REV2: Fraudulent User Prediction in Rating Platforms

Srijan Kumar
Stanford University, USA
srijan@cs.stanford.edu

Bryan Hooi
Carnegie Mellon University, USA
bhooi@cs.cmu.edu

Disha Makhija
Flipkart, India
disha.makhija@flipkart.com

Mohit Kumar
Flipkart, India
k.mohit@flipkart.com

Christos Faloutsos
Carnegie Mellon University, USA
christos@cs.cmu.edu

V.S. Subrahmanian
Dartmouth College, USA
vs@dartmouth.edu

ABSTRACT

Rating platforms enable large-scale collection of user opinion about items (e.g., products or other users). However, fraudulent users give fake ratings for excessive monetary gains. In this paper, we present Rev2, a system to identify such fraudulent users. We propose three interdependent intrinsic quality metrics—fairness of a user, reliability of a rating and goodness of a product. The fairness and reliability quantify the trustworthiness of a user and rating, respectively, and goodness quantifies the quality of a product. Intuitively, a user is fair if it provides reliable scores that are close to the goodness of products. We propose six axioms to establish the interdependency between the scores, and then, formulate a mutually recursive definition that satisfies these axioms. We extend the formulation to address cold start problem and incorporate behavior properties. We develop the Rev2 algorithm to calculate these intrinsic scores for all users, ratings, and products *by combining network and behavior properties*. We prove that this algorithm is guaranteed to converge and has linear time complexity. By conducting extensive experiments on five rating datasets, we show that Rev2 outperforms nine existing algorithms in detecting fraudulent users. We reported the 150 most unfair users in the Flipkart network to their review fraud investigators, and 127 users were identified as being fraudulent (84.6% accuracy). The Rev2 algorithm is being deployed at Flipkart.

ACM Reference Format:
Srijan Kumar, Bryan Hooi, Disha Makhija, Mohit Kumar, Christos Faloutsos, and V.S. Subrahmanian. 2018. REV2: Fraudulent User Prediction in Rating Platforms. In *Proceedings of 11th ACM International Conf. on Web Search and Data Mining (WSDM 2018).* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3159652.3159729

1 INTRODUCTION

User generated reviews are an essential part of several online platforms. For instance, users on major online marketplaces (e.g., Flipkart, Amazon, eBay, TripAdvisor, Yelp) rate the quality of products, restaurants, hotels, and other service offerings. Users trading anonymous cryptocurrency, such as Bitcoin, rate others on their trustworthiness. Because users frequently look at these ratings and reviews before buying a product online or choosing whom to trade

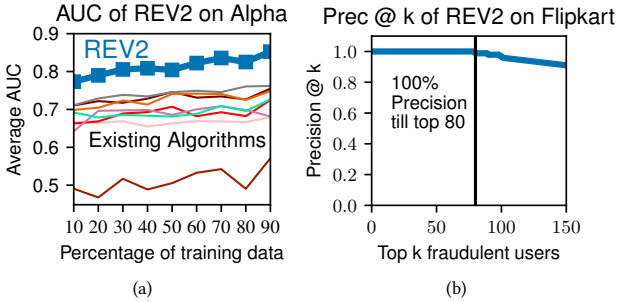


Figure 1: (a) Rev2 consistently performs the best, by having the highest AUC, in predicting fraudulent users with varying percentage of training labels. (b) Rev2 is effective in practice, with 127 out of top 150 flagged by Rev2 involved in fraud on Flipkart.

Bitcoins with, there is a huge monetary incentive for fraudulent users to give fake ratings [11, 13, 15, 34]. *The goal of this paper is to identify such fraudulent users.* This task is challenging due to the lack of training labels, disbalance in the percentage of fraudulent and non-fraudulent users, and camouflage by fraudulent users [8, 28].

In this paper, we present three metrics to quantify the trustworthiness of users and reviews, and the quality of products. We model user-to-item ratings with timestamps as a directed bipartite graph. For instance, on an online marketplace such as Amazon, a user u rates a product p with a rating (u, p). We propose that each user has an (unknown) intrinsic level of fairness $F(u)$, each product p has an (unknown) intrinsic goodness $G(p)$, and each rating (u, p) has an (unknown) intrinsic reliability $R(u, p)$. Intuitively, a fair user should give ratings that are close in score to the goodness of the product, and good products should get highly positive reliable ratings. Clearly, $F(u), G(p), R(u, p)$ are all inter-related, so we define six intuitive axioms that these intrinsic metrics should satisfy in relation to each other. We propose three mutually-recursive equations which satisfy the axioms to evaluate the values of these metrics.

However, the true trustworthiness of users that have given and true quality of products that have received only a few ratings is not certain [18, 21]. We address this *cold start problem* by adding Laplace smoothing in fairness and goodness formulations to incorporate default priors beliefs.

In addition to the rating network used so far, an entity's behavior properties are very indicative of its true quality. For instance, rapid or regular rating behavior is typical of fraudulent entities like fake accounts, sybils and bots [7, 16, 30, 31]. Similarly, fake ratings have short text [24], and bursty ratings received by a product may be indicative of fake reviews [35]. Therefore, we define an axiom that establishes the relation between the intrinsic scores and their

WSDM 2018, February 5–9, 2018, Marina Del Rey, CA, USA

ACM ISBN 978-1-4503-5581-0/18/02…$15.00
https://doi.org/10.1145/3159652.3159729

behavior properties, and propose a Bayesian technique to incorporate the behavior properites of users, ratings, and products into the formulation by penalizing for unusual behavior [7].

Combining the network, cold start treatment and behavioral properties together, we present the Rev2 formulation and an iterative algorithm to find the fairness, goodness and reliability scores of all entities together. Rev2 works in both unsupervised, i.e., when no training labels are present, and in supervised settings. We show that Rev2 formulation satisfies the six axioms, and the algorithm it is guaranteed to converge, in bounded number of iterations, and has linear time complexity.

How well does Rev2 work? We conduct extensive experiments using five real-world data sets—two Bitcoin user trust networks, Epinions, Amazon, and Flipkart dataset, India's biggest online marketplace. By conducting a series of five experiments, we show that Rev2 outperforms several existing methods [2, 7, 18, 22–24, 26, 32] in predicting fraudulent users. Specifically, in an unsupervised setting, Rev2 has the best average precision in eight out of ten cases and second best in the remaining two, showing its effectiveness even when no training labels are present. In two supervised settings, Rev2 has the highest AUC (≥ 0.85) across all five datasets. It consistently performs the best in detecting fraudulent users when percentage of training data is varied between 10 and 90, as shown for the Alpha network in Figure 1(a). Further, we experimentally show that both cold start treatment and behavior properties improve Rev2 algorithm's performance.

Rev2 is practically very useful as well. We reported the 150 most unfair users as predicted by Rev2 in the Flipkart online marketplace. Review fraud investigators at Flipkart studied our recommendations and confirmed that 127 of them were fraudulent *(84.6% accuracy)*, presenting a validation of the utility of Rev2 in identifying real-world review fraud. In fact, *Rev2 is already being deployed at Flipkart.* Figure 1(b) shows the precision@k of Rev2 on Flipkart dataset, showing 100% precision till top 80 unfair users.

Overall, the paper makes the following contributions:

• **Algorithm**: We propose three metrics called fairness, goodness and reliability to rank users, products and ratings, respectively. We propose Bayesian approaches to address cold start problems and incorporate behavioral properties. We propose the Rev2 algorithm to iteratively compute these metrics. It works in both unsupervised and supervised settings.

• **Theoretical guarantees**: Rev2 is guaranteed to converge in a bounded number of iterations, and has linear time complexity.

