what ratings or reviews are bullying behaviors and who are the bully buyers. So what can we do right here with

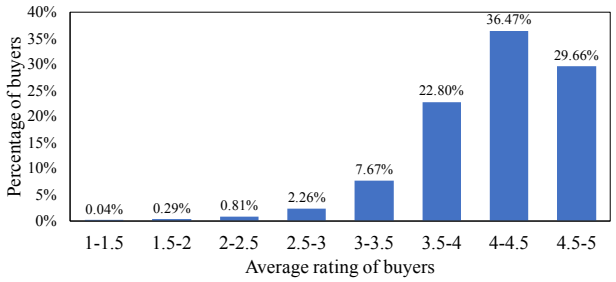


Fig. 8. The distribution of Amazon buyer's average rating (AR).

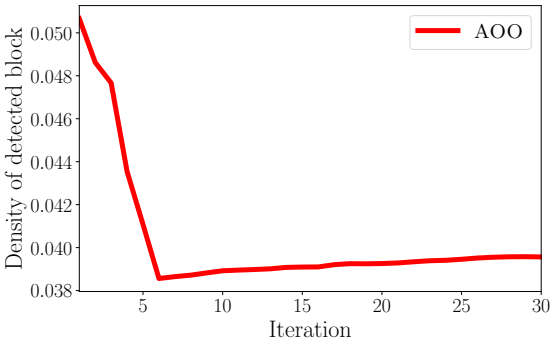


Fig. 9. Convergence on the density of the user-product block detected by the proposed AOO method in Amazon rating data.

this Amazon data? The fortunate thing is we have the text reviews. We do not use the review content in our method. We only use the ratings. Therefore, for the 155 users who give low ratings, it will be interesting to see what is the difference between the content from the ones who would be predicted as *bully-like* users and the ones who would not be. A short conclusion we get from the results (Figure 11) is that bully-like users are more likely to use sarcastic and disrespectful words such as "stupid", "hell", "terrible", and "horrible".

Results. First of all, we find a block of 23 bully-like users among the 155 users who give low ratings. The final density of the user-product block is 0.04. It is higher than the global density but still very low as a sparse block. Here the bully-like users are performing sparse "attacks,", and our AOO captures the behaviors. The average rating score of the block is 1.85.

Secondly, Figure 9 presents the convergence on the density of the user-product block. We observe that the block starts from a small size and high density and then grows to be bigger but a smaller density till the 6^{th} iteration where the density reaches the bottom. The block has included the 23 users at this iterations and then keep updating the threshold vector of sellers to prevent attacks from them. So the density becomes slightly bigger and convergences since the 6^{th} iteration.

Thirdly, Figure 10 presents the convergence on the average rating of the user-product block. We observe that the average rating is extremely low at the early stage (when the block is of a very small size) and then it goes up sharply after a few iterations. The average rating meets an "elbow" at the 6^{th} iteration and then it convergences to be 1.85.

Finally, the most interesting observations come from Figure 11. We plot word clouds of the user's reviews and compare with three categories: (a) all the buyers, (b) the bully-like users, and (c) the users who were not predicted as bullies but

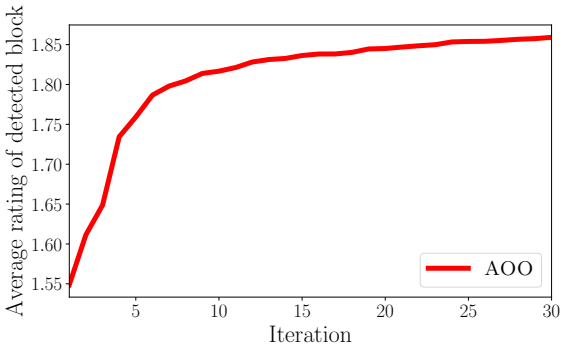


Fig. 10. Convergence on the average rating of the block detected by the proposed AOO method in Amazon rating data.

