Fraud De-Anonymization For Fun and Profit

Nestor Hernandez
FIU, Miami, USA
nestorghh@gmail.com

Mizanur Rahman
FIU, Miami, USA
mrahm031@fiu.edu

Ruben Recabarren
FIU, Miami, USA
recabarren@gmail.com

Bogdan Carbunar
FIU, Miami, USA
carbunar@gmail.com

ABSTRACT

The persistence of search rank fraud in online, peer-opinion systems, made possible by crowdsourcing sites and specialized fraud workers, shows that the current approach of detecting and filtering fraud is inefficient. We introduce a fraud de-anonymization approach to disincentivize search rank fraud: attribute user accounts flagged by fraud detection algorithms in online peer-opinion systems, to the human workers in crowdsourcing sites, who control them. We model fraud de-anonymization as a maximum likelihood estimation problem, and introduce UODA, an unconstrained optimization solution. We develop a graph based deep learning approach to predict ownership of account pairs by the same fraudster and use it to build discriminative fraud de-anonymization (DDA) and pseudonymous fraudster discovery algorithms (PFD).

To address the lack of ground truth fraud data and its pernicious impacts on online systems that employ fraud detection, we propose the first cheating-resistant *fraud de-anonymization validation* protocol, that transforms human fraud workers into ground truth, performance evaluation oracles. In a user study with 16 human fraud workers, UODA achieved a precision of 91%. On ground truth data that we collected starting from other 23 fraud workers, our co-ownership predictor significantly outperformed a state-of-the-art competitor, and enabled DDA and PFD to discover tens of new fraud workers, and attribute thousands of suspicious user accounts to existing and newly discovered fraudsters.

CCS CONCEPTS

• **Security and privacy → Social network security and privacy**; **Social aspects of security and privacy**; • **Information systems →** *Incentive schemes*;

KEYWORDS

Fraud De-Anonymization; Search Rank Fraud; Crowdturfing; Fake Review; Opinion Spam; Sybil Attack; App Store Optimization

ACM Reference Format:
Nestor Hernandez, Mizanur Rahman, Ruben Recabarren, and Bogdan Carbunar. 2018. Fraud De-Anonymization For Fun and Profit. In *2018 ACM SIGSAC Conference on Computer and Communications Security (CCS '18), October 15–19, 2018, Toronto, ON, Canada.* ACM, New York, NY, USA, 16 pages. https://doi.org/10.1145/3243734.3243770

1 INTRODUCTION

Popular online service providers rely on user feedback to rank products and content they host over the Internet. Unfortunately, many review-based platforms (e.g., Google Play [59], TripAdvisor [60], Amazon [78], Twitter [22]) are the targets of undisclosed and deceptive marketing practices whereby product developers engage in fake endorsement either to boost their products or to demote those of a competitor. Black hat crowdsourcing or *crowdturfing* offers an economically viable opportunity for developers to hire specialized workers who spam for profit [42, 71, 74, 76, 82].

This type of propaganda has a detrimental effect on the trustworthiness and quality of online services, and users can suffer from such bait-and-switch schemes. For this reason, most major online, peer-opinion services seek to detect and remove fake reviews that result from hidden endorsements [49, 50, 65], which are unlawful in accordance with FTC regulations [1]. Significant academic work on defenses against online fraud has focused on a binary classification of reviews as fake or honest [18, 27, 31, 36, 37, 41, 43, 44, 47, 58, 61, 67, 71, 77, 79, 83], and of reviewers as fraudulent (Sybil) or genuine [13, 20, 23, 40, 45, 81, 84, 87].

Fraud detection solutions however are not only (1) ineffective in preventing fraud, as observed from the continued profitability of fraud in online services and crowdsourcing platforms but also (2) their accuracy is difficult to evaluate, given that collecting ground truth fraud data is a notoriously hard task.

Services like Yelp acknowledge the validation problem, by not removing but only making suspected fake reviews harder to access, and moving reviews back and forth between the fake and honest classes according to subsequent iterations of their detection algorithms [6, 49]. To address this problem, academic work has built gold standard fraud datasets using rule-based heuristics, assuming that e.g., fraudsters post reviews in a short period of time [32, 33, 40, 44, 87], from the same IP address [40], or have a skewed rating distribution [58, 87]. However, such assumptions are also difficult to validate, especially as they are straightforward to

CCS '18, October 15–19, 2018, Toronto, ON, Canada

ACM ISBN 978-1-4503-5693-0/18/10…$15.00
https://doi.org/10.1145/3243734.3243770

[1] If the endorser has been paid or given something of value to promote the product, the connection between the marketer and endorser should be disclosed [2]

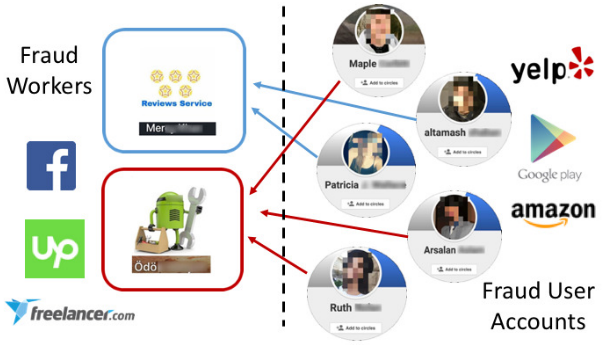


Figure 1: DETEGO de-anonymizes fraud. Fraud detection only identifies suspicious user accounts on the right. Fraud de-anonymization also finds the crowdsourcing account (left side) that controls them. Arrows signify control.

bypass by experienced fraudsters (e.g., using proxies, better distributing the post time and rating of reviews). In this paper, we take steps toward addressing these problems.

Addressing inefficacy. In this paper, we propose to discourage fraud instead of merely discovering it. To this end, as illustrated in Figure 1, we seek to bridge the anonymity gap between existing fraud detection techniques, that only uncover pseudonymous user accounts that post fraud, and the real identities of crowdsourcing site accounts who control them. Specifically, we leverage the observation that crowdsourcing site accounts contain uniquely identifying payment information, e.g., bank, Paypal accounts, to take steps toward de-anonymizing fraud, by attributing accounts uncovered by fraud detection algorithms in online peer-opinion systems, to their human owners in crowdsourcing sites.

We propose a general theoretical framework for the fraud de-anonymization problem via Maximum Likelihood Estimation (MLE) and assume a generative review-posting model wherein fraudster-controlled accounts are more likely to endorse products in a predefined partition of the product space. We introduce UODA, an unconstrained optimization de-anonymization approach that attributes a fraudulent user account to the fraud worker with the highest likelihood of having generated its review history.

We develop DeepCluster, a semi-supervised approach to cluster user accounts based on deep learning features extracted from the common activities of the accounts. We leverage DeepCluster to build a *co-ownership predictor* that determines if two input user accounts are controlled by the same worker. We use the co-ownership predictor to introduce (1) DDA, a discriminative de-anonymization solution that trains a classifier to attribute a fraudulent user account to the worker who controls it, and (2) PFD, a pseudonymous fraudster discovery algorithm that clusters fraudulent accounts that cannot be attributed to known workers, such that each cluster is likely controlled by a different, not yet discovered worker.

We introduce DETEGO [2], a system that combines fraud de-anonymization with fraudster discovery to iteratively expand both knowledge of identifiable fraud workers and the accounts that they control. We believe that DETEGO can help peer-review sites identify the experts from among hundreds of advertised fraud workers, who control large numbers of user accounts, and are responsible for posting substantial numbers of fake reviews. Peer-review sites can use this information to provide counter-incentives for expert fraudsters, e.g., by pursuing them through their bank accounts (retrieved from their crowdsourcing site accounts). Peer-review sites can also disincentivize developers from hiring such identifiable fraudsters, e.g., by "shaming" promoted products with posts displaying information about the fraudsters found to promote them [4].

Addressing validation. We introduce the first cheating-resistant, *fraud de-anonymization validation* protocol, to obtain ground truth confirmation on the performance of developed solutions. The protocol asks human fraud workers to reveal a seed set of user accounts that they control, and subsequently confirm and prove control of accounts that we predict that they control. We introduce multiple verifications of participant attention and honesty, including asking confirmations for accounts for which we already know the answer, as well as e-mail and token based verifications.

Results. We conducted the fraud de-anonymization validation protocol, through a user study with 16 human fraud workers, who revealed control of a total of 230 Google Play accounts. The participants confirmed control of 91% of the user accounts newly discovered by UODA. Further, on 942 ground truth attributed user accounts that we collected from other 23 fraud workers, both DDA and UODA achieved precision and recall that exceed 90%, and attributed thousands of new accounts to these fraudsters.

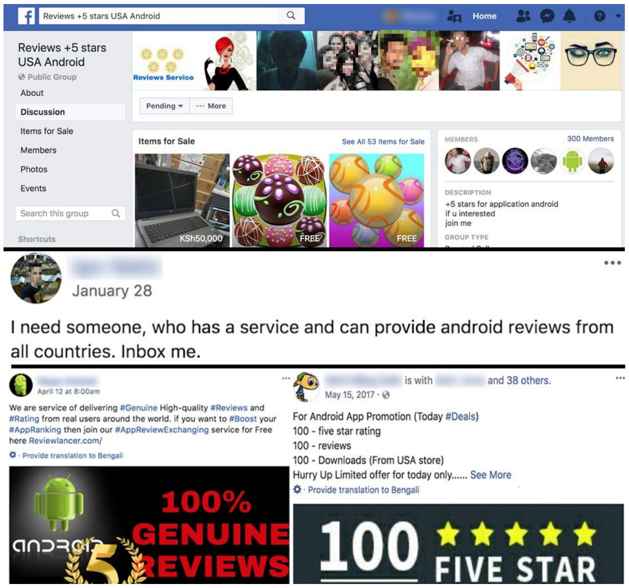
We introduce intuition, and empirically evaluate the impact of features used by our co-ownership predictor. Our predictor outperformed the F1-measure of state-of-the-art, ELSIEDET's Sybil social link builder [87] by more than 12 percentage points, on ground truth attributed data. Further, the PFD algorithm identified thousands of accounts not previously known to be fraudulent, grouped into communities according to common ownership by fraudsters. We analyzed 1.1 billion pairs of reviews from these communities and report orthogonal evidence of fraud, including communities with more than 80% of accounts involved in review text plagiarism. In summary, our contributions are the following:

- **Fraud de-anonymization**. Model fraud de-anonymization as a maximum likelihood estimation problem. Develop UODA, an unconstrained optimization fraud de-anonymization algorithm [§ 4].
- **Co-ownership predictor**. Introduce a graph based deep learning approach to predict ownership of account pairs by the same fraudster [§ 5]. Leverage the predictor to build DDA, a discriminative fraud de-anonymization [§ 6] and PFD, a pseudonymous fraudster discovery algorithm [§ 7].
- **Human fraud de-anonymization oracles**. Develop the first protocol to provide human-fraud-worker-based performance evaluation of fraud de-anonymization algorithms [§ 9]. Evaluate proposed solutions using data collected through this protocol [§ 11].

2 CONCEPTS AND BACKGROUND

In this section, we first formally define the basic terminology used throughout the paper and then provide background details about fraud in peer-opinion systems.

[2] In Latin, *detego* means to uncover, reveal.



+5 stars for application android
if u interested
join me

[redacted]
January 28

I need someone, who has a service and can provide android reviews from all countries. Inbox me.

[redacted]
April 12 at 8:00am

We are service of delivering #Genuine High-quality #Reviews and #Rating from real users around the world. if you want to #Boost your #AppRanking then join our #AppReviewExchanging service for Free here Reviewlancer.com/

Provide translation to Bengali

[redacted] is with [redacted] and 38 others.
May 15, 2017 ·

For Android App Promotion (Today #Deals)
100 - five star rating
100 - reviews
100 - Downloads (From USA store)
Hurry Up Limited offer for today only...... See More

Provide translation to Bengali

Figure 2: Anonymized screenshots of search rank fraud from Facebook. (Top) Page of Facebook group dedicated to search rank fraud. (Middle) Recruitment post from developer. (Bottom) Posts of fraud workers.

2.1 Basic Terminology

User. A person or entity who posts reviews about a *subject* on an online peer-opinion system. Users make use of *user accounts* to establish their identity online.

Subject. A *developer* created object or product that receives *user* created reviews on the peer-opinion system.

Developer. A person or entity that hosts *subjects* on the peer-opinion system. Developers usually have incentives to maximize their subject's visibility via review manipulation for which they hire *workers*. Thus, we also refer to developers as *employers*.

Fraud worker. A person or entity that performs review manipulation about a *subject* on behalf of a *developer*. Workers often use *Sybil accounts* to post fraudulent reviews on the peer-opinion system.

2.2 System and Adversary Model

We consider online peer-opinion systems, e.g., Google Play, Yelp, Amazon, that host accounts for developers, users and products. Developers use their accounts to upload information about products while users are expected to post reviews only for products they have used. The survival of products in peer-opinion services is contingent on their review influenced search rank. Higher ranked products are acquired more frequently and generate more revenue, either through direct payments or ads. For example, a one star boost in rating was shown to help restaurants increase revenue by a 5-9% margin [46]. While online systems keep their ranking algorithms secret for security reasons [65], popular belief claims that large numbers of positive reviews and installs help products achieve higher search rank [5].