• **Effectiveness**: Rev2 outperforms nine existing algorithms in identifying fraudulent users, conducted via five experiments on five rating networks, with AUC ≥ 0.85.

Codes of Rev2 are available in the appendix [1].

2 RELATED WORK

Existing works in rating fraud detection can be categorized into network-based and behavior-based algorithms:

Network-based fraud detection algorithms are based on iterative learning, belief propagation, and node ranking techniques. Similar to the proposed Rev2 algorithm, [17, 22, 32, 33] develop iterative algorithms that jointly assign scores in the rating networks based on consensus of ratings. The closest formulation is that of Wang et al. [33**?**], which also create metrics for users, ratings, and products, but their formulation is different, does not satisfy the axioms (Section 3), and is outperformed by Rev2 (Section 5). FraudEagle [2] is a belief propagation model to rank users, which assumes fraudsters rate good products poorly and bad products positively, and vice-versa for honest users. Random-walk based algorithms have been developed to detect trolls [36] and link farming from collusion on Twitter [6]. [10, 17] identify group of fraudsters based on local neighborhood of the users. A survey on network-based fraud detection can be found in [3].

Table 1: Proposed Rev2 algorithm satisfies all desirable properties.

	BIRDNEST[7]	Trustiness[32, 33]	BAD[22]	FraudEagle[2]	SpEagle[26]	[18, 23, 24]	Rev2
Uses network information	✓	✓	✓	✓	✓		✓
Uses behavior properties	✓				✓	✓	✓
Parameter free	✓		✓			✓	✓
Theoretical Guarantees			✓				✓

Behavioral fraud detection algorithms are often feature-based. Consensus based features have been proposed in [18, 24] – our proposed goodness metric is also inspired by consensus or 'wisdom of crowds'. Commonly used features are derived from timestamps [21, 35, 37] and review text [5, 27, 29]. SpEagle [26] extends FraudEagle [2] to incorporate behavior features. BIRDNEST [7] creates a Bayesian model to estimate the belief of each user's deviation in rating behavior from global expected behavior. [4, 10, 31] study coordinated spam behavior of multiple users. A survey on behavior based algorithms can be found in [11].

The proposed Rev2 algorithm combines both network and behavior components together. Rev2 has theoretical guarantees of convergence and does not require any user inputs (i.e., it is prior free). Table 1 compares Rev2 to the closest existing algorithms, which shows that none of them satisfy all desirable properties.

3 REV2 FORMULATION

We consider directed bipartite rating networks of user-to-product, where each rating is from a user u to an product p. We propose that users and ratings have (unknown) intrinsic scores that quantify how trustworthy they are, and products have (unknown) intrinsic scores that quantify its quality or how a layman is likely to evaluate it. Naturally, these scores are inter-dependent and unknown apriori. In this section, we describe the axioms that establish the inter-dependency between these scores, and propose an algorithm that satisfies the axioms and calculate these scores. The algorithm incorporates network properties, behavior features, and evaluates users with few reviews.

Preliminaries. A bipartite rating graph $G = (U, R, P)$ is a directed, weighted graph, where user $u \in U$ gives a rating $(u, p) \in R$ to product $p \in P$. Let the rating score be represented as `score(u,p)`. Let $\mathcal{U}, \mathcal{R}$ and $\mathcal{P}$ represent the set of all users, ratings and products, respectively, in a given bipartite network. We assume that all rating scores are scaled to be between -1 and +1, i.e. `score(u,p)` $\in [-1, 1] \forall (u, p) \in \mathcal{R}$. Let, `Out(u)` be the set of ratings given by user u and `In(p)` be the set of ratings received by product p. So, `|Out(u)|`

and |In(p)| represents their respective counts. The egonetwork of a user u = Out(u) $\bigcup\{p|(u,p) \in Out(u)\}$ and that of product p = In(p) $\bigcup\{u|(u,p) \in In(p)\}$.

Definition 1 [Identical ratings egonetworks]: Two users u_1 and u_2 are said to have identical ratings egonetworks if |Out(u_1)| = |Out(u_2)| and there exists a one-to-one mapping $h : Out(u_1) \rightarrow Out(u_2)$ such that score(u_1, p) = score(u_2, h(p)) $\forall(u_1,p) \in$ Out(u_1). Similarly, identical ratings egonetworks of two products p_1 and p_2 can be defined in a similar manner using ratings the products receive.

3.1 Intrinsic Properties: Fairness, Goodness and Reliability

We model the bipartite network such that users, ratings and products have (unknown) intrinsic quality scores. User and rating intrinsic scores indicates how trustworthy they are, and product intrinsic scores indicates how likable it is. Their meanings and purpose are explained below:

• Products vary in their quality, measured by a metric called ***goodness***. The goodness score is a single number indicating the most likely rating a fair user would give it. Intuitively, a good product would get several high positive ratings from fair users, and a bad product would receive high negative ratings from fair users. The goodness score $G(p)$ of a product p ranges from -1 (a very low quality product) to $+1$ (a very high quality product) $\forall p \in \mathcal{P}$. The need for incorporating the rating user's fairness arises because simple measures like a product's average or median scores can easily be manipulated by giving ratings using multiple fake accounts [12, 31].

• Users vary in terms of their ***fairness*** that indicates how trustworthy it is. Fair users rate products without bias, i.e., they give high scores to high quality products, and low scores to bad products. On the other hand, users who frequently deviate from the above behavior are 'unfair', e.g., fraudulent users that give high ratings to low quality products and low ratings to good products. Likewise, a 'strict' ('liberal', resp.) user who consistently gives negative (positive, resp.) ratings to high (low, resp.) goodness products is intuitively less fair than a user who gives negative (positive, resp.) ratings to low (high, resp.) goodness products. Fairness score $F(u)$ of a user u lies in the $[0, 1]$ interval $\forall u \in \mathcal{U}$. 0 denotes a 100% untrustworthy user, while 1 denotes a 100% trustworthy user.

• Finally, ratings vary in terms of ***reliability***, which reflects how trustworthy it is. The reliability score $R(u,p)$ of a rating (u,p) ranges from 0 (untrustworthy) to 1 (trustworthy) $\forall$(u,p) $\in \mathcal{R}$.

The reader may wonder: *isn't a rating's reliability identical to its user's fairness?* The answer is 'no'. We need separate intrinsic scores for users and ratings because a user may give ratings with different reliabilities due to their biases and perceptions. In Figure 2 we show the rating reliability distribution, measured by our Rev2 algorithm (explained in Section 4), of ground truth benign and fraudulent users in the Flipkart network. Notice that while most ratings by benign users have high reliability, some of their ratings have low reliability, indicating personal opinions which disagrees with majority (see green arrow). Conversely, even fraudulent users give some high reliability ratings (red arrow), possibly to camouflage themselves as benign users. Thus, having reliability as a rating-specific metric allows us to more accurately characterize this distribution.

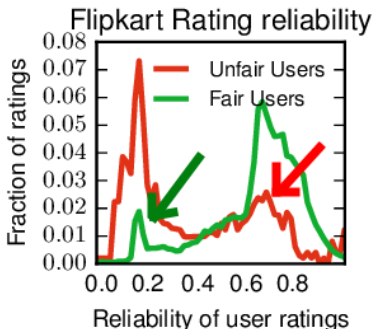


Figure 2: While most ratings by benign users have high reliability, some even have low reliability (green arrow). Conversely, fraudulent users give some highly reliability ratings (red arrow), but most of their ratings have low reliability.