giving low ratings. We use the vocabulary and happiness score metrics in the work of Dodds et al., called "Hedonometer" [37], [38], to filter words in the product reviews. The size of the word in these "clouds" represents the frequency of the word in the reviews. If the size is bigger, the word is more frequent. We have the following observations. (1) As shown in Figure 11(a), most of the words in the reviews, such as "best", "wonderful", "interesting", and "excellent", are describing the user's positive feedback. This is intuitive because the average rating score of the whole dataset if as high as 4.2. (2) The words "bad" and "boring" are the most popular given by the users whose average rating is low. These two words can generally describe the polarity of their comments. (3) As shown in Figure 11(b), bully-like users are more likely to use sarcastic and disrespectful words such as "stupid", "hell", "terrible", and "horrible". The figure has many other words that destroy the interests of other users when reading these comments. (4) Figure 11(c) shows that the users who are giving low ratings but not predicted as bullies, often give words like "problem", "unfortunately", and "disappointed" to provide respectful and constructive comments. Their comments discussed the problems in the products (books or movies), the points they feel disappointed about, and the parts they feel unfortunate of getting positive feedback.

C. Experiments on Efficiency Performance

Figure 12 presents AOO's running time in terms of the number of buyers n. Here the number of sellers m is set as the same as the number of buyers n; the number of ratings n_r by each buyer is proportional to the number of sellers m. We fit the running time curve with a quadratic function and the residual error is 132 seconds. If we fit the running time curve with a linear regression, the residual error is twice bigger. From the figure, we observe that AOO has a quadratic complexity to the number of buyers n.

V. Discussions

This research was memorizing the great sorrow that **Ms. Yingying Zhang**, a 26-year-old visiting scholar at University of Illinois (UI), was abducted on June 9, 2017, when she was going to sign an apartment lease off campus. The criminal suspect who was a graduate in the university allegedly lured her into his car after she got off one bus and tried to flag down the transfer bus. The UI Police Department, the Illinois State Police, and the FBI have been dedicated to investigate her disappearance. People expect to have universities as places of safety, success, and happiness. The powerful-looking "platforms" can hardly take actions to protect us. Students, faculty, and staff start learning how to prevent abductions by ourselves, like never go anywhere with a stranger, even though it would be convenient or interesting. Though we can take actions to protect ourselves, we had little knowledge – the platforms have gathered countless historical data that would help us! *Can algorithms learn "thresholds" like what we have done in this work, from the data that police officers have, so that Yingying, a young woman student who was the first time being abroad, will have a "sensitive threshold" on taking a free ride offered by a stranger?*

Motivation: The motivation of this work is to study how the knowledge mined from big data can be transferred to easy actions in real life. The application we focused on in how to learn optimal blocklists that can be easily adopted by sellers from the rating data (we assume that either the plugins or platforms can access buyers' rating history, which is true). We sincerely hope the research problem and methodology can be transferred to other applications such as cyber security, phone security, and crime alert.

VI. Conclusions

In this paper, we revisited the problem of suspicious behavior detection at the perspective of individuals and bridge them with knowledge from the platform's massive data. We proposed a novel Actionable Objective Optimization (AOO) method that finds actionable knowledge in the form of optimal action for target users to block frauds themselves by collectively considering all target users' decisions. Experimental results demonstrated that our proposed method is effective and efficient. We compared with the award-winning, state-of-the-art algorithms. When most of them could not achieve so good performance that no false positive needs to be concerned, our AOO method can give almost perfect performances under different circumstances. Moreover, it is insensitive to the parameters and of a practical time complexity.

References

[1] "7 types of ebay buyers to add..." http://www.ebay.co.uk/gds/7-Types-of-EBay-Buyers-to-Add-to-Ur-Blocked-Buyer-List-/10000000020467012/g.html, 2011.

[2] B. Prakash, A. Sridharan, M. Seshadri, S. Machiraju, and C. Faloutsos, "Eigenspokes: Surprising patterns and scalable community chipping in large graphs," *Advances in knowledge discovery and data mining*, pp. 435–448, 2010.

[3] M. Jiang, P. Cui, A. Beutel, C. Faloutsos, and S. Yang, "Inferring lockstep behavior from connectivity pattern in large graphs," *Knowledge and Information Systems*, vol. 48, no. 2, pp. 399–428, 2016.

[4] A. Beutel, W. Xu, V. Guruswami, C. Palow, and C. Faloutsos, "Copycatch: stopping group attacks by spotting lockstep behavior in social networks," in *WWW*, 2013, pp. 119–130.