Fraud Origin. The pressure to succeed has created a black market for *search rank fraud*. Specialized fraud workers (also referred to as fraud freelancers, or fraudsters) control multiple user accounts and seek employment by product developers to post fake reviews or activities for their products. The accounts controlled by a fraud worker are also known as Sybils or sockpuppets [13, 23, 40, 45, 81, 84, 85, 87]. Fraud workers advertise their services through crowdsourcing sites [1, 3, 28], social networks (e.g., Facebook groups), and specialized fraud sites [7–11]. Moreover, fraudulent activities are profitable as evidenced by their price ranges. For instance, we identified 44 fraud workers in Facebook groups, Zeerk, Peopleperhour, Freelancer and Upwork that advertised prices ranging from a few cents ($0.56 on average from Zeerk.com) to several dollars per review (up to $10 in Freelancer.com) [56].

Facilitating Fraud. Crowdsourcing sites like *Fiverr*, *Upwork* and *Freelancer* [1, 3, 28] host accounts for *workers* and *employers*. These crowdsourcing accounts have a unique identifier and require a linked bank account for depositing employer's escrow money or withdrawing worker's earnings. Workers on these sites bid on employer posted *jobs* while employers assign jobs to workers after successful negotiation. Thus, these crowdsourcing sites provide a comprehensive platform for performing peer-opinion system fraud.

In addition, workers can also advertise on social networks where they usually encounter no restriction to use keywords associated with search rank fraud and other blackhat services. As a consequence, social networks like Facebook provide high visibility to these services due to their large user base (see Figure 2 for sample snapshots). Furthermore, Facebook groups specializing in search rank fraud efficiently enable developers and fraud workers to find each other and communicate through posts and comments.

Moreover, fraud workers can also create their own service advertising pages hoping that developers discover them using keyword search on Internet search engines.

Effective fraud. In a separate Upwork data set experiment, we collected 161 search rank fraud jobs and their 533 bidding workers. We found that jobs assigned to a single worker occurred less frequently than jobs awarded to 2 workers. Furthermore, some developers assigned a single job to as many as 12 workers. We conjecture that this assignment distribution occurs due to the limited ability of a single worker to effect a significant impact over a subject's search rank. This observation reveals that subjects targeted by search rank fraud will usually receive fake reviews from multiple fraud workers.

3 PROBLEM DEFINITION

The insight that multiple fraud workers usually target a single subject suggests that a binary classification of fraud, e.g., fake vs. honest reviews, fraudulent vs. honest accounts [17, 26, 27, 31, 32, 44, 47, 80], is insufficient to understand and model fraud. Instead, we study the *fraud de-anonymization problem* which deals with attributing fraudulent accounts and fake reviews to the crowdsourcing accounts of the fraud workers who control and post them, respectively.

Formally, let $\mathcal{U}$ be the set of all user accounts, and let $\mathcal{S}$ be the set of all subjects hosted in the online peer-opinion system. We say that a user account is fraudulent or *fraudster-controlled* if it was opened by a fraudster to mainly perform fraudulent activities in the online system, i.e., to target subjects from $\mathcal{S}$.

Moreover, let $U^* \subseteq \mathcal{U}$ be the set of all fraudster-controlled accounts in an online system, and let $\mathcal{W}$ be the set of all fraud worker accounts in crowdsourcing sites. In addition, let $W^* = \{(W_l, U_l, S_l) | \; W_l \in \mathcal{W}, U_l \subseteq U^*, S_l \subseteq \mathcal{S}, l = 1 \ldots f\} \subset \mathcal{V}$ be a known set of f search rank fraud worker profiles where $\mathcal{V}$ is the universe of all worker profiles. A profile consists of a crowdsourcing account id (W_l), an incomplete set of user accounts (U_l) known to be controlled by W_l in the peer-opinion system, and the incomplete set of subjects (S_l) known to have been fraudulently reviewed by W_l. Section 9 describes a protocol to identify crowdsourced fraud workers and build seed profiles for them.

Ideally, we want to attribute each account in U^* to the fraudster who controls it. However, some accounts in U^* may not be controlled by any of the known fraudsters in W^*. To address this issue, we formulate two distinct problems: fraud de-anonymization and pseudonymous fraudster discovery:

Fraud De-Anonymization. Build a function $FDA \colon U^* \setminus \cup_{l=1}^{f} U_l \mapsto W^*$, that, given a user account $u \in U^*$ suspected of participation in search rank fraud, returns the fraud worker in W^* most likely to control u. In Section 4.1 we expand this definition in a maximum likelihood estimation (MLE) based framing of the problem.

Pseudonymous Fraudster Discovery. Build a function $PFD \colon U^* \setminus \cup_{l=1}^{f} U_l \mapsto \mathcal{V} \setminus W^*$ that, given a set of fraudster-controlled accounts that were not assigned to one of the known fraudsters by the FDA function, returns a new set of fraudster profiles from $\mathcal{V} \setminus W^*$ that control these accounts.

Unlike standard de-anonymization, the adversarial process of identifying users from data where their Personally Identifiable Information (PII) has been removed [52], the fraud de-anonymization problem seeks to attribute detected search rank fraud to the humans who posted it. A solution to this problem will enable peer-review services to identify the impactful crowdsourcing fraudsters who target them, and provide appealing fraud feedback proof to customers, e.g., links to the crowdsourcing accounts responsible for boosting a product's rating. Furthermore, accurate fraud de-anonymization will allow online services and law enforcement to retrieve banking information and real identities of fraudsters. Thus, fraud de-anonymization may provide counter-incentives for crowdsourcing workers to participate in fraud jobs, and for product developers to recruit them.

In Section 4 and 6, we introduce unconstrained optimization and discriminative fraud de-anonymization algorithms, respectively, while in Section 7 we propose a pseudonymous fraudster discovery algorithm. In Section 8, we show how Detego iteratively invokes a pseudonymous fraudster discovery algorithm followed by a fraudster de-anonymization algorithm, to expand knowledge of fraud workers and the accounts they control.

4 UNCONSTRAINED OPTIMIZATION BASED DE-ANONYMIZATION

We first propose a maximum likelihood based de-anonymization approach motivated by a realistic generative model of review posting behavior. Next, we compute the likelihood of each worker having generated a given suspicious fraudulent review history. We then find the worker who maximizes such likelihood.

4.1 Definitions and Approach

We postulate a probabilistic review-posting model from accounts controlled by fraudsters, inspired by Su et al. [69]. Specifically, we assume that a fraudulent account u controlled by a fraudster profile $(W, U, S) \in W^*$ is likely to review subjects in a pairwise-disjoint family of sets over $\mathcal{S}$, $\mathcal{F}_W = \{\Omega_1, \Omega_2, \ldots, \Omega_m\}$ ($\Omega_i \cap \Omega_j = \emptyset \; \forall \; i \neq j$) with different multiplicative factors $r_1, r_2, \ldots, r_m$ describing u's responsiveness to each Ω_i. Further, we assume that the review history of a user account is described by a sequence of independent and identically distributed random variables $R_1, R_2, \ldots, R_n$ where $R_k \in \mathcal{S}$ represents the k-th subject reviewed from the account. Therefore, a fraudulent account's review posting behavior is characterized by $\mathcal{F}_W$and r_i for all $i = 1 \ldots m$.

Let $\{p_j\}$ be a probability measure over the sample space $\mathcal{S}$, related to the popularity of the subjects: $p_j \geq 0, \sum_{j=1}^{|\mathcal{S}|} p_j = 1$. For any fraudster profile $(W, U, S) \in W^*$, we define random variable $R_k(\mathcal{F}_{\mathbf{W}}, \mathbf{r})$ with values in $\mathcal{S}$ and with the probability distribution:

$$\mathbb{P}(R_k = s_j) = \begin{cases} \frac{r_1 p_j}{c} & \text{if } s_j \in \Omega_1 \\ \ldots & \\ \frac{r_m p_j}{c} & \text{if } s_j \in \Omega_m \\ \frac{p_j}{c} & \text{if } s_j \in \bigcap_{i=1}^{m} \Omega_i^C \end{cases} \tag{1}$$

where $c = \sum_{i=1}^{m} r_i \sum_{s_j \in \Omega_i} p_j + \sum_{s_j \in \bigcap_{i=1}^{m} \Omega_i^C} p_j$ and $\mathbf{r} = [r_1, \ldots, r_m]^\intercal$ is the vector of multiplicative factors. Specifically, the probability that the k-th review targets subject s_j is proportional to factor r_m if subject s_j satisfies Ω_m's membership properties. Otherwise, this probability is simply given by the ratio p_j/c.

Let $R_1(\mathcal{F}_{\mathbf{W}}, \mathbf{r})$, $R_2(\mathcal{F}_{\mathbf{W}}, \mathbf{r})$, …, $R_n(\mathcal{F}_{\mathbf{W}}, \mathbf{r})$, be a review history suspected to be fraudulent. Given a set of candidate workers, each described by a family of sets $\mathcal{F}_{\mathbf{W}}$, the fraudster de-anonymization problem derives the maximum likelihood estimates $\hat{\mathbf{r}}$ and $\hat{\mathcal{F}_{\mathbf{W}}}$ of the function:

$$\mathcal{L}(\mathcal{F}_{\mathbf{W}}, \mathbf{r}) = \left(\prod_{i=1}^{m} \prod_{R_k \in \Omega_i} \mathbb{P}(R_k \mid \mathcal{F}_{\mathbf{W}}, \mathbf{r}) \right) \prod_{R_k \in \bigcap_{i=1}^{m} \Omega_i^C} \mathbb{P}(R_k \mid \mathcal{F}_{\mathbf{W}}, \mathbf{r}) \tag{2}$$

where $\hat{\mathcal{F}_{\mathbf{W}}}$ is the family of sets associated with the worker most likely linked with the given review history.

4.2 UODA

We introduce UODA, an unconstrained optimization based de-anonymization approach that maximizes the function in Equation (2) without constraints on the multiplicative values $r_1, \ldots, r_m$. Theorem 4.1 characterizes the solution for the fraudster de-anonymization problem under this unconstrained setting.

Theorem 4.1. *Let $\mathcal{S}$ be the set of subjects hosted by the online service, and $\{p_j\}$ be a probability measure on $\mathcal{S}$ ($p_j \geq 0, \sum_{j=1}^{|\mathcal{S}|} p_j = 1$). Let $C = \{\mathcal{F}_{W_1}, \ldots, \mathcal{F}_{W_f}\}$ be a collection of family sets for each fraud worker, where $\mathcal{F}_{W_l}$=$\{\Omega_{l1}$, Ω_{l2}, …, $\Omega_{lm}\}$. For any $\mathcal{F}_W \in C$, define a random variable $R_k(\mathcal{F}_{\mathbf{W}}, \mathbf{r})$ taking values in $\mathcal{S}$ and obeying the probability distribution in Equation* (1). *Given a review history*

$R_1(\mathcal{F}_\mathbf{W}, \mathbf{r}), R_2(\mathcal{F}_\mathbf{W}, \mathbf{r}), \ldots, R_n(\mathcal{F}_\mathbf{W}, \mathbf{r})$ *suspected to be fraudulent, the maximum likelihood estimates* $\hat{\mathbf{r}}$ *and* $\hat{\mathcal{F}_\mathbf{W}}$ *are:*

$$\hat{r}_t = \frac{q_t\left(1-\sum_{i=1}^m P_i\right)}{P_t\left(1-\sum_{i=1}^m q_i\right)} \quad \text{for } t = 1,\ldots,m \tag{3}$$

and

$$\hat{\mathcal{F}_\mathbf{W}} = \operatorname*{argmax}_{\mathcal{F}_W\in C}\left[\sum_{i=1}^m q_i \ln\left(\frac{q_i}{P_i}\right) - \left(1-\sum_{i=1}^m q_i\right)\ln\left(\frac{1-\sum_{i=1}^m P_i}{1-\sum_{i=1}^m q_i}\right)\right] \tag{4}$$

where $q_i = |\{k \mid R_k \in \Omega_i\}|/n$ *and* $P_i = \sum_{s_j\in\Omega_i} p_j$ for $i = 1,\ldots,m$

Intuition. Equation (4) from Theorem 4.1 attributes a user account to the worker profile in W^* most likely responsible for the account's review history $R_1(\mathcal{F}_\mathbf{W}, \mathbf{r}), R_2(\mathcal{F}_\mathbf{W}, \mathbf{r}), \ldots, R_n(\mathcal{F}_\mathbf{W}, \mathbf{r})$. The Ω sets partition worker's reviews into groups of subjects that have different characteristics (features). q_i is the fraction of subjects in the account's review history that are in the investigated worker's Ω_i. P_i is the total popularity of all the subjects in the set Ω_i. The first term of Equation (4) reveals that the $\hat{\mathcal{F}_\mathbf{W}}$ associated worker most likely to control the suspect account has a family of Ω sets for which most of q_i are large and P_i are small; that is, many of the subjects in the account's review history appear in the worker's sets Ω_i that are neither too big or popular.