Definition 2 [Identically reliable egonetworks]: We say that, two users u_1 and u_2 are said to have identically reliabile egonetworks if |Out(u_1)| = |Out(u_2)| and there exists a one-to-one mapping $h : Out(u_1) \rightarrow Out(u_2)$ such that reliability(u_1, p) = reliability(u_2, h(p)) $\forall(u_1,p) \in$ Out(u_1). Simiarly, identical ratings egonetworks of two products p_1 and p_2 can be defined in a similar manner by the ratings the products receive.

Given any user-rating-product graph, all intrinsic scores are unknown apriori. Clearly, these scores are mutually interdependent. Here, we describe five intuitive axioms that establish this interdependency. The first two axioms define the relation between a product and the ratings that it receives.

Axiom 1 (Better products get higher ratings). *If two products have identically reliable egonetworks and for one product, all the rating scores are higher, then quality of that product is more. Formally, for two products p_1 and p_2 have a one-to-one mapping $h : In(p_1) \rightarrow In(p_2)$ such that $R(u,p_1) = R(h(u),p_2)$ and* score(u, p_1) $\geq$ score(h(u), p_2) $\forall(u,p_1) \in In(p_1)$, *then* $G(p_1) \geq G(p_2)$.

Axiom 2 (Better products get more reliable positive ratings). *If two products have identical rating egonetworks and for the first product, all positive ratings are more reliable and all negative ratings are less reliable than for the second product, then the first product has higher quality. Formally, if two products p_1 and p_2 have a one-to-one mapping $h : In(p_1) \rightarrow In(p_2)$ such that $R(u,p_1) \geq R(h(u),p_2) \forall(u,p_1) \in In^+(p_1)$ and $R(u,p_1) \leq R(h(u),p_2) \forall(u,p_1) \in In^-(p_1)$, then $G(p_1) \geq G(p_2)$.*

The next two axioms define the relation between a rating and the user and product it belongs to, respectively. The next axiom uses 'wisdom of crowds', by reducing the reliability of rating that deviates from a product's goodness.

Axiom 3 (Reliable ratings are closer to goodness scores). *For two ratings by equally fair users, the rating with score closer to the product's gooness has higher reliability. Formally, if two ratings (u_1,p_1) and (u_2,p_2) are such that* score(u_1,p_1) = score(u_2,p_2), *$F(u_1) = F(u_2)$, and $|score(u_1,p_1) - G(p_1)| \geq |score(u_2,p_2) - G(p_2)|$, then $R(u_1,p_1) \leq R(u_2,p_2)$.*

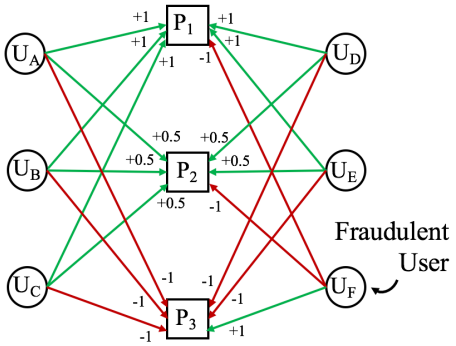


Figure 3: Toy example showing products (P_1, P_2, P_3), users (U_A, U_B, U_C, U_D, U_E and U_F), and rating scores provided by the users to the products. User U_F always disagrees with the ratings of others, so U_F is fraudulent.

One implication of this axiom is that different ratings given by the same user can have different reliability scores.

Axiom 4 (Reliable ratings are given by fairer users). *For two equal ratings to equal goodness products, the one given by more fair user has higher reliability. Formally, if two ratings (u_1, p_1) and (u_2, p_2) are such that* $\texttt{score}(u_1, p_1) = \texttt{score}(u_2, p_2)$, $F(u_1) \geq F(u_2)$, *and $G(p_1) = G(p_2)$, then $R(u_1, p_1) \geq R(u_2, p_2)$.*

This axiom incorporates the user's reputation in measuring reliability. This way same ratings received by a product may have different reliability scores.

The next axiom defines the relation between a user and its ratings.

Axiom 5 (Fairer users give more reliable ratings). *For two users with equal number of ratings, if one has higher reliability for all its ratings than the other, then it has higher fairness. Formally, if two users u_1 and u_2 have a one-to-one mapping $h : Out(u_1) \rightarrow Out(p_2)$ such that $|Out(u_1)| = |Out(u_2)|$ and $R(u_1, p) \geq R(u_2, h(p))$ $\forall (u_1, p) \in Out(u_1)$, then $F(u_1) \geq F(u_2)$*

Example 3.1 (Running Example). Figure 3 shows a simple example in which there are 3 products, P_1 to P_3, and 6 users, U_A to U_F. Each review is denoted as an edge from a user to a product, with rating score between -1 and $+1$. Note that this is a rescaled version of the traditional 5-star rating scale where a 1, 2, 3, 4 and 5 star corresponds to $-1, -0.5, 0, +0.5$ and $+1$ respectively.

One can immediately see that U_F's ratings are frequently inconsistent with those of U_A, U_B, U_C, U_D and U_E. U_F gives poor ratings to P_1 and P_2 which others all agree are very good by consensus. U_F also gives a high rating for P_3 which others agree is very bad. We will use this example to motivate our formal definitions below.

3.2 Our proposed formulation

Now that we have formally established the interdependencies between the three different metrics, below we propose one formulation that satisfies the axioms. Alternate formulations that satisfy the axioms can be developed as well, which we will explore in future work. Our proposed formulation is below. We will add cold start treatment and behavior properties to it later.

$$F(u) = \frac{\sum_{(\texttt{u},\texttt{p}) \in \texttt{Out(u)}} R(u,p)}{|\texttt{Out(u)}|} \tag{1}$$

$$G(p) = \frac{\sum_{(\texttt{u},\texttt{p}) \in \texttt{In(p)}} R(u,p) \cdot \texttt{score(u,p)}}{|\texttt{In(p)}|} \tag{2}$$

$$R(u,p) = \frac{(\gamma_1 \cdot F(u) + \gamma_2 \cdot (1 - \frac{|\texttt{score(u,p)} - G(p)|}{2}))}{\gamma_1 + \gamma_2} \tag{3}$$

Thus, the formulation results in ignoring low reliability ratings and giving more importance to high reliability ones. We show that the above formulation satisfies the axioms in the appendix [1].

In our *running example*, for the rating by user U_F to P_1:

$$R(U_F, P_1) = \frac{1}{2}\left(F(U_F) + (1 - \frac{|-1 - G(P_1)|}{2})\right)$$

Similar equations can be associated with every edge (rating) in the graph of Figure 3.

Addressing Cold Start Problems

In rating networks, most users give and most products receive only a few ratings [23]. For such users and products, there is little information to measure their true quality. For example, a user with few but highly accurate ratings may be a honest user or a fraudster who is camouflaging itself [8]. Conversely, a user with few and highly inaccurate ratings may be a benign novice or a throwaway fraud account [18, 24]. Due to lack of sufficient information, little can be said about their true fairness. The same happens with products that receive a few ratings. This uncertainty due to insufficient information of less active users and products is the *cold start problem*.

We address this cold start issue by adding Laplace smoothing to fairness and goodness scores as follows:

$$F(u) = \frac{\sum_{(u,p) \in \texttt{Out(u)}} R(u,p) + \alpha_1 \cdot \mu_f}{|\texttt{Out(u)}| + \alpha_1}$$

$$G(p) = \frac{\sum_{(u,p) \in \texttt{In(p)}} R(u,p) \cdot \texttt{score(u,p)} + \beta_1 \cdot \mu_g}{|\texttt{In(p)}| + \beta_1}$$

Smoothing parameters α_1 and β_1 are non-negative integer psuedocounts, and μ_f and μ_g are prior beliefs for fairness and goodness scores of new nodes, respectively. The prior is the default score each user and product has if it didn't give or get any ratings. The smoothing parameter sets the relative importance of prior – the lower (higher, resp.) the value of priors, the more (less, resp.) the fairness and goodness scores depend on the network component.