[5] N. Shah, A. Beutel, B. Gallagher, and C. Faloutsos, "Spotting suspicious link behavior with fbox: An adversarial perspective," in *ICDM*, 2014, pp. 959–964.

[6] M. Jiang, P. Cui, A. Beutel, C. Faloutsos, and S. Yang, "Catchsync: catching synchronized behavior in large directed graphs," in *KDD*, 2014, pp. 941–950.

[7] M. Jiang, A. Beutel, P. Cui, B. Hooi, S. Yang, and C. Faloutsos, "Spotting suspicious behaviors in multimodal data: A general metric and algorithms," *IEEE TKDE*, vol. 28, no. 8, pp. 2187–2200, 2016.

(a) All user's reviews (b) Bully-like user's reviews (c) Non-bully-like low-rating user's reviews

Fig. 11. Comparing word clouds from review content of different categories of buyers. Bully-like users are more likely to use sarcastic and disrespectful words. Non-bully buyers who are giving low ratings use words like "problem", "unfortunately", and "disappointed" to provide respectful and constructive comments.

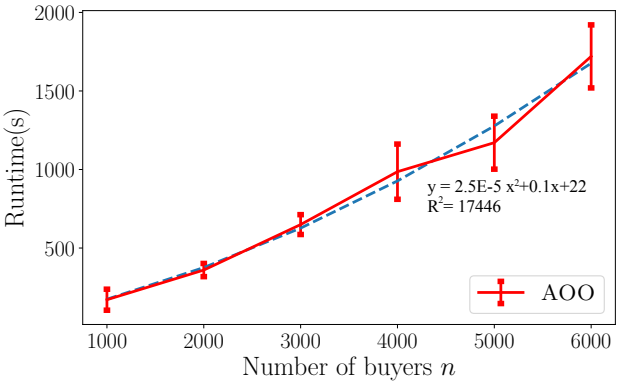


Fig. 12. The time complexity of the proposed AOO method.

[8] B. Hooi, H. A. Song, A. Beutel, N. Shah, K. Shin, and C. Faloutsos, "Fraudar: Bounding graph fraud in the face of camouflage," in *KDD*, 2016, pp. 895–904.

[9] K. Shin, B. Hooi, J. Kim, and C. Faloutsos, "Densealert: Incremental dense-subtensor detection in tensor streams," in *KDD*, 2017.

[10] S. Kumar, W. L. Hamilton, J. Leskovec, and D. Jurafsky, "Community interaction and conflict on the web," in *WWW*, 2018, pp. 933–943.

[11] M. Jiang, P. Cui, and C. Faloutsos, "Suspicious behavior detection: Current trends and future directions," *IEEE Intelligent Systems*, vol. 31, no. 1, pp. 31–39, 2016.

[12] D. Wang, M. Jiang, Q. Zeng, Z. Eberhart, and N. V. Chawla, "Multi-type itemset embedding for learning behavior success," in *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. ACM, 2018, pp. 2397–2406.

[13] X. Hu, J. Tang, and H. Liu, "Online social spammer detection." in *AAAI*, vol. 14, 2014, pp. 59–65.

[14] C. Chiu, J. Zhan, and F. Zhan, "Uncovering suspicious activity from partially paired and incomplete multimodal data," *IEEE Access*, vol. 5, pp. 13 689–13 698, 2017.

[15] S. Liu, B. Hooi, and C. Faloutsos, "Holoscope: Topology-and-spike aware fraud detection," in *CIKM*, 2017, pp. 1539–1548.

[16] S. Vosoughi, M. Mohsenvand, and D. Roy, "Rumor gauge: predicting the veracity of rumors on twitter," *ACM TKDD*, vol. 11, no. 4, p. 50, 2017.

[17] H. Shen, F. Ma, X. Zhang, L. Zong, X. Liu, and W. Liang, "Discovering social spammers from multiple views," *Neurocomputing*, vol. 225, pp. 49–57, 2017.

[18] S. Pandit, D. H. Chau, S. Wang, and C. Faloutsos, "Netprobe: a fast and scalable system for fraud detection in online auction networks," in *WWW*, 2007, pp. 201–210.

