Combating Crowdsourced Review Manipulators: A Neighborhood-Based Approach

Parisa Kaghazgaran
Texas A& M University
College Station, TX
kaghazgaran@tamu.edu

James Caverlee
Texas A& M University
College Station, TX
caverlee@tamu.edu

Anna Squicciarini
Pennsylvania State University
State College, PA
asquicciarini@ist.psu.edu

ABSTRACT

We propose a system called **TwoFace** to uncover crowdsourced review manipulators who target online review systems. A unique feature of TwoFace is its three-phase framework: (i) in the first phase, we intelligently sample actual evidence of manipulation (e.g., review manipulators) by exploiting low moderation crowdsourcing platforms that reveal evidence of strategic manipulation; (ii) we then propagate the suspiciousness of these seed users to identify similar users through a random walk over a "suspiciousness" graph; and (iii) finally, we uncover (hidden) distant users who serve structurally similar roles by mapping users into a low-dimensional embedding space that captures community structure. Altogether, the TwoFace system recovers 83% to 93% of all manipulators in a sample from Amazon of 38,590 reviewers, even when the system is seeded with only a few samples from malicious crowdsourcing sites.

ACM Reference Format:
Parisa Kaghazgaran, James Caverlee, and Anna Squicciarini. 2018. Combating Crowdsourced Review Manipulators: A Neighborhood-Based Approach. In *WSDM 2018: WSDM 2018: The Eleventh ACM International Conference on Web Search and Data Mining , February 5–9, 2018, Marina Del Rey, CA, USA.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3159652.3159726

1 INTRODUCTION

User review aggregators like Amazon, Netflix, and Yelp play a central role in how we decide what movies to view, products to purchase, restaurants to patronize, and even doctors to visit. With this importance, the reviews at the heart of these aggregators are vulnerable to manipulation [32]. This manipulation – often in the form of artificial ratings and reviews – can degrade trust in these online platforms and in their products and services. Indeed, many previous efforts have explored methods to uncover this manipulation, often by applying machine learning or graph-based algorithms, e.g., [23], [31], [1], [25], [12], [34]. These methods typically are built and validated over a dataset of "known" manipulated reviews. And yet, most make one of several critical assumptions:

- *Manual labeling of fake reviews:* In the first approach, judges – often either researchers themselves or a team of labelers at a review site – assess individual reviews to determine if they are fake or not [17], [24]. These methods sometimes rely on unsupervised algorithms (e.g., the output of a proprietary company algorithm) or on manual and possibly error-prone labeling of fake reviews without access to a ground truth of the actual intent of the review writers themselves.
- *Ex post analysis of outliers:* A second approach is to validate detection algorithms through ex post analysis of suspicious reviews. Typically, an algorithm is run over a collection of reviews and the top-ranked results are examined [31], [1], [34]. This approach tends to focus on highly-visible fake behaviors (e.g., a reviewer who posts dozens of reviews in a period of minutes), but may miss more subtle behaviors.
- *Simulation of bad behavior:* A recent third approach is to *simulate* the behaviors of malicious workers [19]. In this approach, volunteers are asked to imagine themselves as fake review writers and then post fake reviews. While encouraging, this method necessarily lacks insight into the strategies and motivations of actual fake review writers.

We seek to complement these foundational studies by leveraging a large collection of *actual review manipulators*. In contrast to previous efforts where the critical knowledge of intent to deceive is missing, we collect a set of review manipulators *for whom we have strong evidence of intent to deceive*. Concretely, we monitor low moderation crowdsourcing sites like RapidWorkers, ShortTask, and Microworkers, where attacks on review sites can be launched by malicious paymasters. By tracking these workers from the crowdsourcing platform to a target review site like Amazon, we can identify deceptive review manipulators. In our analysis (see Figure 4) we find that these review manipulators engage in a deceptive mix of legitimate reviews (to build reputation and trust) and fraudulent reviews (to cash in on this trust). This two-faced behavior poses great challenges to traditional detection mechanisms since the fraudulent reviews may be masked by the large number of legitimate reviews. Indeed, through our investigation of 300 target products on Amazon and their reviewers, we find that although malicious and benign reviewers occasionally behave differently in terms of rating, burstiness of reviews, review length and so on, these traditional features do not provide strong power to distinguish between review manipulators and legitimate reviewers.