This works because if a user gives only a few ratings, then its fairness score is close to default score μ_f. The more ratings it gives, the more its fairness score moves towards the average of rating reliabilities. This way less active accounts have little impact on product goodness scores, and more importance is given to more active accounts.

The values of α_1, β_1 are set using parameter sweep (Section 4), and μ_f and μ_g are set as the mean scores of all users' fairness and all products' goodness scores, respectively.

Incorporating Behavioral Properties

In the formulation so far, we have solely used the rating graph to calculate the fairness, goodness and reliability values. But, the behavior and characteristics of users, ratings and products are very informative of their qualities as well, which may not be captured by the rating graph alone. For example, as shown in several works [7,

$$F(u) = \frac{\sum_{(u,p)\in \texttt{Out(u)}} R(u,p) + \alpha_1 \cdot \mu_f + \alpha_2 \cdot \Pi_U(u)}{|\texttt{Out(u)}| + \alpha_1 + \alpha_2}$$

$$R(u,p) = \frac{\gamma_1 \cdot F(u) + \gamma_2 \cdot (1 - \frac{|\texttt{score(u,p)} - G(p)|}{2}) + \gamma_3 \cdot \Pi_R(u,p)}{\gamma_1 + \gamma_2 + \gamma_3}$$

$$G(p) = \frac{\sum_{(u,p)\in \texttt{In(p)}} R(u,p) \cdot \texttt{score(u,p)} + \beta_1 \cdot \mu_g + \beta_2 \cdot \Pi_P(p)}{|\texttt{In(p)}| + \beta_1 + \beta_2}$$

Figure 4: The Rev2 formulation: this is the set of mutually recursive definitions of fairness, reliability and goodness of the proposed Rev2 algorithm. The yellow shaded part addresses the cold start problems and gray shaded part incorporates the behavioral properties.

16, 31], fraudulent users are bursty, i.e., they give several ratings in a very short timespan, or very regular, i.e., they give ratings after fixed intervals. Even though the rating scores themselves may be very accurate, the unusually rapid or regular behavior of the user is suspicious. Therefore, behavioral properties of the ratings that a user gives or a product receives are indicative of their true nature.

Let, $\Pi_U(u)$ represent the behavior 'normality' score of a user u, with respect to some set of behavior properties. Lower (higher, resp.) score indicates more anomalous (normal, resp.) behavior. The following axiom ties the relation between behavior properties and other components:

Axiom 6. *For two users with identically reliable egonetworks, the one with higher behavior score has higher fairness. Formally, if two users u_1 and u_2 have a one-to-one mapping $h : Out(u_1) \rightarrow Out(p_2)$ such that $|Out(u_1)| = |Out(u_2)|$, $R(u_1, p) = R(u_2, h(p))$ $\forall (u_1, p) \in Out(u_1)$, and $\Pi_U(u_1) \geq \Pi_U(u_2)$, then $F(u_1) \geq F(u_2)$.*

Similar axioms can be defined for ratings and products as well.

Instead of developing our own algorithm to model behavior, we incorporate the state-of-the-art one in our framework. BIRDNEST [7] calculates a Bayesian estimate of how much a user's properties deviates from the global population of all users' properties, and assigns a suspicion score $0 \leq BN_U(u) \leq 1$ to each user u. Then, normality score is simply $\Pi_U(u) = 1 - BN_U(u)$. Normality score for a product $\Pi_P(p)$ and a rating $\Pi_R(u,p)$ are calculated from the properties of all products and all ratings, respectively. In our experiments, we use user's and product's inter-rating times, and a rating's text for calculating this score.

We propose a Bayesian approach to incorporate behavior properties into the formulation so far. We add Laplace smoothing to the network components, where the behavior normality scores are taken as the priors. As opposed to the global priors in cold start treatment, these priors can be different for each user, review and product based on their normality scores. The resulting equations are given in Figure 4. The smoothing parameters α_2, γ_3 and β_2 are non-negative integers.

Overall, the Rev2 formulation, represented in Figure 4, is the set of three equations that define the fairness, goodness and reliability scores in terms of each other, by combining rating network and behavior properties together.

Algorithm 1 Rev2 Algorithm

1: **Input**: Rating network $(\mathcal{U}, \mathcal{R}, \mathcal{P})$, α_1, α_2, β_1, β_2, γ_1, γ_2, γ_3
2: **Output**: Fairness, Reliability and Goodness scores, given α_1, α_2, β_1, β_2, , γ_1, γ_2, γ_3
3: Calculate $\Pi_U(u) \forall u \in \mathcal{U}$, $\Pi_R(u,p) \forall (u,p) \in \mathcal{R}$, and $\Pi_P(p) \forall p \in \mathcal{P}$.
4: Initialize $F^0(u) = \Pi_U(u)$, $R^0(u,p) = \Pi_R(u,p)$, and $G^0(p) = \Pi_P(p)$, $\forall u \in \mathcal{U}$, $(u,p) \in \mathcal{R}$, $p \in \mathcal{P}$.
5: $t = 0$
6: Initialize $\mu_f = \frac{\sum_{u\in\mathcal{U}} F^t(u)}{|\mathcal{U}|}$, $\mu_g = \frac{\sum_{p\in\mathcal{P}} G^t(p)}{|\mathcal{P}|}$
7: **do**
8: $t = t + 1$
9: Update goodness of products using Equation 4: $\forall p \in \mathcal{P}$,
$G^t(p) = \frac{\sum_{(u,p)\in \texttt{In(p)}} R^{t-1}(u,p).\texttt{score(u,p)} + \beta_1 \cdot \mu_g + \beta_2 \cdot \Pi_P(p)}{|\texttt{In(p)}| + \beta_1 + \beta_2}$.
10: Update reliability of ratings using Equation 4: $\forall (u,p) \in \mathcal{R}$,
$R^t(u,p) = \frac{\gamma_1 \cdot F^{t-1}(u) + \gamma_2 \cdot (1 - \frac{|\texttt{score(u,p)} - G^t(p)|}{2}) + \gamma_3 \cdot \Pi_R(u,p)}{\gamma_1 + \gamma_2 + \gamma_3}$.
11: Update fairness of users using Equation 4: $\forall u \in \mathcal{U}$,
$F^t(u) = \frac{\sum_{(u,p)\in \texttt{Out(u)}} R^t(u,p) + \alpha_1 \cdot \mu_f + \alpha_2 \cdot \Pi_U(u)}{|\texttt{Out(u)}| + \alpha_1 + \alpha_2}$
12: error = $max(\sum_{u\in\mathcal{U}} |F^t(u) - F^{t-1}(u)|,$
13: $\sum_{(u,p)\in\mathcal{R}} |R^t(u,p) - R^{t-1}(u,p)|, \sum_{p\in\mathcal{P}} |G^t(p) - G^{t-1}(p)|)$
14: **while** $error > \epsilon$
15: **Return** $F^t(u)$, $R^t(u,p)$, $G^t(p)$, $\forall u \in \mathcal{U}$, $(u,p) \in \mathcal{R}$, $p \in \mathcal{P}$

4 THE REV2 ALGORITHM

Having formulated the fairness, goodness and reliability metrics in Section 3, we present the Rev2 algorithm in Algorithm 1 to calculate the metrics' values for all users, products and ratings. The algorithm is iterative, so let F^t, G^t and R^t denote the fairness, goodness and reliability score at the end of iteration t. Given the rating network and non-negative integers $\alpha_1, \alpha_2, \beta_1, \beta_2, \gamma_1, \gamma_2$ and γ_3, we first initialize all scores using their respective behavior scores Π_U, Π_R, and Π_P. When behavior scores are not present (e.g., when behavior properties are unknown), then these scores are initialized to highest value 1. Then we iteratively update the scores using equations in Figure 4 until convergence (see lines 7–14). Convergence occurs when all scores change minimally (see line 14). ϵ is the acceptable error bound, which is set to a very small value, say 10^{-6}.