[19] S. Ghosh, B. Viswanath, F. Kooti, N. K. Sharma, G. Korlam, F. Benevenuto, N. Ganguly, and K. P. Gummadi, "Understanding and combating link farming in the twitter social network," in *WWW*, 2012, pp. 61–70.

[20] L. Akoglu, R. Chandy, and C. Faloutsos, "Opinion fraud detection in online reviews by network effects," in *WSDM*, 2013, pp. 2–11.

[21] C. Liu, X. Yan, H. Yu, J. Han, and P. S. Yu, "Mining behavior graphs for "backtrace" of noncrashing bugs," in *SDM*, 2005, pp. 286–297.

[22] Z. Zou, J. Li, H. Gao, and S. Zhang, "Mining frequent subgraph patterns from uncertain graph data," *IEEE TKDE*, vol. 22, no. 9, pp. 1203–1218, 2010.

[23] C. Giatsidis, D. M. Thilikos, and M. Vazirgiannis, "D-cores: Measuring collaboration of directed graphs based on degeneracy," in *ICDM*, 2011, pp. 201–210.

[24] J. Chen and Y. Saad, "Dense subgraph extraction with application to community detection," *IEEE TKDE*, vol. 24, no. 7, pp. 1216–1230, 2012.

[25] B. Perozzi, L. Akoglu, P. Iglesias Sánchez, and E. Müller, "Focused clustering and outlier detection in large attributed graphs," in *KDD*, 2014, pp. 1346–1355.

[26] C. Giatsidis, D. M. Thilikos, and M. Vazirgiannis, "Evaluating cooperation in communities with the k-core structure," in *ASONAM*, 2011, pp. 87–93.

[27] C. Tsourakakis, "The k-clique densest subgraph problem," in *WWW*, 2015, pp. 1122–1132.

[28] E. Galbrun, A. Gionis, and N. Tatti, "Top-k overlapping densest subgraphs." *DMKD*, vol. 30, no. 5, pp. 1134–1165, 2016.

[29] K. Shin, T. Eliassi-Rad, and C. Faloutsos, "Corescope: Graph mining using k-core analysispatterns, anomalies and algorithms," in *ICDM*, 2016, pp. 469–478.

[30] S. Khuller and B. Saha, "On finding dense subgraphs," in *International Colloquium on Automata, Languages, and Programming*. Springer, 2009, pp. 597–608.

[31] C. Tsourakakis, F. Bonchi, A. Gionis, F. Gullo, and M. Tsiarli, "Denser than the densest subgraph: extracting optimal quasi-cliques with quality guarantees," in *KDD*, 2013, pp. 104–112.

[32] A. Epasto, S. Lattanzi, and M. Sozio, "Efficient densest subgraph computation in evolving graphs," in *WWW*, 2015, pp. 300–310.

[33] K. Maruhashi, F. Guo, and C. Faloutsos, "Multiaspectforensics: Pattern mining on large-scale heterogeneous networks with tensor analysis," in *ASONAM*, 2011, pp. 203–210.

[34] K. Shin, B. Hooi, J. Kim, and C. Faloutsos, "D-cube: Dense-block detection in terabyte-scale tensors," in *WSDM*, 2017, pp. 681–689.

[35] S. Virdhagriswaran and G. Dakin, "Camouflaged fraud detection in domains with complex relationships," in *KDD*, 2006, pp. 941–947.

[36] B. Hooi, N. Shah, A. Beutel, S. Günnemann, L. Akoglu, M. Kumar, D. Makhija, and C. Faloutsos, "Birdnest: Bayesian inference for ratings-fraud detection," in *SDM*, 2016, pp. 495–503.

[37] H. D. Dixon, R. Frank, Y. Ng, and A. Oswald, "Economics and happiness," *The Economic Journal*, vol. 107, pp. 1812–1814, 1997.

[38] P. S. Dodds, K. D. Harris, I. M. Kloumann, C. A. Bliss, and C. M. Danforth, "Temporal patterns of happiness and information in a global social network: Hedonometrics and twitter," *PloS one*, vol. 6, no. 12, p. e26752, 2011.