Hence, we propose in this paper the TwoFace system to uncover crowdsourced review manipulators who target online review systems. First, we intelligently sample actual evidence of manipulation (e.g., review manipulators) by exploiting low moderation crowdsourcing platforms that reveal evidence of strategic manipulation. Since we find that many traditional features do not have strong power to identify two-faced reviewers with a mix of legitimate and deceptive reviews, we propose to exploit *neighborhood-based*

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

ACM ISBN 978-1-4503-5581-0/18/02…$15.00
https://doi.org/10.1145/3159652.3159726

characteristics of the reviewers themselves. The intuition is that although individual behaviors may be easy to mask, the collective campaign organization may be uncovered by exploiting the network around reviewers and products. Based on this intuition, we then propagate the suspiciousness of the original seed users to identify "similar" users through a random walk over a "suspiciousness" graph. In this way, we can identify users who may be likely to participate in deceptive review campaigns. Note, however, that users who are distant in the "suspiciousness" graph will rarely be considered as deceptive reviewers. Hence, our final step is to uncover these (hidden) distant users who serve structurally similar roles by mapping users into a low-dimensional embedding space that captures community structure. In this way, we can distinguish deceptive reviewers through their community embedding structure.

Through experiments, we evaluate the TwoFace system over a sample from Amazon of 38,590 reviewers and 580,000 reviews. We discover that social features perform much stronger in distinguishing manipulators and regular reviewers compared to behavioral features. We also find that manipulators and regular users behave relatively similar in terms of rating, review burstiness, and so on. In addition, manipulators who participate in multiple crowdsourcing campaigns play a key role in uncovering other malicious but not obviously fraudulent reviewers. We observe that many of the reviewers who write only one review on target products are more similar to actual malicious reviewers than regular users. Altogether, the TwoFace system recovers 83% to 93% of all manipulators, even when the system is seeded with only dozens of examples from malicious crowdsourcing sites, and outperforms a state-of-the-art baseline.

2 RELATED WORK

Research in review manipulation typically focuses on identifying either fake reviews or the coordinated activities of fraudulent review groups. We highlight here several major research thrusts:

In the first, the *content of the reviews* themselves may offer indicators of "spam-ness" or deception. Many works here develop text or NLP models to distinguish fake reviews from legitimate reviews, e.g., [18–20].

Complementing the review text itself, the *behavioral signals* revealed by the reviewers or the reviews may yield clues as to which are fraudulent [13]. For example, [2] detects lock-step behaviors using temporal patterns in the Facebook graph wherein users and pages are nodes and "Likes" are edges. A few works address the temporal behavior of reviews [5, 33], for example by finding burstiness or other synchronized behaviors. Bayesian approaches have been applied in rating time-series to detect anomalies [10].

In another perspective, *graph-based approaches* have become popular in fraud or anomaly detection by modeling users as nodes and their relationships as edges. Approaches include spectral methods like eigen-decomposition or singular value decomposition to cluster similar nodes in the graph [12, 23, 25]. Iterative models to label nodes as trustworthy or non-trustworthy are proposed in [6, 31]. Markov Random Fields (MRF) and belief propagation approaches have been used to identify dense and most likely suspicious subgraphs [1, 21]. In another view, malicious users are detected through graph-based measures like neighborhood diversity [34].