Proof. Setting $R_k = s_k$, we rewrite Equation (2) as:

$$\mathcal{L}(\mathcal{F}_\mathbf{W}, \mathbf{r}) = \prod_{k=1}^n\left(\sum_{i=1}^m \frac{r_i p_k}{c} X_{\Omega_i}(s_k) + \frac{p_k}{c} X_{\bigcap_{i=1}^m \Omega_i^C}(s_k)\right)$$

when using indicator functions $X_{\Omega_i}(s)$ for $i = 1,\ldots,m$, i.e. $X_{\Omega_i}(s) = 1$ if $s \in \Omega_i$, and $X_{\Omega_i}(s) = 0$ otherwise. We can then write the log-likelihood function as follows:

$$\begin{aligned}
\ln\mathcal{L}(\mathcal{F}_\mathbf{W}, \mathbf{r}) &= \sum_{k=1}^n \ln\left(\sum_{i=1}^m \frac{r_i p_k}{c} X_{\Omega_i}(s_k) + \frac{p_k}{c} X_{\bigcap_{i=1}^m \Omega_i^C}(s_k)\right)\\
&= \sum_{k=1}^n\left(\sum_{i=1}^m X_{\Omega_i}(s_k)\ln\left(\frac{r_i p_k}{c}\right) + X_{\bigcap_{i=1}^m \Omega_i^C}(s_k)\ln\left(\frac{p_k}{c}\right)\right)\\
&= n\left(\sum_{i=1}^m q_i\ln(r_i) + \ln(p_k) - \ln(c)\right)
\end{aligned}$$

We can further rewrite c:

$$c = \sum_{i=1}^m r_i \sum_{s_j\in\Omega_i} p_j + \sum_{s_j\in\bigcap_{i=1}^m \Omega_i^C} p_j = \sum_{i=1}^m P_i(r_i - 1) + 1$$

Therefore,

$$\ln\mathcal{L}(\mathcal{F}_\mathbf{W}, \mathbf{r}) = n\left(\sum_{i=1}^m q_i\ln(r_i) + \ln(p_k) - \ln\left(\sum_{i=1}^m P_i(r_i - 1) + 1\right)\right)$$

The first-order necessary conditions are:

$$\frac{\partial\ln\mathcal{L}(\mathcal{F}_\mathbf{W}, \mathbf{r})}{\partial r_i} = \frac{-nP_i}{\sum_{i=1}^m P_i(r_i - 1) + 1} + \frac{nq_i}{r_i} = 0 \quad \text{for } i \in [m] \tag{5}$$

We can also write (5) as the $m \times m$ non-homogeneous system of linear equations:

$$[P_i(1-q_i)]r_i - q_i\sum_{d\neq i} P_d r_d = q_i\left(1 - \sum_{i=1}^m P_i\right) \quad \text{for } i \in [m] \tag{6}$$

To solve the system of equations (6), we introduce the following lemma, whose proof is in Appendix A.

Lemma 4.2. *The system of linear equations*

$$[P_i(1-q_i)]r_i - q_i\sum_{d\neq i} P_d r_d = q_i\left(1 - \sum_{i=1}^m P_i\right) \quad \text{for } i \in [m]$$

has solutions given by $r_t = \frac{q_t(1-\sum_{i=1}^m P_i)}{P_t(1-\sum_{i=1}^m q_i)}$

This enables us to write c as:

$$\begin{aligned}
c &= \sum_{i=1}^m P_i(r_i - 1) + 1\\
&= \sum_{i=1}^m P_i\left(\frac{q_i}{P_i}\frac{(1-\sum_{i=1}^m P_i)}{(1-\sum_{i=1}^m q_i)} - 1\right) + 1\\
&= \frac{\sum_{i=1}^m q_i(1-\sum_{i=1}^m P_i) + (1-\sum_{i=1}^m q_i)(1-\sum_{i=1}^m P_i)}{1-\sum_{i=1}^m q_i}\\
&= \frac{1-\sum_{i=1}^m P_i}{1-\sum_{i=1}^m q_i}
\end{aligned}$$

Thus, the value of $\mathbf{r}$ at which $\ln\mathcal{L}(\mathcal{F}_\mathbf{W}, \mathbf{r})$ reaches its maximum must also maximize the function $L(\mathcal{F}_\mathbf{W}, \mathbf{r})$ defined as:

$$\begin{aligned}
L(\mathcal{F}_\mathbf{W}, \mathbf{r}) &= \sum_{i=1}^m q_i\ln(r_i) - \ln(c)\\
&= \sum_{i=1}^m q_i\ln(r_i) - \ln\left(\frac{1-\sum_{i=1}^m P_i}{1-\sum_{i=1}^m q_i}\right)\\
&= \sum_{i=1}^m q_i\ln\left(\frac{q_i}{P_i}\right) - \left(1-\sum_{i=1}^m q_i\right)\ln\left(\frac{1-\sum_{i=1}^m P_i}{1-\sum_{i=1}^m q_i}\right)
\end{aligned}$$

□

In Section 11.2 we instantiate UODA for two features that define the Ω sets.

5 CO-OWNERSHIP PREDICTOR

We develop a co-ownership predictor function $cowPred\colon \mathcal{U}\times\mathcal{U} \mapsto \{0,1\}$ that determines if two user accounts are controlled by the same fraud worker. Specifically, given two user accounts u_i and u_j, $cowPred(u_i, u_j) = 1$ if u_i and u_j are controlled by the same fraudster. *cowPred* uses several features, that model similarity of behaviors between the input accounts. One such feature is extracted by DeepCluster, a semi supervised learning approach that we propose to cluster user accounts.

Algorithm 1: DeepCluster identifies communities of fraudulent accounts who targeted input subjects $s_1, .., s_k$, based on the similarity of their DeepWalk features extracted from the union fraud graph of the subjects.

```
   Input  : $CoR[1 \dots k]$; # Co-review graphs of reviewers of
            subjects $s_1, \dots, s_k$;
            $DWParams$; # Best DeepWalk parameters;
            $UFG$; # Union Fraud Graph over CoR[];
   Output : $clusters[1 \dots k][\,]$; # Best clusters for $s_1, \dots, s_k$
1  $UFeatures[\,][\,] = UFG.DWFeatures(DWParams)$
2  for $i = 1$ to $k$ do
3      $candidates[\,][\,] = CoR[i].V \ltimes UFeatures$
4      $candidates[\,][\,] = FilterHonest(candidates[\,][\,])$
5      $clusters[i] = getBestClusters(candidates)$
6  end
7  return $clusters[\,][\,]$
```

5.1 DeepCluster

DeepCluster leverages DeepWalk features [54] extracted from co-review graphs. Given a subject s and its reviewer set $\mathcal{U}_s \subset \mathcal{U}$ (i.e., accounts who reviewed it), we define its **co-review graph** to be a weighted graph $G_s = (V_s, E_s)$, where $V_s = \mathcal{U}_s$ and $(u_i, u_j) \in E_s$ iff users u_i, u_j have reviewed the same $w(u_i, u_j)$ subjects other than s itself. Further, given a set of co-review graphs $\mathcal{G} = \{G_1, \dots, G_k\}, G_i = (V_i, E_i)$, we define their **union fraud graph** to be the union of all the individual co-review graphs, viz., $V = \cup V_i$ and $E = \cup E_i$ for $1 \leq i \leq m$.

DeepCluster, see Algorithm 1, clusters co-review graph nodes (user accounts) based on their DeepWalk features [54], that go beyond their 1-hop neighbors and are based on random walks in the union fraud graph. DeepCluster precomputes the DeepWalk features of each account in the union fraud graph (line 1). We discuss the choice of DeepWalk parameters in § 11. For each subject s_i, $i \in [k]$, DeepCluster extracts all its users' features (line 3), and uses any fraud account detection algorithm, e.g. [12, 57] to filter out the subject's honest reviewers and their accounts (line 4). DeepCluster then uses a clustering algorithm (e.g., K-means) to group the fraudulent candidate accounts of subject s_i, $i \in [k]$ (line 5).

5.2 Features

DeepCluster returns k cluster sets, one set for each of the k subjects s_i (line 7). We use these clusters to extract *cowPred*'s first feature, **Co-cluster weight**: The number of times that u_i and u_j have appeared in the same cluster identified by DeepCluster. We further introduce several other features:

- **Co-review weight**. The co-review weight of two accounts is computed over their commonly reviewed subjects. Specifically, if $\mathcal{S}_k$ is the set of subjects reviewed by u_k, we define the co-review weight of u_i and u_j as $|\mathcal{S}_i \cap \mathcal{S}_j|$.
- **Inter-review times**. We define the *date difference* attribute for a subject $s_k \in \mathcal{S}_i \cap \mathcal{S}_j, i \neq j$ as $\Delta_{Tij}(s_k) = |dt(u_i, s_k) - dt(u_j, s_k)|$, where $dt(u, s)$ denotes the date on which user u performed an activity on subject s. Let the multiset $L_{ij} = \{\Delta_{Tij}(s_k)\}_{k=1}^{|\mathcal{S}_i \cap \mathcal{S}_j|}$. L_{ij} is a multiset, thus can contain duplicate elements. We compute the minimum, mean, median, maximum, mode, and standard deviation over L_{ij}, and obtain a vector of review-time related features in $\mathbb{R}^6$. Further, we define the *unique lockstep* feature, $u_L \in \mathbb{N}$, to be the number of unique ways (with respect to review-posting time) in which two accounts were used across subjects, i.e., the number of unique elements in the multiset L_{ij}.
- **Rating difference**. We define the *rating difference* predictor as $\Delta_{Rij}(s_k) = |R(u_i, s_k) - R(u_j, s_k)|$, where $R(u, s)$ is the rating assigned by user u to subject s. We use the multiset $L_{Rij} = \{\Delta_{Rij}(s_k)\}_{k=1}^{|\mathcal{S}_i \cap \mathcal{S}_j|}$ to derive minimum, mean, median, maximum, mode, and standard deviation for this feature over all the subjects in the intersection and obtain a vector of rating features in $\mathbb{R}^6$. Further, we also extract its number of unique elements $u_R \in \mathbb{N}$.

Intuition. Accounts with high co-review and co-cluster weights are more likely to be controlled by the same fraudster. They have not only reviewed many subjects in common, but they also have similar neighbors (as identified by DeepWalk and DeepCluster) in the individual co-review graphs of those subjects.

For the inter-review features, the statistics computed over L_{ij} leverage the observation that fraudsters synchronize the activities of the accounts that they control, e.g., in a "lockstep" behavior [18, 67, 71]. Since fraudsters need to meet tight deadlines [64], we expect $\Delta_{Tij}(s_k)$ to be lower for user accounts controlled by the same worker (fake review "burstiness" assumption [17, 27, 31, 32, 44, 47]). Further, we expect the unique lockstep u_L to be lower for pair of accounts governed by the same fraudster.

For the rating difference features, we expect u_R to be lower for pair of accounts controlled by the same worker, which would imply that both accounts tend to post the same rating for their common subjects. In Section 11.4 we use regularized logistic regression to provide further insights into the impact of these features.

We train the co-ownership predictor on the 16 features above. In Section 6 we use *cowPred* to devise a fraud de-anonymization algorithm, while in Section 7 we use it to propose a pseudonymous fraudster discovery algorithm.

6 DDA: DISCRIMINATIVE DE-ANONYMIZATION

We introduce a discriminative de-anonymization solution (DDA), a classifier that approximates the function $FDA\colon U^* \setminus \cup_{l=1}^{f} U_l \mapsto W^*$ defined in Section 3. We exploit the intuition that in DeepCluster, accounts in a union fraud graph that are controlled by the same fraudster, form a densely connected subgraph, or cluster. Knowledge that some accounts in such a cluster are controlled by a fraud worker, would allow one to attribute the other accounts in that cluster, to the same worker. However, our experiments revealed that clusters often contain accounts controlled by different fraudsters, as fraudsters tend to collaborate in search rank fraud jobs.

To disambiguate this fraud attribution problem, we leverage the co-ownership predictor, of Section 5. Specifically, DDA analyzes the clusters returned by DeepCluster (see Section 5.1). Some of the clusters may consist of both un-attributed accounts and user accounts known to be controlled by a fraud worker profile in W^*. DDA separately processes each un-attributed account u in such clusters. First, it creates links (u, u_w), for each account u_w controlled by a worker w in u's cluster. Then, it uses $cowPred(u, u_w)$

Algorithm 2: Detego system iteratively attributes new fraud to known fraudsters and discovers new fraudsters.

```
  Input  : $W^*[\,][\,]$; # seed worker profiles
  Output : $W^*[\,][\,]$; # extended worker profiles
1 $S = W^*.getProducts()$; $f = W^*.size()$;
2 while $(S.notEmpty())$ do
3     $U = S.getReviewerAccounts()$;
4     $< W^*[1..f], U_N >= FDA.(U, S, W^*)$;
5     $W^*[f+1, ..f+k] = PFD(U_N)$;
6     $S = W^*.getFreshProducts()$; $f = W^*.size()$;
7 end
8 return
```

to determine if u and u_w share the same owner. Note that u may appear in multiple clusters, computed by DeepCluster for multiple subjects. DDA extracts $|W^*|$ features for u: for each fraudster profile in W^*, the feature consists of the number of nodes controlled by that fraudster, to whom u has a link according to $cowPred(u, u_w)$. DDA uses these features to train a supervised learning algorithm.