But how do we set the values of $\alpha_1, \alpha_2, \beta_1, \beta_2, \gamma_1, \gamma_2$ and γ_3? In an ***unsupervised*** scenario, it is not possible to find the best combination of these values apriori. Therefore, the algorithm is run for several combinations of $\alpha_1, \alpha_2, \beta_1, \beta_2, \gamma_1, \gamma_2$ and γ_3 as inputs, and the final scores of a user across all these runs are averaged to get the final Rev2 score of the user. Formally, let C be the set of all parameter combinations $\{\alpha_1, \alpha_2, \beta_1, \beta_2, \gamma_1, \gamma_2, \gamma_3\}$, and $F(u|\alpha_1, \alpha_2, \beta_1, \beta_2, \gamma_1, \gamma_2, \gamma_3)$ be the fairness score of user u after running Algorithm 1 with $\alpha_1, \alpha_2, \beta_1, \beta_2, \gamma_1, \gamma_2, \gamma_3$ as inputs. So the final fairness score of user u is $F(u) = \frac{\sum_{(\alpha_1,\alpha_2,\beta_1,\beta_2,\gamma_1,\gamma_2,\gamma_3)\in C} F(u|\alpha_1,\alpha_2,\beta_1,\beta_2,\gamma_1,\gamma_2,\gamma_3)}{|C|}$. Similarly, $G(p)$ and $R(u,p)$ are calculated. In our experiments, we varied all these parameters from 0 through 2, i.e., $0 \leq \alpha_1, \alpha_2, \beta_1, \beta_2, \gamma_1, \gamma_2, \gamma_3 \leq 2$, giving $3^7 = 2187$ combinations. So, altogether, scores from 2187 different runs were averaged to get the final *unsupervised* Rev2 scores. This final score is used for ranking the users.

In a ***supervised*** scenario, it is indeed possible to learn the relative importance of parameters. We represent each user u as a feature vector of its fairness scores across several runs, i.e. $F(u|\alpha_1, \alpha_2, \beta_1, \beta_2, \gamma_1,$

$\gamma_2, \gamma_3)$, $\forall(\alpha_1, \alpha_2, \beta_1, \beta_2, \gamma_1, \gamma_2, \gamma_3) \in C$ are the features for user u. Given a set of fraudulent and benign user labels, a random forest classifier is trained that learns the appropriate weights to be given to each score. The higher the weight, the more important the particular combination of parameter values is. The learned classifier's prediction labels are then used as the *supervised* Rev2 output.

Example 4.1 (Running Example). Let us revisit our running example. Let, $\alpha_1 = \alpha_2 = \beta_1 = \beta_2 = \gamma_1 = \gamma_2 = \gamma_3 = 0$. We initialize all fairness, goodness and reliability scores to 1 (line 4 in Algorithm 1). Consider the first iteration of the loop (i.e., when t is set to 1 in line 8). In line 9, we must update the goodness of all products. Let us start with product P_1. Its goodness $G^0(P_1)$ was 1, but this gets updated to

$$G^1(P_1) \quad = \quad \frac{-1(1) + 1(1) + 1(1) + 1(1) + 1(1) + 1(1)}{6} = 0.67.$$

We see that the goodness of P_1 has dropped because of U_A's poor rating. The following table shows how the fairness and goodness values change over iterations (we omit reliability for brevity):

		Fairness/Goodness in iterations				
Node	Property	0	1	2	5	9 (final)
P_1	$G(P_1)$	1	0.67	0.67	0.67	0.68
P_2	$G(P_2)$	1	0.25	0.28	0.31	0.32
P_3	$G(P_3)$	1	-0.67	-0.67	-0.67	-0.68
U_A–U_E	$F(U_A)$–$F(U_E)$	1	0.92	0.89	0.86	0.86
U_F	$F(U_F)$	1	0.62	0.43	0.24	0.22

By symmetry, nodes U_A to U_E have the same fairness values throughout. After convergence, U_F has low fairness score, while U_A to U_E have close to perfect scores. Confirming our intuition, the algorithm quickly learns that U_F is a fraudulent user as all of its ratings disagree with the rest of the users. Hence, the algorithm then downweighs U_F's ratings in its estimation of the products' goodness, raising the score of P_1 and P_2 as they deserve.

4.1 Theoretical Guarantees of Rev2

Here we present the theoretical properties of Rev2. Let, $F^\infty(u), G^\infty(p)$ and $R^\infty(u,p)$ be the final scores after convergence, for some input $\alpha_1, \alpha_2, \beta_1, \beta_2, \gamma_1, \gamma_2, \gamma_3$. Proofs are shown in appendix [1] due to lack of space.

Lemma 4.2 (**Lemma 1**). *The difference between a product p's final goodness score and its score after the first iteration is at most 1, i.e.* $|G^\infty(p) - G^1(p)| \leq 1$. *Similarly,* $|R^\infty(u,p) - R^1(u,p)| \leq 3/4$ *and* $|F^\infty(u) - F^1(u)| \leq 3/4$.

Theorem 4.3 (**Convergence Theorem**). *The difference during iterations is bounded as* $|R^\infty(u,p) - R^t(u,p)| \leq (\frac{3}{4})^t, \forall(u,p) \in \mathcal{R}$. *As t increases, the difference decreases and* $R^t(u,p)$ *converges to* $R^\infty(u,p)$. *Similarly,* $|F^\infty(u) - F^t(u)| \leq (\frac{3}{4})^t, \forall u \in \mathcal{U}$ *and* $|G^\infty(p) - G^t(p)| \leq (\frac{3}{4})^{(t-1)}, \forall p \in \mathcal{P}$.

As the algorithm converges for all combinations of $\alpha_1, \alpha_2, \beta_1, \beta_2, \gamma_1, \gamma_2, \gamma_3$, the entire algorithm is guaranteed to converge.

Corollary 4.4 (**Iterations till Convergence**). *The number of iterations needed to reach convergence is at most* $2 + \lceil \frac{\log(\epsilon/2)}{\log(3/4)} \rceil$. *In other words, treating* ϵ *(acceptable error) as constant, the number of iterations needed to reach convergence is bounded by a constant.*

Thus, the lower the ϵ value, higher the number of iterations till convergence.

Table 2: Five rating datasets used for evaluation.

Network	# Users (% unfair, fair)	# Products	# Edges
OTC	4,814 (3.7%, 2.8%)	5,858	35,592
Alpha	3,286 (3.1%, 4.2%)	3,754	24,186
Amazon	256,059 (0.09%, 0.92%)	74,258	560,804
Flipkart	1,100,000 (-, -)	550,000	3,300,000
Epinions	120,486 (0.84%, 7.7%)	132,585	4,835,208

Linear algorithmic time complexity. In each iteration, the algorithm updates goodness, reliability and fairness scores of each product, rating and user, respectively. Each of these updates takes constant time. So, the complexity of each iteration is $O(|E| + |V|) = O(|E|)$. By Corollary 4.4, the algorithm converges in a constant number of iterations. Hence the time complexity is $O(k|E|)$, which is linear in the number of edges, and k is a constant equal to the product of the number of iterations till convergence and the number of runs of the algorithm.