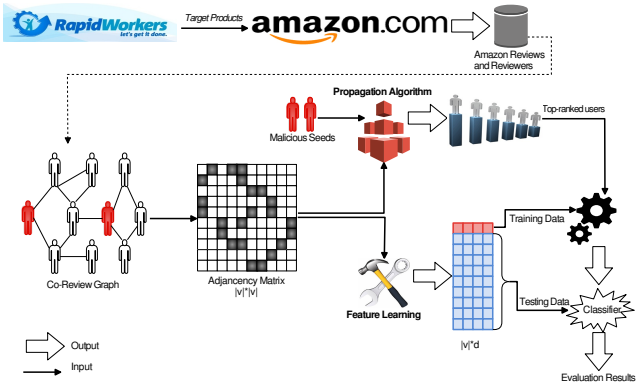


Figure 1: TwoFace overall framework.

A related direction is in *detecting dense blocks* in a review-rating matrix [11, 26, 27]. Extraordinary dense blocks correspond to groups of users with lockstep behaviors, e.g., [11]. Moreover, this method has been lately extended from matrix to tensor representation to incorporate more dimensions (e.g., temporal aspects) [26, 27]. However, these approaches may have difficulty in detecting subtle attacks where there are not such clearly defined dense block characteristics. Finally, some recent work focuses on detecting not just fake reviews but fake crowd activities, e.g., [30].

3 TWOFACE SYSTEM DESIGN

In this section, we introduce the overall system design for TwoFace (see Figure 1). TwoFace is designed to be deployed in real-world scenarios with the following characteristics:

- First, we assume there is only some *small, partial evidence of review manipulation*. In contrast to many previous efforts, we make no assumption that there is a large, curated collection of positive and negative examples of review manipulation. This corresponds to real-world scenarios of evolving and newly emerging types of review manipulation.
- Second, we target scenarios in which review manipulation engages in *difficult-to-detect behaviors*. That is, the review manipulators may engage in a mix of legitimate and deceptive reviews, so that many traditional content-based or dense-block methods may have difficulty in finding them.
- Finally, TwoFace is *recall-focused*. Our goal of uncovering review manipulators is to identify as many as possible out of the entire space of reviewers, in contrast to precision-focused approaches that may miss the vast majority of manipulators. We assume system operators can expend resources to further examine these potential manipulators.

In the following, we introduce each of the key components of TwoFace, before turning to a comprehensive evaluation in Section 4.

3.1 Identifying Suspicious Seeds

Our first task is to identify suspicious users. As we have argued, most existing methods for identifying deceptive reviews and reviewers have focused on indirect approaches, where the ground truth is necessarily an approximation. For example, previous efforts

What is expected from workers?
Read the product description before writing down a review.
Go to https://goo.gl/7QfW0h.
Leave a relevant 5-star review with at least 40 words.
Provide proof that you left the review yourself.

Figure 2: An example crowdsourcing task.

★★★★★ **Best cortisol blocker to reduce high levels of stress**, *July 24, 2017*
My stress levels have increased lately due to heavy work loads in my office and that directly impacting my life. I have gained weight and easily gets tired. I have tried many products to reduce my cortisol levels which is causing stress, but those products don't fetch any results and based on my uncle recommendation I have tried this product and it has relieved my stress and assisted in returning my cortisol levels to a more natural state. I feel more energized and active than before and the product also helped in losing body fat. Must try the product.

Figure 3: An example review written by a crowd worker.

have identified users with many reviews in a short burst of time or with a skewed rating distribution as suspicious [10, 11, 14, 27]. In contrast, many seemingly "normal" reviewers may be overlooked in practice if their deceptive reviews are blended in with legitimate ones.

Sampling Crowdsourcing Sites for Identifying Suspicious Users. Alternatively, we aim to identify users who have been tasked by a crowdsourcing site. In this way, we can identify with high confidence users who are indeed deceptive even if the majority of their reviews are legitimate. As an example, consider the task posted to RapidWorkers in Figure 2. This type of task is common on RapidWorkers and related sites like ShortTask and Microworkers. As an example of the type of review that is created by crowd worker, Figure 3 shows a sample of a crowdsourced review for a "cortisol supplement" product sold by Amazon. On examination, this review displays few (if any) clear signals of it being fraudulently written.