7 PFD: PSEUDONYMOUS FRAUDSTER DISCOVERY

Following the fraud attribution process (e.g., UODA or DDA), we are left with suspected fraudulent user accounts that have not been attributed to any of the known fraudsters. We introduce now the pseudonymous fraudster discovery (PFD) algorithm that groups these un-attributed accounts into communities likely controlled by the same, albeit not yet discovered, fraudster.

PFD uses the co-ownership predictor of Section 5 to build a co-ownership graph $G_c = (V_c, E_c)$ over the unknown accounts. Nodes V_c are fraudster-controlled but un-attributed user accounts, while an edge in E_c exists between two nodes if the accounts are controlled by the same worker as predicted by *cowPred*. PFD then recursively applies a Karger [38], weighted min-cut inspired algorithm to partition the co-ownership graph into two subgraphs. These subgraphs are more densely connected than the original graph and connected through links of minimal total weight. We use triangle density $\rho(G) = \frac{t(V)}{\binom{|V|}{3}}$ for an un-weighted graph $G = (V, E)$, where $t(V)$ is the number of triangles formed by the edges in E.

8 PUTTING IT ALL TOGETHER

We introduce Detego, a fraud attribution and fraudster discovery system (see Algorithm 2). Detego takes as input a seed set W^* of f known fraudster profiles, which include user accounts known to be controlled by each fraudster. Detego expands this seed data, iteratively attributing more accounts to the known fraudsters, and identifying new fraudsters.

Detego identifies the subjects S reviewed by the accounts controlled by the seed fraudsters (Algorithm 2, line 1), then retrieves all the user accounts U who reviewed these subjects (line 3). The accounts in U include accounts controlled by the f fraudster profiles in W^*, as well as accounts controlled by other, not yet identified fraudsters, and also honest accounts. Detego uses a fraud de-anonymization (FDA) algorithm, e.g., either UODA or DDA to (1) attribute accounts from U to the fraudster profiles in W^* (line 4), and (2) identify the other, non-attributed accounts from U, denoted by U_N. Detego uses the PFD algorithm (line 5) to group the accounts from U_N into communities belonging to k new fraudsters. It then continues to iterate over newly discovered subjects, reviewed by these new fraudsters or by the previously known fraudsters (line 6), and over newly identified fraudsters, e.g., using the techniques described in Section 9.

Algorithm 3: Interaction protocol with human fraud workers, to provide ground truth performance evaluation for fraud de-anonymization algorithms.

```
   Input  : $P$; # User study participant;
            $m$, $n$, $q$ ; # Numbers of accounts
   Output : $A[]$; # Accounts attributed to $P$;
 1 $A = P$.revealAccounts($m$);
 2 $Data[]$ = BFS($A$, 2);
 3 $newAccounts[n]$ = FDA($A$, $Data$);
 4 $ACAccounts[q]$ = genAttentionCheckAccounts();
 5 $Q$ = genQuestionnaire($newAccounts$, $ACAccounts$);
 6 $Answers$ = send($A$.randomAccount(), Q);
 7 if Answers.passAttentionCheck() then
 8     if newAccounts.getConfirmed().verifyOwnership() then
 9         $A$.add(newAccounts.getConfirmed());
10     end
11 end
12 return $A$
```

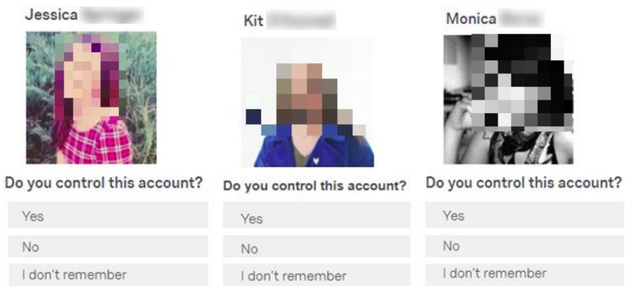
9 FRAUD DE-ANONYMIZATION ORACLES

We leverage the observation that fraud workers know the user accounts that they control, to introduce a novel approach to validate fraud de-anonymization solutions, that converts human fraud workers into FDA oracles. In Section 10 we use this approach to evaluate UODA.

Algorithm 3 outlines our validation protocol, where m, n, q are integer parameters. The protocol consists of 2 main interaction steps. In the first step, we ask each participant, i.e., recruited human fraud worker, to reveal m user accounts that they control in Google Play, by sending their Google e-mail addresses associated with these accounts (Algorithm 3, line 1). We then use a depth-2 breath first search approach to collect (1) all the apps reviewed by the m accounts and (2) all the reviewers of these apps (line 2). We apply a fraud de-anonymization solution (see next section) to identify n new, *candidate accounts*, i.e., other Google Play accounts suspected to be controlled by the same participant (line 3).

For the second interaction step, we have designed a questionnaire that asks the participant to confirm if they control each of these n candidate accounts, see Figure 3. Specifically, for each account, we show the account's profile photo and name, and ask the participant if they control the account. We provide 3 options, "Yes", "No" and "I don't remember".

Participant validation. We have developed the following tests to validate participant attention and honesty:


Jessica [redacted]

Do you control this account?

Yes

No

I don't remember

Kit [redacted]

Do you control this account?

Yes

No

I don't remember

Monica [redacted]

Do you control this account?

Yes

No

I don't remember

Figure 3: Anonymized screenshots of 3 questionnaire pages, for accounts (left) revealed in step 1 to be controlled by the participant, (center) known not to be controlled, and (right) suspected by UODA to be controlled by the participant.

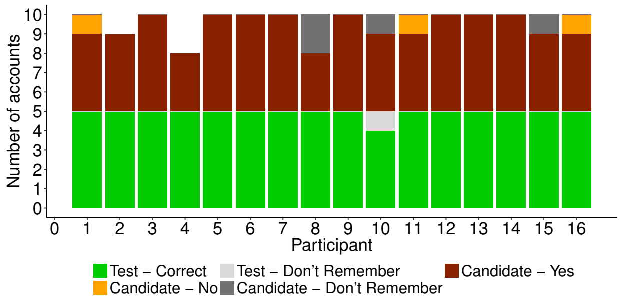


Figure 4: Results of UODA on data validated by 16 human fraud worker participants. UODA achieves an overall precision of 91%.

- **Attention check**. In addition to the n candidate accounts, we add to the questionnaire q other *test* accounts (line 4), for which we know the answer: (1) accounts that we know that the participant controls, i.e., picked randomly from among the m accounts revealed in the first step, and (2) accounts that we know that the participant does not control, i.e., accounts that have at least 20 followers and significant other activities in Google Plus (posting photos, videos). We present the questions for the $n + q$ candidate and test accounts, in randomized order (line 5).
- **E-mail knowledge**. Each Google Play account A has an associated e-mail address E. Given E, one can easily retrieve the account A. However, E is not public, and, given only knowledge of A, one cannot find E. We leverage this observation to ask each participant to reveal the e-mail address E of each Google Play account A that they claim to control. We use E to find the corresponding account A'. The participant fails this test if A' does not exist or $A' \neq A$.
- **E-mail based validation**. To verify ownership of claimed accounts, we send the questionnaire to one of the m e-mail addresses revealed in the first step (randomly chosen) (line 6).
- **Token and e-mail based validation**. To verify ownership of accounts confirmed in the questionnaire (line 8), we choose randomly one of the n accounts confirmed, and send to its corresponding e-mail address, a random, 6 character token. The accounts verify iff. the participant can reproduce the token.

10 USER STUDY

We have recruited 16 fraud workers from India (4), Bangladesh (4), UK (2), Egypt (2), USA (1), Pakistan (1), Indonesia (1), and Morocco (1), 12 male and 4 female, who claimed to control between 40 to 500 accounts (M=211, SD=166). We have used these participants to evaluate the performance of UODA. We have set m=10, n=5 and q=5, thus each participant reveals 10 accounts controlled in Google Play, then further confirms or denies control of 5 other UODA detected accounts, and 5 test accounts. To run UODA, we have used the 10 accounts revealed by each participant in the first step, to collect (via BFS) 718 apps, 265,724 reviewers and 341,993 reviews in total. We collected up to 175 apps, 37,056 reviews and 22,848 reviewers from a single worker. The participation incentive was set to $10 for each participant.

Ethical considerations. We have developed IRB-approved protocols to ethically interact with participants and collect data. We have not asked the participants to post any fraud on the online service. We restricted the volatile handling of emails and photos of accounts revealed by participants, to the validation process. We have immediately discarded them after validation. We believe that this information cannot be used to personally identify fraudsters: recruited fraudsters control between 40 and 500 accounts each (M=211, SD=166) thus any such account is unlikely to contain PII. Further, since we do not preserve these emails and photos, their handling does not fall within the PII definition of NIST SP 800-122. Under GDPR, the use of emails and photos without context, e.g., name or personal identification number, is not considered to be "personal information".

In the following we first detail the instantiation of UODA that we evaluated, then describe the results of the user study.

10.1 UODA Parameters

We evaluate UODA (see § 4) using two features, defined by the sets (1) $C_{l\geq} = \{(s, s') \in S_l \mid cr(s, s') \geq b_1\}$, where $cr(s, s')$ is the number of reviewers shared by subjects s and s' and (2) $U_{l\geq} = \{s \in S_l \mid u_l(s) \geq b_2\}$,where $u_l(s)$ is the number of accounts controlled by worker W_l who has reviewed subject s. Specifically, these features define the family of sets $\mathcal{F}_{W_l}$ with m=4:

$$\begin{aligned}
\Omega_{l1} &= \{s \in S_l \mid s \in C_{l\geq} \setminus U_{l\geq}\} \\
\Omega_{l2} &= \{s \in S_l \mid s \in U_{l\geq} \setminus C_{l\geq}\} \\
\Omega_{l3} &= \{s \in S_l \mid s \in C_{l\geq} \cap U_{l\geq}\} \\
\Omega_{l4} &= \{s \in S_l \mid s \in (C_{l\geq} \cup U_{l\geq})^C\}
\end{aligned} \qquad (7)$$

The rationale behind this selection of Ω sets is that fraudsters are hired to provide large number of reviews for different subjects. Thus, a fraudulent account u controlled by a fraudster profile $(W, U, S) \in W^*$ is more likely to post reviews for subjects that were reviewed by other accounts under its control, see e.g. [35, 48, 70, 87].

10.2 Results

Figure 4 shows that 15 of the 16 participants have provided correct responses to all 5 test accounts. The remaining participant answered "I don't remember" for a single test account, known not to be controlled by the participant. We have thus decided to keep the data from all participants. Further, for participants 2 and 4, UODA found less than 5 suspected accounts (i.e., 4 and 3 respectively).

We observe that 10 out of 16 participants have confirmed control (and passed our verification) of all UODA proposed accounts. 5 participants confirmed control of 4 out of 5 UODA recommended accounts and 1 participant confirmed control of only 3 accounts out of 5 UODA recommended accounts. UODA's precision ($\frac{TP}{TP+FP}$, where TP is the number of true positives and FP is the number of false positives) is thus 91%, i.e., 7 unconfirmed accounts among 77 predicted. We note that for 3 out of the 7 unconfirmed accounts, the participants did not remember if they control them or not.

11 EMPIRICAL EVALUATION

11.1 Attributed Account Data

We have recruited an additional set of 23 fraud workers and performed only the first step of the fraud de-anonymization validation protocol of § 9, where we asked each participant to reveal at least 15 accounts that they control in Google Play. Figure 5 shows the number of accounts (bottom, red segments) revealed by each of the 23 workers, between 22 and 86 accounts revealed per worker, for a total of 942 attributed fraud accounts.

We have selected the top 640 *fraud apps*, that received the highest percentage of reviews from accounts controlled by the 23 fraudsters, and crawled their reviews once every 2 days, over a 6 month period. The 640 apps had between 7 to 3,889 reviews. Half of these apps had at least 51% of their reviews written from accounts controlled by the 23 fraudsters. On the whole, the 640 apps have received 159,469 reviews, of which 17,575 were written from the above 942 attributed fraud accounts.

In the following, we use this data to evaluate the ability of developed solutions to (1) attribute *unknown* accounts to existing seed workers and (2) reveal hidden relationships among reviewers towards uncovering previously unknown fraudulent workers.

11.2 DeepCluster Parameter Tuning

We have built the union fraud graph over the user accounts who reviewed the 640 fraud apps. To run DeepWalk, we transform this union fraud graph into a non-weighted graph, where we replace an edge between nodes u_i and u_j with weight $w_{ij} = w(u_i, u_j)$, by w_{ij} non-weighted edges between u_i and u_j. This ensures that the probability of DeepWalk choosing node u_j as next hop while at node u_i is proportional to w_{ij}. The resulting union fraud graph has 56,950 nodes and 34,742,730 edges (5,858,940 unique edges) and consists of 202 disconnected components.

Algorithm 4 shows the pseudocode for the grid search process that we used to identify the best performing DeepWalk parameters on the union fraud graph: $d = 300, t = 100, \gamma = 80, w = 5$. d is the number of dimensions when representing nodes in the graph, t is the maximum length of a random walk, γ is the number of random walks started from each node, and w is the the number of neighbors used as the context in each iteration of its SkipGram component.