5 EXPERIMENTAL EVALUATION

In this section, we present the results of the Rev2 algorithm to identify fraudulent users.. We conduct extensive experiments on five different rating networks and show five major results:
(i) We compare Rev2 algorithm with five state of the art algorithms to predict fraudulent and benign users in an *unsupervised setting*. We show that Rev2 performs the best in eight out of ten cases and second best in the remaining two.
(ii) In a *supervised setting* when training data is present, we show that Rev2 outperforms nine algorithms across all datasets.
(iii) We show that Rev2 is robust to the percentage of training data available, and consistently performs the best.
(iv) We show that both cold start treatment and behavior properties improve the performance of Rev2 algorithm.
(v) We show the linear running time of Rev2.

The Rev2 algorithm is already being deployed at Flipkart. All codes and datasets (except Flipkart) are in the appendix [1].

5.1 Datasets: Rating Networks

Table 2 shows the properties of the five datasets used. All ratings are rescaled between -1 and +1.

• **Flipkart** is India's biggest online marketplace where users rate products. The ground truth labels are manually verified by review fraud investigators in Flipkart.

• **Bitcoin OTC** is a user-to-user trust network of Bitcoin users trading using OTC platform [14]. The network is made bipartite by splitting each user into a 'rater' with all its outgoing edges and 'product' with all incoming edges. The ground truth is defined as: benign users are the platform's founder and users he rated highly positively (≥ 0.5). Fraudulent users are the ones that these trusted users uniformly rate negatively (≤ -0.5).

• **Bitcoin Alpha** is the Bitcoin trust network of Alpha platform users [14]. Its ground truth is created similar to OTC, starting from the founder of this platform.

• **Epinions** network has two independent components—user-to-post rating network and user-to-user trust network [19]. Algorithms are run on the rating network and ground truth is defined using the separate trust network: a user is defined as benign if its total trust rating is $\geq +10$, and fraudulent if ≤ -10.

• **Amazon** is a user-to-product rating network [20]. Ground truth

Table 3: Unsupervised Predictions: The table shows the Average Precision values of all algorithms in unsupervised prediction of fraudulent and benign users across five datasets. The best algorithm in each column is colored blue and second best is light blue. Overall, REV2 performs the best or second best in 9 of the 10 cases. *nc* indicates 'no convergence'.

	Fraudulent user prediction					Benign user prediction				
	OTC	Alpha	Amazon	Epinions	Flipkart	OTC	Alpha	Amazon	Epinions	Flipkart
FraudEagle	93.67	86.08	47.21	*nc*	*nc*	86.94	71.99	96.88	*nc*	*nc*
BAD	79.75	63.29	55.92	58.31	79.96	77.41	68.31	97.19	97.09	38.07
SpEagle	74.40	68.42	12.16	*nc*	*nc*	80.91	82.23	93.42	*nc*	*nc*
BIRDNEST	61.89	53.46	19.09	37.08	85.71	46.11	77.18	93.32	98.53	62.47
Trustiness	74.11	49.40	40.05	*nc*	*nc*	84.09	78.19	97.33	*nc*	*nc*
REV2	96.30	75.29	64.89	81.56	99.65	92.85	84.85	100.0	99.81	42.83

is defined using helpfulness votes, which is indicative of malicious behavior [5]—users with at least 50 votes are benign if the fraction of helpful-to-total votes is ≥ 0.75, and fraudulent if ≤ 0.25. Experiments on Yelp dataset [26] is not done as it has all reviews for some products but not all reviews by users, leading to a sparse network.

5.2 Baselines

We compare REV2 with nine state-of-the-art unsupervised and supervised algorithms. The **unsupervised** algorithms are:

- *Bias and Deserve (BAD)* [22] assigns a bias score $bias(u)$ to each user u, which measures user's tendency to give high or low ratings. $1 - |bias(u)|$ is the prediction made by BAD.
- *Trustiness* [32, 33] algorithm assigns a trustiness, honesty and reliability score to each user, product and rating, respectively. We use the trustiness score as its prediction.
- *FraudEagle* [2] is a belief propagation based algorithm. Users are ranked according to their fraud score.
- *SpEagle* [26] incorporates behavior features (user, rating text, and product) into FraudEagle, and the final spam scores of users are used for ranking.
- *BIRDNEST* [7] ranks users by creating a Bayesian model with users' timestamp and rating distributions.

We also compare with **supervised** algorithms that need training data:

- *SpEagle+* [26] is a supervised extension of SpEagle that leverages known training labels in the ranking.
- *SpamBehavior* [18]: This technique uses user's average rating deviations as feature.
- *Spamicity* [23] is creates each user's features as its review burstiness and maximum reviews per day.
- *ICWSM'13* [24] uses user's fraction of positive reviews, maximum reviews in a day. and average rating deviation as features.

For fair comparison, all algorithms that use behavior properties, including REV2, are given users' inter-rating time distribuion, ratings' text features (e.g., number of words, LIWC category distribution [25], sentiment [9]), and products' inter-rating time distribution as the properties.

5.3 Experiment 1: Unsupervised Prediction

In this experiment, the task is to rank the users based on how fraudulent they are. We compare *unsupervised* REV2 with the suite of five unsupervised algorithms in terms of their *Average Precision scores*, which measures the relative ordering the algorithm gives to the ground truth fraudulent and benign users. These are calculated for the top and bottom 100 users, and correspond to the area under the precision-recall curve. Table 3 shows the resulting average precision scores on the five datasets.

Table 4: Supervised Predictions: AUC values of 10-fold cross validation to predict fradulent users with individual predictions as features in a Random Forest classifier. REV2 performs the best across all datasets. *nc* means 'no convergence'.

	OTC	Alpha	Amazon	Epinions	Flipkart
FraudEagle	0.89	0.76	0.81	*nc*	*nc*
BAD	0.79	0.68	0.80	0.81	0.64
SpEagle	0.69	0.57	0.63	*nc*	*nc*
BIRDNEST	0.71	0.73	0.56	0.84	0.80
Trustiness	0.82	0.75	0.72	*nc*	*nc*
SpEagle+	0.55	0.66	0.67	*nc*	*nc*
SpamBehavior	0.77	0.69	0.80	0.80	0.60
Spamicity	0.88	0.74	0.60	0.50	0.82
ICWSM'13	0.75	0.71	0.84	0.82	0.82
REV2	0.90	0.88	0.85	0.90	0.87

We see that REV2 algorithm performs the best in 4 out of 5 cases and second best in the remaining one, for identifying both fraudulent and benign users. There is no consistently well performing existing algorithm. Note that FraudEagle, SpEagle and Trustiness are not scalable and do not converge for the two largest networks, Epinions and Flipkart, as opposed to REV2 which is guaranteed to converge. We discuss scalability in Section 5.7.

We also show the precision@k curve for identifying unfairs by REV2 on Flipkart network in Figure 1(b). REV2 achieves 100% precision till top 80. The other networks have similar precision@k curve, shown in appendix [1]. Among the top 150 unfair users identified by REV2, Flipkart fraud investigators confirmed total of 127 as fraudulent (84.6% accuracy).