We have used K-means as clustering algorithm in DeepCluster (see § 1) considering that we have prior knowledge about the number of workers who targeted each subject. We identified the optimum K value required by K-means for each subject s_i experimentally, as follows. Iterate for values of K ranging from 2 to $|W_i|$ where $|W_i|$ is the number of distinct workers known to have targeted subject s_i. Since K-means is susceptible to local optima, we run it 100 times on the embeddings of the co-review graph of subject s_i, and assess the quality of the returned clusters. We use a quasi-F1 score that gages how good a cluster configuration is with regards to our ground truth. We also adjust for the number of accounts in each cluster and compute the weighted average across all clusters in one cluster configuration.

Algorithm 4: DeepWalk parameter tuning. For each parameter set, compute Deepwalk embeddings on the union fraud graph and run stratified cross validation (SCV) using a learning algorithm *Alg* and only seed accounts as part of the training and validation set (lines 3-5). We save the best performing configuration (lines 6-8).

```
Input  : $CRG$ # Co-review Graph
         $S$ # seed accounts
         $Alg$ # learning algorithm
Output : $DWParams$ # Best DeepWalk parameters
1  $F_{max} = 0, DWParams = \emptyset$
2  $ParamSet$ = Generate.Grid($\{t, d, \gamma, w\}$)
3  for $p \in ParamSet$ do
4      $D = S \ltimes CRG.DWFeatures(p)$
5      $F$ = SCV($D, Alg$)
6      if $F > F_{max}$ then
7          $DWParams = p$
8      end
9      $F_{max}$ = max$\{F, F_{max}\}$
10 end
11 return $DWParams$
```

Table 1: Performance of UODA and DDA on ground truth data set. DDA performs better. However, with only 2 features, UODA reaches an F1 of 83%.

Approach	Algorithm	Precision	Recall	F1
UODA	Top 1	85.11%	82.59%	83.83%
	Top 2	92.05%	90.32%	91.11%
	Top 3	94.23%	92.91%	93.57%
DDA	KNN	94.28%	93.35%	93.81%
	MLP	94.90%	94.10%	94.50%
	RF	94.37%	93.31%	93.84%

11.3 Fraud De-Anonymization

We compare the ability of the UODA and DDA algorithms to de-anonymize the ground truth attributed account dataset of § 11.1. For this, we first set randomly aside 75% of the seed accounts from each worker into a set G_T (Ground Truth) and let the remaining 25% accounts be the T_T (Testing Truth) set. For DDA, we train the co-ownership predictor using accounts in G_T, then apply the predictor to all accounts in T_T and extract as features the number of nodes in each class (known fraudster) to whom the account has a link according to the co-ownership predictor. Finally, we train a classifier on these features using stratified 10-fold cross validation.

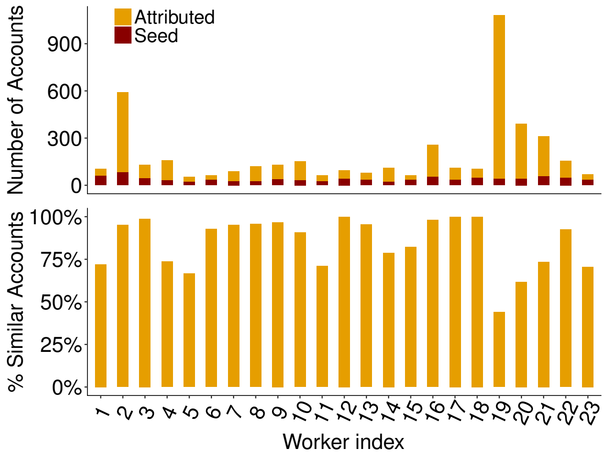


Figure 5: (Top) Distribution of seed and DDA attributed accounts across the 23 fraudulent workers. DDA attributed 3,547 accounts to these fraudsters, 3.7 times more than the size of the seed set. (Bottom) Per worker percentage of newly attributed accounts suspected of self-plagiarism. Almost all (≥ 90%) of the newly attributed accounts for 13 out of 23 fraud workers have self-plagiarized reviews.

For UODA, following the G_T/T_T split, we compute the Ω sets as described in (7) using accounts in G_T and test the algorithm on the review histories of all accounts in T_T. We fix the same $b_1 = 10$ and $b_2 = 15$ (obtained through a grid search) across all the workers. Then, given an account u in T_T, we select as candidate the worker whose partition maximizes the function in Equation (4), i.e., we evaluate such function 23 times (one for each worker) and attribute u to the worker that maximizes it. Note that to evaluate the function, we need P_i: the popularity volume of all the subjects in each Ω_i. We approximate $P_i = \epsilon \sum_{s_j \in \Omega_i} R(s_j)$ where $R(s_j)$ is the number of reviews that subject s_j received from fraudster accounts in the G_T set and ϵ was set to mimic a probability distribution on $\mathcal{S}$. In practice, we have evaluated multiple values for ϵ, and chose $\epsilon = 10^{-6}$ as best performer.

Table 1 compares UODA and DDA results after 10 different random G_T/T_T splits. We observe that DDA achieves an F1 measure of 94.5%, outperforming UODA's top 1 choice. UODA's performance, however, significantly increases when allowed to make mistakes. Specifically, Top 2 UODA achieves an average F1 of 91.11% while Top 3 UODA achieves an average F1 of 93.57%.

Fraud Attribution in the Wild. We have further trained DDA on all the ground truth information (both G_T and T_T sets). We then applied the trained DDA to 3,681 accounts that appeared in at least one seed cluster but never appeared in an unknown cluster of the 640 suspicious apps (§ 11.1). Figure 5 (top) shows the distribution of 3,547 of these accounts attributed to the 23 fraud workers. Only 134 accounts were not assigned to any fraud worker. To validate this result, we computed the review's Jaccard similarity between each newly attributed $\hat{U}_l$ account and all seed U_l accounts, using the review's $k-$shingle representation as defined in [19].

Figure 5 (bottom) shows the proportion of newly assigned accounts $u \in \hat{U}_l$ that have at least one review similar ($J(R_{\hat{u}}, R_u) \geq 0.5$) to those of accounts in its respective seed set. We have set $k = 3$ and considered only reviews with at least 10 characters in length. We observe that 13 out of 23 fraudulent workers have around 90% of their new attributed accounts with similar reviews to the ones written by its seed accounts. Likewise, 22 out of 23 fraudsters have at least 50% of their accounts with similar reviews. These results confirm DDA's outcome and previous work on crowdsourced review manipulation, e.g., [36].

Table 2: Performance of our co-ownership predictor *cowPred* vs. ELSIEDET [87] on ground truth data. *cowPred* significantly outperforms ELSIEDET.

Solution	ML Algo.	Precision	Recall	F1
cowPred	**GBM**	**96.40%**	**96.94%**	**96.67%**
	RF	96.30%	97.01%	96.65%
	SVM	93.75%	95.34%	94.54%
	RLR	93.72%	94.42%	94.07%
	NB	88.44%	95.66%	91.91%
ELSIEDET	Grid search	82.41%	85.92%	84.13%

11.4 Co-Ownership Predictor

We evaluate the performance of the co-ownership predictor *cowPred* of Section 5, and compare it against ELSIEDET's state-of-the-art solution [87]. For this, we build training data as follows. First, create complete graphs from among seed attributed accounts found in clusters across all the product space, i.e., create a link (u, v) for $u, v \in C_j$ where C_j is a cluster in product j. Then, using the 942 accounts of § 11.1, generate "positive" links (class 1) when both accounts in the link are known to be controlled by the same fraudster and "negative" links (class 0) when controlled by different fraudsters. Finally, for each link (u, v), extract the 16 features described in Section 6 and append its class. Our training set consists of 17,695 pairs of user accounts, 79.5% of which are controlled by the same fraudster.

We use this data to train several supervised learning algorithms and select the top performer as the co-ownership predictor. Specifically, we used several sampling strategies and supervised learning algorithms that train on the features of the co-ownership predictor: Gradient Boosting Machine (GBM), Random Forests (RF), Support Vector Machine (SVM), Regularized Logistic Regression (RLR), and Naive Bayes (NB). We also set aside 20% of the 17,695 links as a test set to assess the quality of the co-ownership predictor after training with 10-fold CV. Further, to evaluate the impact of class imbalance, we compared the *no sampling* strategy against strategies of *undersampling* and *oversampling*. For the *undersampling* strategy, we created a 50-50 training set with 2,901 links for each class. For the *oversampling* strategy, we used the SMOTE algorithm [21] and created synthetic data along the line segments joining any or all of the k minority class nearest neighbors. *cowPred*'s results were very similar for the no sampling and oversampling strategies, outperforming the undersampling strategy. Thus, in the following we present results only for the no sampling strategy.

The ELSIEDET co-ownership predictor. We compare *cowPred* against the state-of-the-art ELSIEDET's Sybil social link builder [87]. ELSIEDET builds social links between Sybil user accounts based on their similarity: (i) whether their reviews were posted for the same app, (ii) within a fixed time window ΔT, and (iii) were either 1-star

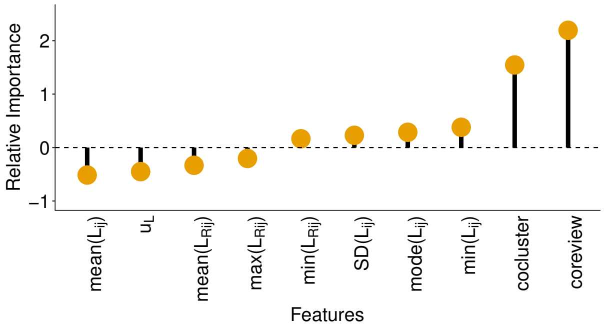


Figure 6: Relative importance (shown as $sign(y) * log(1 + abs(y))$) for statistically significant features in the co-ownership predictor using logistic regression. *Co-review* **and** *co-cluster* **have the highest positive impact, while the mean date difference on L_{ij} and the unique lockstep u_{ij} have the largest negative weight.**

or 5-star. Accounts u and v are considered to form a Sybil social link iff $sim(u, v) \geq \beta$, where β and ΔT are parameters. Zheng et al. [87] manually tuned these parameters, as they observed that several supervised learning techniques were not sensitive to different thresholds employed. We have improved on this manual tuning process, by implementing a grid search to obtain the best parameters $\Delta T^*, \beta^*$, using the same training set used for our *cowPred* predictor. We compute performance for ELSIEDET based on whether links (u, v) were predicted to be controlled by the same worker.

Comparison results. Table 2 compares *cowPred*'s performance on the test set, for the best performing supervised learning algorithms evaluated, against ELSIEDET's Sybil social link builder, with best parameters $\Delta T^* = 30$ and $\beta^* = 0.01$. For *cowPred*, GBM and RF achieved the best overall results. *cowPred* significantly outperformed ELSIEDET, with an F1-measure of 96.67% vs. 84.13%. While ELSIEDET was designed for a different type of social network (i.e., Dianping, Yelp), and a different adversary type (elite reviewer), we believe that *cowPred*'s advantage stems from its use of features extracted from common review behaviors exhibited by Sybil accounts. We note that we were not able to compare *cowPred* against other related solutions, e.g., Kumar et al.'s sockpuppet pair detection approach [40], as they leverage features not available in Google Play, such as community features (whether account is blocked, fraction of reported and deleted posts).

Feature Insights via Regularized Logistic Regression. In order to understand the impact of and confirm the intuition behind the *cowPred* features (see § 5.2), we train *cowPred* on the entire data set (17,695 links) using a regularized logistic regression model [29]. Figure 6 shows the relative importance of the statistically significant variables after applying Wald Chi-Squared test. We measure importance as the value of the coefficients corresponding to the trained model.

We observe that the co-review and co-cluster features have a strong positive effect on the probability of two accounts being controlled by the same worker. The higher their values the more likely it is that two accounts are owned by the same underlying worker. Similarly, a positive weight for $mode(L_{ij})$ and $min(L_{ij})$ (see § 5.2) suggests that if a long period of time between reviews is repeated across most of the commonly reviewed apps then it is more likely that the two accounts are handled by the same worker. However, the unique lockstep feature u_L shows a negative effect, i.e., the larger its value, the less likely it is that both accounts belong to the same worker. Equivalently, contrary to the burstiness assumption, the time difference for all reviews in common are rarely similar. The sign effects of $mean(L_{ij})$ and $SD(L_{ij})$ are less intuitive. We conjecture these sign effects are the result of existing correlation across all variables. Further, $mean(L_{R_{ij}})$ impacts negatively the probability of co-ownership. Hence, accounts controlled by the same worker tend to award similar star rating to their commonly reviewed apps. However, we notice that rating features have the least significant effect. This observation implies that most workers post either positive or negative reviews.

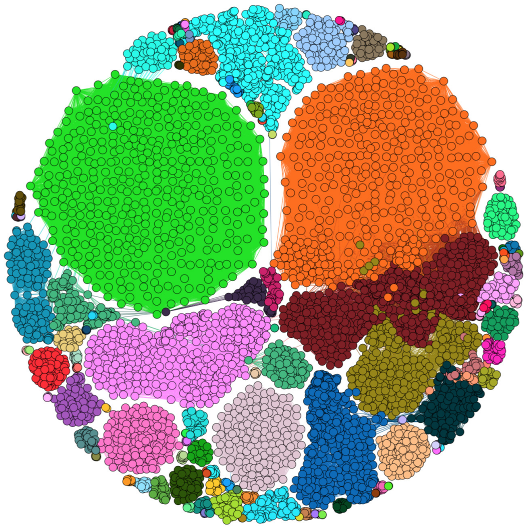


Figure 7: Co-ownership (co-w) graph over 5,548 user accounts who reviewed 640 apps involved in fraud. Two accounts are connected if they were predicted to be controlled by the same fraudster. Partition algorithm identified 129 user account components, each potentially controlled by a different fraudster. The largest cluster has 962 nodes and 54 components have more than 10 nodes.

11.5 Pseudonymous Fraudster Discovery

We applied the *cowPred* predictor with no sampling strategy and GBM with Bernoulli loss function. We used 279,431 links from 5,690 unknown (un-attributed) user accounts that reviewed 640 suspicious apps. These accounts occurred in clusters without seed accounts (unknown clusters). The resulting co-ownership graph consists of 5,548 user accounts and 97,448 edges. Figure 7 shows 129 components identified by PFD. We conjecture that each of these dense components is controlled by a different fraudster. In the following, we validate this conjecture.

Result Validation. We use orthogonal evidence of fraud to validate the dense components of Figure 7. Specifically, we inspect reviews' text written by accounts in each cluster. Upon manual investigation, we found many suspicious behaviors, including **singular coincidence**: The review *"this game is Really cute and awesome. I think*

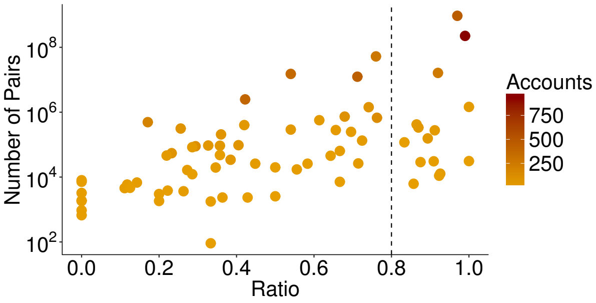


Figure 8: Scatterplot for 71 fraudster communities (shown as dots) discovered by PFD: the percentage of users who wrote reviews that are at least 50% Jaccard similar to other reviews (x axis) vs. the number of review pairs (in log scale) in each component (y axis). 15 communities have at least 80% of their user accounts suspected of plagiarism.

this is so addicting cause when my kid play this game; i can't resist her to playing it." was posted from three different accounts in the same component for three different apps on the same date; **the enthusiastic reviewer**: A user account posted the review: *"Try it guys for who never use this app.. I'm enjoy and love app...thanks very much.. because i really enjoy with this app..."* for 40 apps in two days; and **the lazy high-level editors**: We found 12 accounts in one component that used the review *"[App Name] It is very exciting. I like it Nice app! Beautiful screenshot. Very interesting It is useful. I like it so much"* as a template to post reviews for 8 apps. The fraudster would tailor this template by adding the name of the app as a prefix.

In addition, similar to the validation in § 11.3, we have computed the Jaccard similarity for every pair of reviews using their text's k-shingle representation with $k = 3$. We performed this calculation over each of the 71 detected components with at least 6 accounts. This experiment generated a total of 1.1 billion Jaccard pairs from 118,281 reviews belonging to 5,364 accounts. Moreover, we evaluate the possibility that accounts responsible for reviews with low similarity are generated by accounts not engaged in review manipulation. Specifically, we first computed, a, the number of user accounts in a component that posted reviews with Jaccard similarity at least 0.5 to other reviews in that component. Next, we computed, b, the total number of accounts for each of the selected components. Finally, we computed the ratio a/b. Figure 8 highlights fifteen components (1967 users) with ratio greater than 0.8. Very few components have a ratio below 0.3. This result suggests that, even for large components, users that generated very dissimilar reviews are in fact also engaged in review manipulation that reuse high amounts of text.

12 DISCUSSION AND LIMITATIONS

Underground fraud markets. If successful, the fraud de-anonymization approach proposed in this paper may drive fraudsters to underground markets. This is however compatible with our objectives, to degrade fraudster capabilities and real-life impact. Further, we observe that DETEGO's ground truth collection and solution validation approach, identifies and leverages intrinsic vulnerabilities in the developer-to-fraudster interactions, i.e., developers need to verify claimed fraudster expertise and fraudsters need to make a profit. Even underground markets need to provide basic functionality that includes worker expertise, developer reputation verifications, and payment mechanisms. When underground fraud markets become accessible to regular developers, they will also be accessible to researchers, who can exploit the same vulnerabilities for ground truth collection and fraud de-anonymization validation purposes.

Evasion strategies. Fraudsters can try to game the DETEGO system. For instance, a fraudster can use multiple sets of disjoint accounts and never use them while reviewing the same app. We observe however that DETEGO introduces a tradeoff between the fraud operation's efficiency and its detectability. Decreasing account reuse decreases profits, as reputable accounts are often preferred in search rank fraud jobs [22, 68, 87]. Increasing account reuse exposes the fraud operation to DETEGO detection and attribution. Thus, DETEGO forces fraudsters to minimize account reuse and reduces review fraud incentives.

Further, an adversarial developer who wants to boost the average rating of her app, needs to commission a number of fake reviews that is linear in the number of the app's honest reviews [55]. Such behavior however affects the temporal distribution of the app's reviews [55], which makes it detectable, i.e., through the inter-review-time and rating-difference features of DETEGO.

Importance of seed fraud data. DETEGO can effectively provide fraud de-anonymization only in the presence of seed ground truth information about accounts controlled by known fraudsters. Future work may explore the ability of cross-site identity linking attacks [15, 16, 34, 62, 86] (see § 13) to e.g., link reviews of detected Sybil communities to public profiles of crowdsourcing accounts.

Informed consent. To recruit 16 participants for the user study of Section 10, we have contacted 320 fraud workers. This small turnout may be due to a combination of factors, that include deserted accounts, lack of interest, and the online consent form used as part of our IRB approved validation process. We note that the 16 participants were honest (a single "I don't remember" among 80 test accounts). Future work may investigate the use of IRB approved deception to evaluate the impact of the consent form on the number of participants, their honesty, and the precision of fraud de-anonymization algorithms.

We believe that realization of consequences will not be a major factor in the recruitment process. Our results suggest that reward driven participation is enough for certain fraudsters. Proofs of expertise are normal in crowdsourcing sites, where they enable developers gain confidence when hiring workers. Thus, DETEGO's data collection (or variations) can blend in with regular recruitment of fraud. Further, the use of deception may increase the probability of successful recruiting.

Fraud account memorability. Search rank fraud workers can control hundreds of accounts in the online system, which can impact memorability. However, in our study, participants were able to correctly detect ground truth controlled and non-controlled accounts. The caveat is that we only presented participants with 5 test accounts. Future work should determine the maximum number of questions that we can ask participants, before factors like fatigue and boredom impact their honesty and accuracy.

I.i.d. assumption. UODA assumes that the review history of a fraudulent user account is independent and identically distributed, i.e., that an element in the sequence of reviews is independent of the element that came before it. A possible future work is to explore UODA assuming a Markovian review-posting model.

13 RELATED WORK

Author identification and cross-site identity linking. The *author identification* problem seeks to identify the original author of a document [51]. Narayanan et al. [51] used linguistic stylometry to perform large scale identification of blog post authors and argue damaging implications to anonymous bloggers and whistleblowers. Another closely related problem is that of *cross-site identity linking* attacks [15, 16, 34, 62, 86]. Adversaries were shown to be able to exploit linguistic [14] and location [30] patterns to link pseudonymous identities of the same user across different sites. Backes et al. [16] introduced relative and absolute linkability measures that rank identities by their anonymity, and used information about matching identities to estimate linkability risks. Andreou et al. [15] further studied relationships between anonymity and risks of linkability of Facebook and Twitter accounts.

Venkatadri et al. [73] leveraged this attack to develop a framework to transfer trust between sites and identify trustworthy accounts. Jain et al. [34] observed that Facebook and Twitter profiles share attributes, to develop identity search methods that link Twitter accounts to their owners' Facebook accounts. Cloning attacks [39], where adversaries clone the accounts of victims from one site to another, may thwart this linkage.

In the context of our work, de-anonymization is not an attack but a desirable feature. This problem is also more challenging: unlike Twitter and Facebook, crowdsourcing and peer-opinion sites do not facilitate explicit forms of inter-connection. Further, instead of finding a one-to-one mapping, our research focuses on a many-to-one de-anonymization strategy that seeks to attribute many fake identities to a real identity (i.e. underlying fraud worker).

Sybil community detection. The pseudonymous fraudster discovery problem is equivalent to uncovering Sybil (or sockpuppet) communities. Sybil accounts disconnect physical from online identities, thus have a suite of malicious uses, that include gaining control over systems [25], vandalism [63], or creating the illusion of widespread support of ideas, people and products [66]. Early Sybil detection work in online systems has focused on social networks [23, 72, 84, 85], and made the assumption that attackers can easily form social relationships between Sybil accounts they control, but find it hard to establish links to honest accounts. However, Yang et al. [81] showed that in Renren, Sybil accounts do not form tight-knit communities, and are well connected with honest users.

In peer-opinion systems that lack strong social links between user accounts, social graphs can be replaced by *co-activity graphs*, such as our co-review graphs. Then, in discussion communities, Kumar et al. [40] showed that Sybil accounts still differ from honest accounts through social network structure, posting behavior and linguistic traits. They leveraged the discovery that pairs of accounts controlled by the same individual are more likely to interact on the same discussion, to build a co-ownership predictor. Zheng et al. [87] predict Sybil links between user accounts based on the similarity of their reviews, in terms of the products targeted, times and ratings.

In Section 11.4 we show that our co-ownership predictor significantly outperforms the accuracy of Zheng et al. [87]'s predictor. We did not compare against the predictor of Kumar et al. [40], that uses community feedback features that are unavailable in sites like Google Play. Further, after detecting Sybil communities, DETEGO seeks to de-anonymize them by finding the crowdsourcing account of the human fraud worker who controls them.

Fraud detection. There is a large body of research on defending against online system fraud. State of the art approaches use inference on the social graph [12, 37, 53, 58, 75] and classical machine learning based on several assumptions. These assumptions include: (i) bursty activity [27, 43, 44, 83], (ii) review plagiarism [31, 36, 37, 47] and distinguishability of machine vs. human generated reviews [82], (iii) extreme reviews and deviation [47, 58, 77, 79], (iv) lockstep behavior [18, 67, 71], and (v) ratio of singleton accounts [58, 61, 83]. Unlike this work, that has focused on providing binary classification of reviews as fake or honest, and accounts as fraudulent or benign, we seek to identify the prolific workers responsible for significant fraud. We implement a maximum likelihood estimation and deep learning based guilt-by-association process to expand seed, fraudster-controlled account sets, and assign them to the crowdsourcing worker who controls them.

Fraud data collection. De Cristofaro et al. [24] deployed Facebook honeypot pages and analyzed *like farms* based on demographic, temporal and social dimensions. Some farms seemed to be operated by bots while others mimic regular users' behaviors. Stringhini et al. [68] studied Twitter follower markets by purchasing followers from different merchants and used such ground truth to discover patterns and detect "market" accounts in the wild. In this paper we use fraudster responses to conduct a live validation of our solutions, and map accounts in the online peer-opinion system to the controlling crowdsourcing worker.

14 CONCLUSIONS

In this paper we study the search rank fraud de-anonymization problem and show that it is different from the well studied fraud or spammer detection problem. We model fraud de-anonymization as a maximum likelihood estimation problem and develop an unconstrained optimization fraud de-anonymization algorithm. We introduce a graph based deep learning approach to predict co-ownership of fraudulent account pairs, and use it to build discriminative fraud de-anonymization and pseudonymous fraudster discovery algorithms. Further, we introduce the first protocol to involve human fraud workers in the task of evaluating the performance of fraud de-anonymization algorithms. We show that our solutions achieve high precision and recall on ground truth data, significantly outperform a state-of-the-art approach and are able to attribute thousands of new accounts to known crowdsourced fraudsters.

ACKNOWLEDGMENTS

We thank the anonymous reviewers for their insightful feedback. This research was supported by NSF grant CNS-1527153 and by the Florida Center for Cybersecurity.

REFERENCES

[1] [n. d.]. Freelancer. http://www.freelancer.com.

[2] [n. d.]. The FTC's Endorsement Guides: What People Are Asking. https://tinyurl.com/p7hk9uz.

[3] [n. d.]. Upwork Inc. https://www.upwork.com.

[4] 2012. Yelp tries public shaming to discourage businesses from gaming reviews and ratings. Digital Trends, https://www.digitaltrends.com/social-media/yelp-cracking-down-on-fake-reviews/.

[5] 2013. Google I/O 2013 - Getting Discovered on Google Play. www.youtube.com/watch?v=5Od2SuL2igA.

[6] 2013. Why does Yelp hide reviews? Washington Post https://www.youtube.com/watch?v=s1lJuu44cJA.

[7] Last accessed November 2016. App Reviews. http://www.app-reviews.org.

[8] Last accessed November 2016. App Such. http://www.appsuch.com.

[9] Last accessed November 2016. Apps Viral. http://www.appsviral.com/.

[10] Last accessed November 2016. Rank Likes. http://www.ranklikes.com/.

[11] Last accessed November 2016. The Social Marketeers. http://www.thesocialmarketeers.org/.