5.4 Experiment 2: Supervised Prediction

In this experiment, the task is to predict the fraudulent and benign users, given some labels from both categories. The performance is measured using *area under the ROC curve (AUC)* which is a standard measures when data is imbalanced, as is our case. For each algorithm, a feature vector is created for each user and used to train a binary classifier. As a reminder from Section 4, for each user u, supervised REV2 creates a 2187 dimentional feature vector of its fairness scores $F(u|\alpha_1, \alpha_2, \beta_1, \beta_2, \gamma_1, \gamma_2, \gamma_3)$, one for each of the 2187 combinations of $\alpha_1, \alpha_2, \beta_1, \beta_2, \gamma_1, \gamma_2, \gamma_3 \in [0, 2]$. For fair comparison, baselines are compared by using their scores as features as well. For FraudEagle, BAD, SpEagle, SpEagle+, BIRDNEST and Trustiness, the feature vector for user u is the score the baseline

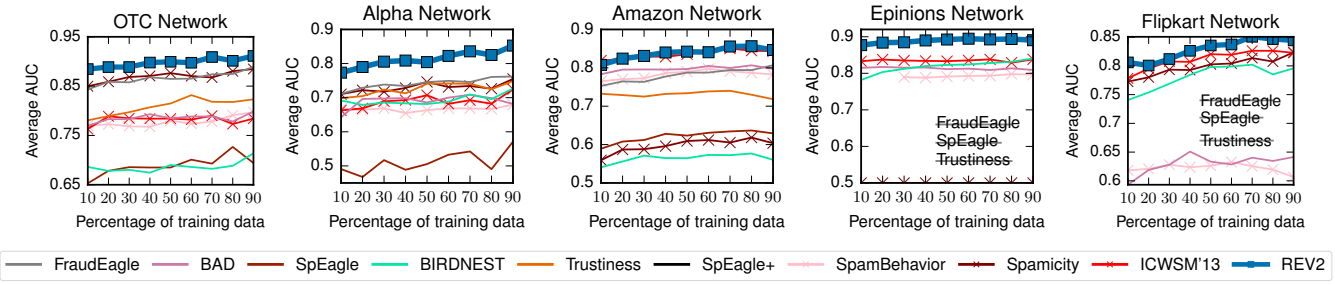
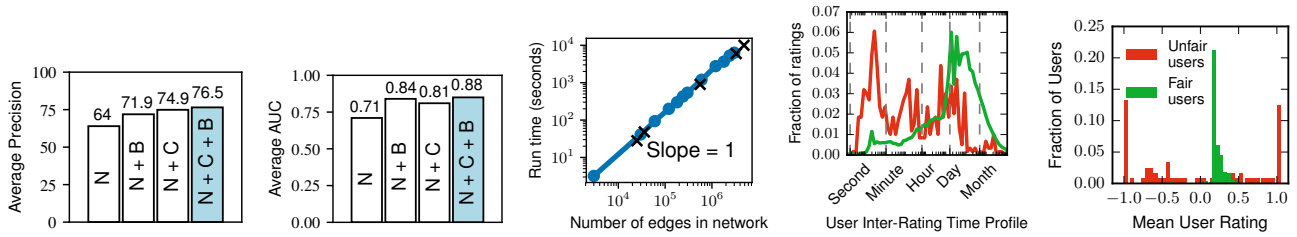


Figure 5: Variation of AUC for fraudulent user prediction with percentage of training data available for supervision. Rev2 consistently performs the best across all settings, and its performance is robust to the training percentage.



(a) Unsupervised Rev2 (b) Supervised Rev2 (c) Scalability of Rev2 (d) (e)

Figure 6: (a-b) Change in performance of Rev2 on Alpha network in (a) unsupervised and (b) supervised experiments when different components are used: network (N), cold start treatment (C) and behavioral (B). (c) Rev2 scales linearly—the running time increases linearly with the number of edges. (d-e) Fraudulent users identified by Rev2 in the OTC network are (d) faster in rating, and (e) give extreme ratings.

gives to u and u's outdegree. For baselines SpamBehavior, Spamicity and ICWSM'13, the feature vector for user u is their respective features explained in Section 5.2.

We perform stratified 10-fold cross-validation using random forest classifier. The resulting AUCs are reported in Table 4. We see that Rev2 outperforms all existing algorithms across all datasets and consistently has an AUC ≥ 0.85.

5.5 Experiment 3: Robustness of Rev2

In this experiment, we evaluate the performance of the algorithms as the percentage of training data changes. We vary the training data from 10% to 90% in steps of 10. Figure 5 shows the average AUC on test sets, averaged over 50 random iterations of training data. We make two observations. First, Rev2 is robust to the amount of training data. Its performance is relatively stable (AUC ≥ 0.80 in almost all cases) as the amount of training data varies. Second, Rev2 outperforms other algorithms consistently across all datasets for almost all training percentages. Together, these two show the efficiency of supervised Rev2 algorithm even when a small amount of training data is available.

5.6 Experiment 4: Importance of Network, Cold Start and Behavior

In this experiment, we show the importance of the different components in the Rev2 algorithm—network (given by equations 1, 2 and 3; shown as N), cold start treatment (C), and behavioral properties (B). Figure 6(a) shows the average precision in unsupervised case for the Alpha dataset, when network property is combined with the other two components, and Figure 6(b) shows the average AUC of 10-fold cross validation in the supervised case. In both cases, network properties alone has the lowest performance. Adding either the cold start treatment component or the behavioral property component to it increases this performance. To combine network properties with behavioral property component only, the cold start property is removed by setting $\alpha_2 = \beta_2 = \gamma_3 = 0$ in the Rev2 algorithm. Likewise, α_1 and β_1 are set to 0 to combine with cold start treatment alone. Further, combining all three together gives the best performance. Similar observations hold for the other datasets as well, as shown in the appendix [1]. This shows that all three components are important for predicting fraudulent users.

5.7 Experiment 5: Linear scalability

We have theoretically proven in Section 4.1 that Rev2 is linear in running time in the number of edges. To show this experimentally as well, we create random networks of increasing number of nodes and edges and compute the running time of the algorithm till convergence. Figure 6(c) shows that the running time increases linearly with the number of edges in the network, which shows that Rev2 algorithm is indeed scalable to large size networks for practical use. The figure also shows the running time of Rev2 on real networks, marked as 'x'.

5.8 Discoveries

In this part, we look at some insights and discoveries about fraudulent users found by Rev2.

As seen previously in Figure 2, most ratings given by fraudulent users have low reliability, while some have high reliability, indicating camouflage to masquerade as benign users. At the same time, most ratings by benign users have high reliability, but some ratings have low reliability, indicating biases and personal opinion about products that disagrees with the 'consensus'.

We also observe in Figure 6(d-e) that fraudulent users in OTC network: (d) give ratings in quick succession of less than a few minutes, and (e) exhibit bimodal rating pattern: they either give all

-1.0 ratings (possibly, bad-mouthing a competitor) or all +1.0 ratings (possibly, over-selling their products/friends). As an example, the most fraudulent user found by Rev2 had 3500 ratings, all with +0.5 score, and almost all given 15 seconds apart (apparently, a bot). On the other hand, benign users have a day to a month between consecutive ratings, and they give mildly positive ratings (between 0.1 and 0.5). These observations are coherent with existing research [16].

In summary, fraudulent users detected by Rev2 exhibit strange characteristics with respect to their behavior:

• They have bimodal rating pattern—they give too low (bad-mouthing) or too high (over-selling) ratings.

• They are less active, have no daily periodicity, and post quickly, often less than a few minutes apart.

• They camouflage their behavior to masquerade as benign users.

6 CONCLUSIONS

We presented the Rev2 algorithm to address the problem of identifying fraudulent users in rating networks. This paper has the following contributions:

• **Algorithm**: We defined three mutually-recursive metrics—fairness of users, goodness of products and reliability of ratings. We incorporated behavioral properties of users, ratings, and products in these metrics, and extended it to address cold start problem. We proposed the Rev2 algorithm to iteratively compute the values of these metrics.