[12] Leman Akoglu, Rishi Chandy, and Christos Faloutsos. 2013. Opinion Fraud Detection in Online Reviews by Network Effects. In *Proceedings of AAAI ICWSM.*

[13] Muhammad AL-Qurishi, Mabrook Alrakhami, Atif Alamri, Majed Alrubaian, Sk Md Mizanur Rahman, and M Hossain. 2017. Sybil Defense Techniques in Online Social Networks: A Survey. PP (01 2017), 1–1.

[14] Mishari Almishari and Gene Tsudik. 2012. Exploring linkability of user reviews. In *European Symposium on Research in Computer Security*. Springer, 307–324.

[15] Athanasios Andreou, Oana Goga, and Patrick Loiseau. 2017. Identity vs. Attribute Disclosure Risks for Users with Multiple Social Profiles. In *Proceedings of the IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*. 163–170.

[16] Michael Backes, Pascal Berrang, Oana Goga, Krishna P Gummadi, and Praveen Manoharan. 2016. On profile linkability despite anonymity in social media systems. In *Proceedings of the 2016 ACM on Workshop on Privacy in the Electronic Society*. 25–35.

[17] Prudhvi Ratna Badri Satya, Kyumin Lee, Dongwon Lee, Thanh Tran, and Jason Jiasheng Zhang. 2016. Uncovering Fake Likers in Online Social Networks. In *Proceedings of the ACM CIKM*.

[18] Alex Beutel, Wanhong Xu, Venkatesan Guruswami, Christopher Palow, and Christos Faloutsos. 2013. CopyCatch: Stopping Group Attacks by Spotting Lockstep Behavior in Social Networks. In *Proceedings of the WWW*.

[19] A. Broder. 1997. On the Resemblance and Containment of Documents. In *Proceedings of the Compression and Complexity of Sequences 1997 (SEQUENCES '97)*. IEEE Computer Society, Washington, DC, USA, 21–. http://dl.acm.org/citation.cfm?id=829502.830043

[20] Qiang Cao, Michael Sirivianos, Xiaowei Yang, and Tiago Pregueiro. 2012. Aiding the Detection of Fake Accounts in Large Scale Social Online Services. In *Presented as part of the 9th USENIX Symposium on Networked Systems Design and Implementation (NSDI 12)*. USENIX, San Jose, CA, 197–210. https://www.usenix.org/conference/nsdi12/technical-sessions/presentation/cao

[21] Nitesh V. Chawla, Kevin W. Bowyer, Lawrence O. Hall, and W. Philip Kegelmeyer. 2002. SMOTE: Synthetic Minority Over-sampling Technique. *J. Artif. Int. Res.* 16, 1 (June 2002), 321–357. http://dl.acm.org/citation.cfm?id=1622407.1622416

[22] Nicholas Confessore, Gabriel Dance, Richard Harris, and Mark Hansen. 2018. The Follower Factory. *The New York Times* (Jan 2018). https://www.nytimes.com/interactive/2018/01/27/technology/social-media-bots.html

[23] George Danezis and Prateek Mittal. 2009. SybilInfer: Detecting Sybil Nodes using Social Networks. In *NDSS*.

[24] Emiliano De Cristofaro, Arik Friedman, Guillaume Jourjon, Mohamed Ali Kaafar, and M. Zubair Shafiq. 2014. Paying for Likes?: Understanding Facebook Like Fraud Using Honeypots. In *Proceedings of the 2014 Conference on Internet Measurement Conference (IMC '14)*. 129–136.

[25] John R. Douceur. 2002. The Sybil Attack. In *International workshop on peer-to-peer systems*. 251–260.

[26] Amir Fayazi, Kyumin Lee, James Caverlee, and Anna Squicciarini. 2015. Uncovering Crowdsourced Manipulation of Online Reviews. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '15)*. ACM, New York, NY, USA, 233–242. https://doi.org/10.1145/2766462.2767742

[27] Geli Fei, Arjun Mukherjee, Bing Liu, Meichun Hsu, Malu Castellanos, and Riddhiman Ghosh. 2013. Exploiting Burstiness in Reviews for Review Spammer Detection. In *Proceedings of AAAI ICWSM*.

[28] Fiverr. [n. d.]. https://www.fiverr.com/.

[29] Jerome Friedman, Trevor Hastie, and Robert Tibshirani. 2010. Regularization Paths for Generalized Linear Models via Coordinate Descent. *Journal of Statistical Software* 33, 1 (2010), 1–22. http://www.jstatsoft.org/v33/i01/

[30] Oana Goga, Howard Lei, Sree Hari Krishnan Parthasarathi, Gerald Friedland, Robin Sommer, and Renata Teixeira. 2013. Exploiting innocuous activity for correlating users across sites. In *Proceedings of the 22nd international conference on World Wide Web*. 447–458.

[31] Atefeh Heydari, Mohammadali Tavakoli, and Naomie Salim. 2016. Detection of Fake Opinions Using Time Series. *Expert Syst. Appl.* 58, C (Oct. 2016), 83–92. https://doi.org/10.1016/j.eswa.2016.03.020

[32] Bryan Hooi, Neil Shah, Alex Beutel, Stephan Günnemann, Leman Akoglu, Mohit Kumar, Disha Makhija, and Christos Faloutsos. 2015. BIRDNEST: Bayesian Inference for Ratings-Fraud Detection. *CoRR* abs/1511.06030 (2015).

[33] Bryan Hooi, Hyun Ah Song, Alex Beutel, Neil Shah, Kijung Shin, and Christos Faloutsos. 2016. FRAUDAR: Bounding Graph Fraud in the Face of Camouflage. In *Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '16)*. ACM, New York, NY, USA, 895–904. https://doi.org/10.1145/2939672.2939747

[34] Paridhi Jain, Ponnurangam Kumaraguru, and Anupam Joshi. 2013. @ i seek'fb. me': Identifying users across multiple online social networks. In *Proceedings of the 22nd international conference on World Wide Web*. ACM, 1259–1268.

[35] Nitin Jindal and Bing Liu. 2008. Opinion spam and analysis. In *Proceedings of the international conference on Web search and web data mining (WSDM '08)*. ACM, New York, NY, USA, 219–230. https://doi.org/10.1145/1341531.1341560

[36] Parisa Kaghazgaran, James Caverlee, and Majid Alfifi. 2017. Behavioral Analysis of Review Fraud: Linking Malicious Crowdsourcing to Amazon and Beyond.. In *Proceedings of ICWSM.*

[37] Parisa Kaghazgaran, James Caverlee, and Anna Squicciarini. 2018. Combating Crowdsourced Review Manipulators: A Neighborhood-Based Approach. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining (WSDM '18)*. ACM, New York, NY, USA, 306–314. https://doi.org/10.1145/3159652.3159726

[38] David R Karger. 1993. Global Min-cuts in RNC, and Other Ramifications of a Simple Min-Cut Algorithm.. In *SODA*, Vol. 93.

[39] Georgios Kontaxis, Iasonas Polakis, Sotiris Ioannidis, and Evangelos P Markatos. 2011. Detecting social network profile cloning. In *Pervasive Computing and Communications Workshops (PERCOM Workshops), 2011 IEEE International Conference on*. IEEE, 295–300.

[40] Srijan Kumar, Justin Cheng, Jure Leskovec, and V.S. Subrahmanian. 2017. An Army of Me: Sockpuppets in Online Discussion Communities. In *Proceedings of the 26th International Conference on World Wide Web (WWW '17)*. International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, Switzerland, 857–866. https://doi.org/10.1145/3038912.3052677

[41] 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 the ACM International Conference on Web Search and Data Mining*. 333–341.

[42] Kyumin Lee, Steve Webb, and Hancheng Ge. 2014. The Dark Side of Micro-Task Marketplaces: Characterizing Fiverr and Automatically Detecting Crowdturfing. https://www.aaai.org/ocs/index.php/ICWSM/ICWSM14/paper/view/8078

[43] Huayi Li, Zhiyuan Chen, Arjun Mukherjee, Bing Liu, and Jidong Shao. 2015. Analyzing and Detecting Opinion Spam on a Large-scale Dataset via Temporal and Spatial Patterns. In *Proceedings of ICWSM*. AAAI Press, 634–637.

[44] Huayi Li, Geli Fei, Shuai Wang, Bing Liu, Weixiang Shao, Arjun Mukherjee, and Jidong Shao. 2017. Bimodal Distribution and Co-Bursting in Review Spam Detection. In *Proceedings of the 26th International Conference on World Wide Web (WWW '17)*. International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, Switzerland, 1063–1072. https://doi.org/10.1145/3038912.3052582

[45] Changchang Liu, Peng Gao, Matthew Wright, and Prateek Mittal. 2015. Exploiting Temporal Dynamics in Sybil Defenses. In *Proceedings of the 22Nd ACM SIGSAC Conference on Computer and Communications Security (CCS '15)*. ACM, New York, NY, USA, 805–816. https://doi.org/10.1145/2810103.2813693

[46] Michael Luca and Georgios Zervas. 2016. Fake It Till You Make It: Reputation, Competition, and Yelp Review Fraud. In *Management Sciences*. 3412–3427.

[47] Arjun Mukherjee, Abhinav Kumar, Bing Liu, Junhui Wang, Meichun Hsu, Malu Castellanos, and Riddhiman Ghosh. 2013. Spotting Opinion Spammers Using Behavioral Footprints. In *Proceedings of ACM KDD.*

[48] Arjun Mukherjee, Bing Liu, and Natalie Glance. 2012. Spotting Fake Reviewer Groups in Consumer Reviews. In *Proceedings of ACM WWW.*

[49] Arjun Mukherjee, Vivek Venkataraman, Bing Liu, and Natalie Glance. 2013. What Yelp Fake Review Filter Might Be Doing. In *Proceedings of the International Conference on Weblogs and Social Media.*

[50] Kazushi Nagayama and Andrew Ahn. 2016. Keeping the Play Store trusted: fighting fraud and spam installs. Android Developers Blog, https://android-developers.googleblog.com/2016/10/keeping-the-play-store-trusted-fighting-fraud-and-spam-installs.html.

[51] Arvind Narayanan, Hristo Paskov, Neil Zhenqiang Gong, John Bethencourt, Emil Stefanov, Eui Chul Richard Shin, and Dawn Song. 2012. On the Feasibility of Internet-Scale Author Identification. In *Proceedings of the IEEE Symposium on Security and Privacy*. 300–314.

[52] Arvind Narayanan and Vitaly Shmatikov. 2008. Robust De-anonymization of Large Sparse Datasets. In *Proceedings of the 2008 IEEE Symposium on Security and Privacy (SP '08)*. IEEE Computer Society, Washington, DC, USA, 111–125.

https://doi.org/10.1109/SP.2008.33

[53] Shashank Pandit, Duen Horng Chau, Samuel Wang, and Christos Faloutsos. 2007. Netprobe: A Fast and Scalable System for Fraud Detection in Online Auction Networks. In *Proceedings of the 16th International Conference on World Wide Web (WWW '07)*. 201–210.

[54] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. 2014. DeepWalk: Online Learning of Social Representations. In *Proceedings of the International Conference on Knowledge Discovery and Data Mining*. 701–710.

[55] Mahmudur Rahman, Bogdan Carbunar, Jaime Ballesteros, George Burri, and Duen Horng (Polo) Chau. 2014. Turning the Tide: Curbing Deceptive Yelp Behaviors. In *Proceedings of the SIAM International Conference on Data Mining (SDM)*.

[56] Mizanur Rahman, Nestor Hernandez, Bogdan Carbunar, and Duen Horng Chau. 2018. Search Rank Fraud De-Anonymization in Online Systems. In *Proceedings of the ACM Conference on Hypertext and Social Media*.

[57] Mizanur Rahman, Ruben Recabarren, Bogdan Carbunar, and Dongwon Lee. 2017. Stateless Puzzles for Real Time Online Fraud Preemption. In *Proceedings of the ACM Web Science Conference (WebSci)*.

[58] Shebuti Rayana and Leman Akoglu. 2015. Collective Opinion Spam Detection: Bridging Review Networks and Metadata. In *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '15)*. ACM, New York, NY, USA, 985–994. https://doi.org/10.1145/2783258.2783370

[59] Brian Reigh. 2017. Fake reviews on the Play Store reportedly growing and getting smarter. *Android Authority* (April 2017). https://www.androidauthority.com/fake-reviews-play-store-reportedly-growing-761928/

[60] Eli Rosenberg. 2017. The Shed at Dulwich' was London's top-rated restaurant. Just one problem: It didn't exist. *The Washington Post* (Dec 2017). https://www.nytimes.com/interactive/2018/01/27/technology/social-media-bots.html

[61] Vlad Sandulescu and Martin Ester. 2015. Detecting Singleton Review Spammers Using Semantic Similarity. In *Proceedings of the 24th International Conference on World Wide Web (WWW '15 Companion)*. ACM, New York, NY, USA, 971–976. https://doi.org/10.1145/2740908.2742570

[62] Giuseppe Silvestri, Jie Yang, Alessandro Bozzon, and Andrea Tagarelli. 2015. Linking Accounts across Social Networks: the Case of StackOverflow, Github and Twitter.. In *KDWeb*. 41–52.