• **Theoretical guarantees**: We proved that Rev2 algorithm has linear time complexity and is guaranteed to converge in a bounded number of iterations.

• **Effectiveness**: By conducting five experiments, we showed that Rev2 outperforms nine existing algorithms to predict fraudulent users. In unsupervised experiment, Rev2 is the best in 8 out of 10 cases and second best in the other two. In supervised experiment, Rev2 performs the best with AUC ≥ 0.85. Rev2 is practically useful, and already being deployed at Flipkart.

ACKNOWLEDGMENTS

Parts of this research is supported by the ARO Grant W911NF1610342, National Science Foundation Grants CNS-1314632, IIS-1408924, and Army Research Laboratory Cooperative Agreement Number W911NF-09-2-0053.

Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation, or other funding parties. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation here on.

REFERENCES

[1] Rev2 online appendix. https://cs.stanford.edu/~srijan/rev2/.

[2] L. Akoglu, R. Chandy, and C. Faloutsos. Opinion fraud detection in online reviews by network effects. In *International Conference on Web and Social Media*, 2013.

[3] L. Akoglu, H. Tong, and D. Koutra. Graph based anomaly detection and description: a survey. *ACM Transactions on Knowledge Discovery from Data*, 2015.

[4] C. Chen, K. Wu, V. Srinivasan, and X. Zhang. Battling the internet water army: Detection of hidden paid posters. In *International Conference on Advances in Social Networks Analysis and Mining*, 2013.

[5] A. Fayazi, K. Lee, J. Caverlee, and A. Squicciarini. Uncovering crowdsourced manipulation of online reviews. In *Special Interest Group on Information Retrieval*, 2015.

[6] 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 *International Conference on World Wide Web*, 2012.

[7] B. Hooi, N. Shah, A. Beutel, S. Gunneman, L. Akoglu, M. Kumar, D. Makhija, and C. Faloutsos. Birdnest: Bayesian inference for ratings-fraud detection. In *SIAM International Conference on Data Mining*, 2016.

[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 *ACM International conference on Knowledge Discovery and Data Mining*, 2016.

[9] C. J. Hutto and E. Gilbert. Vader: A parsimonious rule-based model for sentiment analysis of social media text. In *Eighth international AAAI conference on weblogs and social media*, 2014.

[10] M. Jiang, P. Cui, A. Beutel, C. Faloutsos, and S. Yang. Catchsync: catching synchronized behavior in large directed graphs. In *ACM International Conference on Knowledge Discovery and Data Mining*, 2014.

[11] M. Jiang, P. Cui, and C. Faloutsos. Suspicious behavior detection: Current trends and future directions. *IEEE Intelligent Systems*, 31(1):31–39, 2016.

[12] S. Kumar, J. Cheng, J. Leskovec, and V. Subrahmanian. An army of me: Sockpuppets in online discussion communities. In *International Conference on World Wide Web*, 2017.

[13] S. Kumar and N. Shah. False information on web and social media: A survey. *Social Media Analytics: Advances and Applications*, 2018.

[14] S. Kumar, F. Spezzano, V. Subrahmanian, and C. Faloutsos. Edge weight prediction in weighted signed networks. In *IEEE 16th International Conference on Data Mining*, 2016.

[15] T. Lappas, G. Sabnis, and G. Valkanas. The impact of fake reviews on online visibility: A vulnerability assessment of the hotel industry. *INFORMS*, 27(4), 2016.

[16] H. Li, G. Fei, S. Wang, B. Liu, W. Shao, A. Mukherjee, and J. Shao. Bimodal distribution and co-bursting in review spam detection. In *International Conference on World Wide Web*, 2017.

[17] R.-H. Li, J. Xu Yu, X. Huang, and H. Cheng. Robust reputation-based ranking on bipartite rating networks. In *SIAM International Conference on Data Mining*, 2012.

[18] E.-P. Lim, V.-A. Nguyen, N. Jindal, B. Liu, and H. W. Lauw. Detecting product review spammers using rating behaviors. In *International Conference on Information and Knowledge Management*, 2010.

[19] P. Massa and P. Avesani. Trust-aware recommender systems. In *ACM Conference on Recommender Systems*, 2007.

[20] J. J. McAuley and J. Leskovec. From amateurs to connoisseurs: modeling the evolution of user expertise through online reviews. In *International Conference on World Wide Web*, 2013.

[21] A. J. Minnich, N. Chavoshi, A. Mueen, S. Luan, and M. Faloutsos. Trueview: Harnessing the power of multiple review sites. In *International Conference on World Wide Web*, 2015.

[22] A. Mishra and A. Bhattacharya. Finding the bias and prestige of nodes in networks based on trust scores. In *International World Wide Web conference*, 2011.

[23] A. Mukherjee, A. Kumar, B. Liu, J. Wang, M. Hsu, M. Castellanos, and R. Ghosh. Spotting opinion spammers using behavioral footprints. In *ACM International conference on Knowledge Discovery and Data Mining*, 2013.

[24] A. Mukherjee, V. Venkataraman, B. Liu, and N. S. Glance. What yelp fake review filter might be doing? In *International Conference on Web and Social Media*, 2013.

[25] J. W. Pennebaker, M. E. Francis, and R. J. Booth. Linguistic inquiry and word count: Liwc 2001. *Mahway: Lawrence Erlbaum Associates*, 71(2001):2001, 2001.

[26] S. Rayana and L. Akoglu. Collective opinion spam detection: Bridging review networks and metadata. In *ACM International conference on Knowledge Discovery and Data Mining*, 2015.

[27] V. Sandulescu and M. Ester. Detecting singleton review spammers using semantic similarity. In *International Conference on World Wide Web*, 2015.

[28] V. Subrahmanian and S. Kumar. Predicting human behavior: The next frontiers. *Science*, 355(6324):489–489, 2017.

[29] H. Sun, A. Morales, and X. Yan. Synthetic review spamming and defense. In *ACM International conference on Knowledge Discovery and Data Mining*, 2013.

[30] B. Viswanath, M. A. Bashir, M. Crovella, S. Guha, K. P. Gummadi, B. Krishnamurthy, and A. Mislove. Towards detecting anomalous user behavior in online social networks. In *USENIX Security*, 2014.

[31] B. Viswanath, M. A. Bashir, M. B. Zafar, S. Bouget, S. Guha, K. P. Gummadi, A. Kate, and A. Mislove. Strength in numbers: Robust tamper detection in crowd computations. In *Conference on Online Social Networks*, 2015.

[32] G. Wang, S. Xie, B. Liu, and S. Y. Philip. Review graph based online store review spammer detection. In *IEEE International Conference on Data Mining series*, 2011.

[33] G. Wang, S. Xie, B. Liu, and P. S. Yu. Identify online store review spammers via social review graph. *ACM Transactions on Intelligent Systems and Technology*, 3(4):61, 2012.

[34] J. Wang, A. Ghose, and P. Ipeirotis. Bonus, disclosure, and choice: what motivates the creation of high-quality paid reviews? In *International Conference on Information Systems*, 2012.

[35] G. Wu, D. Greene, and P. Cunningham. Merging multiple criteria to identify suspicious reviews. In *ACM Conference on Recommender Systems*, 2010.

[36] Z. Wu, C. C. Aggarwal, and J. Sun. The troll-trust model for ranking in signed networks. In *ACM International Conference on Web Search and Data Mining*, 2016.

[37] S. Xie, G. Wang, S. Lin, and P. S. Yu. Review spam detection via temporal pattern discovery. In *ACM International Conference on Knowledge Discovery and Data Mining*, 2012.