[63] Thamar Solorio, Ragib Hasan, and Mainul Mizan. 2013. A case study of sockpuppet detection in wikipedia. In *Proceedings of the Workshop on Language Analysis in Social Media*. 59–68.

[64] Jonghyuk Song, Sangho Lee, and Jong Kim. 2015. CrowdTarget: Target-based Detection of Crowdturfing in Online Social Networks. In *Proceedings of the 22Nd ACM SIGSAC Conference on Computer and Communications Security (CCS '15)*. ACM, New York, NY, USA, 793–804. https://doi.org/10.1145/2810103.2813661

[65] Tao Stein, Erdong Chen, and Karan Mangla. 2011. Facebook Immune System. In *Proceedings of the 4th Workshop on Social Network Systems*. 8:1–8:8.

[66] Brad Stone and Matt Richtel. 2007. The Hand That Controls the Sock Puppet Could Get Slapped. The New York Times, https://www.nytimes.com/2007/07/16/technology/16blog.html.

[67] Gianluca Stringhini, Pierre Mourlanne, Gregoire Jacob, Manuel Egele, Christopher Kruegel, and Giovanni Vigna. 2015. EVILCOHORT: Detecting Communities of Malicious Accounts on Online Services. In *24th USENIX Security Symposium (USENIX Security 15)*. USENIX Association, Washington, D.C., 563–578. https://www.usenix.org/conference/usenixsecurity15/technical-sessions/presentation/stringhini

[68] Gianluca Stringhini, Gang Wang, Manuel Egele, Christopher Kruegel, Giovanni Vigna, Haitao Zheng, and Ben Y. Zhao. 2013. Follow the Green: Growth and Dynamics in Twitter Follower Markets. In *Proceedings of the 2013 Conference on Internet Measurement Conference (IMC '13)*. ACM, New York, NY, USA, 163–176. https://doi.org/10.1145/2504730.2504731

[69] Jessica Su, Ansh Shukla, Sharad Goel, and Arvind Narayanan. 2017. Deanonymizing Web Browsing Data with Social Networks. In *Proceedings of the 26th International Conference on World Wide Web (WWW '17)*. International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, Switzerland, 1261–1269. https://doi.org/10.1145/3038912.3052714

[70] Kurt Thomas, Damon McCoy, Chris Grier, Alek Kolcz, and Vern Paxson. 2013. Trafficking Fraudulent Accounts: The Role of the Underground Market in Twitter Spam and Abuse. In *Proceedings of the 22Nd USENIX Conference on Security (SEC'13)*. USENIX Association, Berkeley, CA, USA, 195–210. http://dl.acm.org/citation.cfm?id=2534766.2534784

[71] Tian Tian, Jun Zhu, Fen Xia, Xin Zhuang, and Tong Zhang. 2015. Crowd Fraud Detection in Internet Advertising. In *Proceedings of the 24th International Conference on World Wide Web (WWW '15)*. International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, Switzerland, 1100–1110. https://doi.org/10.1145/2736277.2741136

[72] Nguyen Tran, Bonan Min, Jinyang Li, and Lakshminarayanan Subramanian. 2009. Sybil-resilient Online Content Voting. In *Proceedings of the 6th USENIX Symposium on Networked Systems Design and Implementation (NSDI'09)*. USENIX Association, Berkeley, CA, USA, 15–28. http://dl.acm.org/citation.cfm?id=1558977.1558979

[73] Giridhari Venkatadri, Oana Goga, Changtao Zhong, Bimal Viswanath, Krishna P. Gummadi, and Nishanth Sastry. 2016. Strengthening Weak Identities Through Inter-Domain Trust Transfer. In *Proceedings of the 25th International Conference on World Wide Web (WWW '16)*. 1249–1259.

[74] Colleen Wamback. 2017. WPI Research Detects When Online Reviews and News Are a Paid-for Pack of Lies. *Worcester Polytechnic Institute* (November 2017). https://www.wpi.edu/news/wpi-research-detects-when-online-reviews-and-news-are-paid-pack-lies

[75] Binghui Wang, Neil Zhenqiang Gong, and Hao Fu. 2017. GANG: Detecting Fraudulent Users in Online Social Networks via Guilt-by-Association on Directed Graphs. In *Proceedings of ICDM*.

[76] Gang Wang, Tianyi Wang, Haitao Zhang, and Ben Y. Zhao. 2014. Man vs. Machine: Practical Adversarial Detection of Malicious Crowdsourcing Workers. In *Proceedings of the 23rd USENIX Conference on Security Symposium*. 239–254.

[77] Guan Wang, Sihong Xie, Bing Liu, and Philip S. Yu. 2011. Review Graph Based Online Store Review Spammer Detection. *IEEE ICDM* (2011).

[78] Emma Woollacott. 2017. Amazon's Fake Review Problem Is Now Worse Than Ever, Study Suggests. *Forbes* (September 2017). https://www.forbes.com/sites/emmawoollacott/2017/09/09/exclusive-amazons-fake-review-problem-is-now-worse-than-ever/#77a903177c0f

[79] Zhen Xie and Sencun Zhu. 2014. GroupTie: Toward Hidden Collusion Group Discovery in App Stores. In *Proceedings of the 2014 ACM Conference on Security and Privacy in Wireless &#38; Mobile Networks (WiSec '14)*. ACM, New York, NY, USA, 153–164. https://doi.org/10.1145/2627393.2627409

[80] Zhen Xie, Sencun Zhu, Qing Li, and Wenjing Wang. 2016. You Can Promote, but You Can'T Hide: Large-scale Abused App Detection in Mobile App Stores. In *Proceedings of the 32nd Annual Conference on Computer Security Applications (ACSAC '16)*. ACM, New York, NY, USA, 374–385. https://doi.org/10.1145/2991079.2991099

[81] Zhi Yang, Christo Wilson, Xiao Wang, Tingting Gao, Ben Y Zhao, and Yafei Dai. 2014. Uncovering social network sybils in the wild. *ACM Transactions on Knowledge Discovery from Data (TKDD)* 8, 1 (2014), 2.

[82] Yuanshun Yao, Bimal Viswanath, Jenna Cryan, Haitao Zheng, and Ben Y. Zhao. 2017. Automated Crowdturfing Attacks and Defenses in Online Review Systems. In *Proceedings of the ACM Conference on Computer and Communications Security*. 1143–1158.

[83] Junting Ye, Santhosh Kumar, and Leman Akoglu. 2016. Temporal Opinion Spam Detection by Multivariate Indicative Signals. In *Proceedings of ICWSM*. AAAI Press, 743–746.

[84] Haifeng Yu, Phillip B. Gibbons, Michael Kaminsky, and Feng Xiao. 2010. SybilLimit: A Near-optimal Social Network Defense Against Sybil Attacks. *IEEE/ACM Trans. Netw.* 18, 3 (June 2010), 885–898. https://doi.org/10.1109/TNET.2009.2034047

[85] Haifeng Yu, Michael Kaminsky, Phillip B. Gibbons, and Abraham D. Flaxman. 2008. SybilGuard: Defending Against Sybil Attacks via Social Networks. *IEEE/ACM Trans. Netw.* 16, 3 (June 2008), 576–589. https://doi.org/10.1109/TNET.2008.923723

[86] Reza Zafarani and Huan Liu. 2013. Connecting users across social media sites: a behavioral-modeling approach. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*. 41–49.

[87] Haizhong Zheng, Minhui Xue, Hao Lu, Shuang Hao, Haojin Zhu, Xiaohui Liang, and Keith Ross. 2018. Smoke Screener or Straight Shooter: Detecting Elite Sybil Attacks in User-Review Social Networks. In *Proceedings of the Network and Distributed System Security Symposium (NDSS)*.

A PROOF OF LEMMA 4.2

Proof. We note that (6) can be expressed in matrix form as:

$$(\text{diag}(\mathbf{p}) - \mathbf{q}\mathbf{p}^{\mathsf{T}})\mathbf{r} = \left(1 - \sum_{i=1}^{m} P_i\right)\mathbf{q} \tag{8}$$

where $\mathbf{p} = [P_1, \ldots, P_m]^{\mathsf{T}}$, $\mathbf{q} = [q_1, \ldots, q_m]^{\mathsf{T}}$, $\mathbf{r} = [r_1, \ldots, r_m]^{\mathsf{T}}$ and $\text{diag}(\mathbf{p})$ is the $m \times m$ diagonal matrix with $\text{diag}(\mathbf{p})_{ii} = P_i$.

We also note that:

$$\begin{aligned}\mathbf{A} &= \text{diag}(\mathbf{p}) - \mathbf{q}\mathbf{p}^{\mathsf{T}} \\ &= \text{diag}(\mathbf{p}) - \mathbf{q}\mathbf{1}^{\mathsf{T}}\,\text{diag}(\mathbf{p}) \\ &= (\mathbf{I} - \mathbf{q}\mathbf{1}^{\mathsf{T}})\,\text{diag}(\mathbf{p})\end{aligned}$$

and therefore:

$$\begin{aligned}
\det(\mathbf{A}) &= \det((\mathbf{I} - \mathbf{q}\mathbf{1}^\intercal)\,\mathrm{diag}(\mathbf{p})) \\
&= \det(\mathbf{I} - \mathbf{q}\mathbf{1}^\intercal)\prod_{i=1}^{m} P_i \\
&= \left(1 - \sum_{i=1}^{m} q_i\right)\prod_{i=1}^{m} P_i
\end{aligned}$$

where $\mathbf{1} = [1, \dots, 1]^\intercal$ and the last equality follows from Sylvester's determinant theorem.

Let $\mathbf{A_t}$ be the matrix formed by replacing the t-th column of $\mathbf{A}$ by the column vector $\left(1 - \sum_{i=1}^{m} P_i\right)\mathbf{q}$. Thus,

$$\mathbf{A_t} = \left[\mathbf{a_1}, \dots, \left(1 - \sum_{i=1}^{m} P_i\right)\mathbf{q}, \dots, \mathbf{a_m}\right]$$

where $\mathbf{a_t}$ represents the $t-$th column of matrix $\mathbf{A}$. We also note that

$$\begin{aligned}
\mathbf{a_t} &= P_t\mathbf{e_t} - P_t\mathbf{q} \\
\mathbf{q} &= \mathbf{e_t} - \frac{1}{P_t}\mathbf{a_t}
\end{aligned}$$

where $\mathbf{e_t}$ denotes the vector with a 1 in the t-th coordinate and 0's elsewhere. By properties of the determinant, it is plain that:

$$\begin{aligned}
&\det(\mathbf{A_t}) = \\
&\left(1 - \sum_{i=1}^{m} P_i\right)\det([\mathbf{a_1}, \dots, \mathbf{q}, \dots, \mathbf{a_m}]) \\
&= -\frac{\left(1 - \sum_{i=1}^{m} P_i\right)}{P_t}\det([\mathbf{a_1}, \dots, \mathbf{a_t} - P_t\mathbf{e_t}, \dots, \mathbf{a_m}]) \\
&= -\frac{\left(1 - \sum_{i=1}^{m} P_i\right)}{P_t}(\det(\mathbf{A}) - P_t\det([\mathbf{a_1}, \dots, \mathbf{e_t}, \dots, \mathbf{a_m}])) \\
&= -\frac{\left(1 - \sum_{i=1}^{m} P_i\right)}{P_t}(\det(\mathbf{A}) - P_t(-1)^{t+t}\mathrm{Minor}(\mathbf{A})_{tt}) \\
&= -\frac{\left(1 - \sum_{i=1}^{m} P_i\right)}{P_t}\left[\left(1 - \sum_{i=1}^{m} q_i\right)\prod_{i=1}^{m} P_i - P_t\left(1 - \sum_{i\neq t} q_i\right)\prod_{i\neq t} P_i\right] \\
&= -\frac{\left(1 - \sum_{i=1}^{m} P_i\right)}{P_t}\left[\left(1 - \sum_{i=1}^{m} q_i - 1 + \sum_{i\neq t} q_i\right)\prod_{i=1}^{m} P_i\right] \\
&= \frac{q_t(1 - \sum_{i=1}^{m} P_i)\prod_{i=1}^{m} P_i}{P_t}
\end{aligned}$$

By Cramer's rule it follows that:

$$r_t = \frac{\det(\mathbf{A_t})}{\det(\mathbf{A})} = \frac{q_t\left(1 - \sum_{i=1}^{m} P_i\right)}{P_t\left(1 - \sum_{i=1}^{m} q_i\right)}$$

We are left to prove that $\mathrm{Minor}(\mathbf{A})_{tt} = (1 - \sum_{i\neq t} q_i)\prod_{i\neq t} P_i$, but this follows from the construction of $\mathbf{A}$. Take
$\mathbf{p}_{-t} = [P_1, \dots, P_{t-1}, P_{t+1}, \dots, P_m]^\intercal$ and
$\mathbf{q}_{-t} = [q_1, \dots, q_{t-1}, q_{t+1}, \dots, q_m]^\intercal$, we then have:

$$\begin{aligned}
\det(\mathbf{A}_{-t,-t}) &= \det(\mathrm{diag}(\mathbf{p}_{-t}) - \mathbf{q}_{-t}\mathbf{p}_{-t}^\intercal) \\
&= \left(1 - \sum_{i\neq t} q_i\right)\prod_{i\neq t} P_i = \mathrm{Minor}(\mathbf{A})_{tt}
\end{aligned}$$

□