By monitoring such requests, we can begin to study the behaviors of fraudulent review writers. Although not representative of all types of manipulation, this approach does provide the tantalizing opportunity to study malicious behaviors in the wild. To collect suspicious users, TwoFace implements two crawlers: one for identifying crowdsourcing tasks and the other crawler has been designed to collect reviews from a target site. In this paper, we focus on tasks posted to RapidWorkers that target Amazon and simultaneously ask for fake reviews, similar to the example shown in Figure 2. Note that there are many such sites[1] and many additional targets (e.g., Yelp, App Store, Play Store).

Concretely, we crawl all such tasks from July 2016 to February 2017. In total, we identify 300 unique Amazon product IDs. By linking these IDs to Amazon, we crawl all reviews associated with each targeted product. Altogether we find 21,162 reviews that have been written by 12,212 unique reviewers. Typically, a target on these crowdsourcing sites (e.g., a product on Amazon) may be subject to dozens of fake reviews. We find that the number of required fake reviews requested by paymasters varies: the average is 13, but some paymasters ask for only 1, while the maximum requester asked for 75 reviews. Contrary to previous efforts in dense block detection or in methods that require high numbers of fake ratings, we find that 84% of the tasks require fewer than 20 fake reviews. This indicates the challenge in identifying fraudulent reviewers.

Two-Faced Reviewers. For each reviewer we encounter, we additionally collect all of their reviews (which may include products beyond those targeted by these crowdsourcing sites). Ultimately, our dataset contains the following information: product ID, review ID, reviewer ID, review title, review content, rating, and time-stamp. In total, we obtain 580,000 unique reviews in our original dataset. To investigate the behavior of fraudulent vs non-fraudulent reviewers, we sampled reviewers with at least 10 reviews. In the following, we examine their behavior in terms of rating, review burstiness, review length, self-similarity, and lexical diversity. For non-fraudulent reviewers, we sampled reviewers from the Amazon dataset introduced in [15], who co-reviewed products beyond those crowdsourcing products targeted by a crowdsourcing attack.

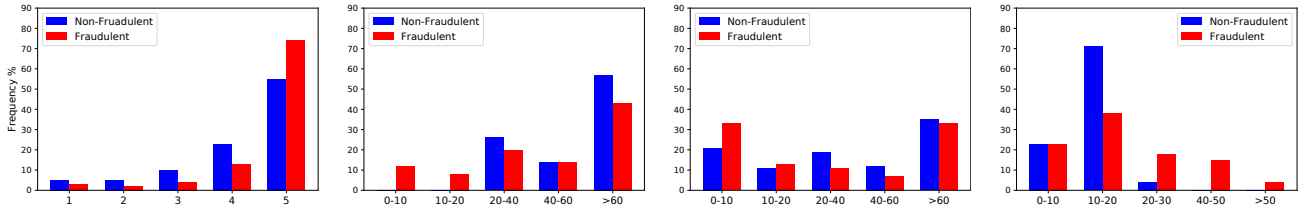
Ratings. We begin with Figure 4a, which shows the ratings distribution for reviews written by our two types of reviewers. Echoing previous studies, e.g., [9], we see that crowdsourcing workers tend to write 4 or 5-star reviews. While crowdsourcing efforts could be targeted at suppressing the ratings for a competitor, we see instead that most efforts focus on promotion. Compared to legitimate reviewers, the rate of 5-star reviews is 20% higher for fraudulent reviewers.

Review Length. We see in Figure 4b the distribution of the review length in terms of number of words between the two groups. We can see that reviews by fraudulent reviewers are relatively short. Even though task requestors require a minimum number of words for payment, these graphs show that all reviews written by fraudulent reviewers are not necessarily a response to crowdsourcing tasks which often require a word count minimum.

Burstiness of Reviews. Intuitively, crowd workers may seek to complete several tasks in a short time to maximize their payoff. Hence, for each reviewer we measure the standard deviation of the timestamp for that person's reviews – we consider the last 10 reviews for reviewers with more than 10 reviews. We plot the distribution for this "burstiness" as seen in Figure 4c. In this case, a small standard deviation corresponds to many reviews being posted in a short time window, whereas a higher standard deviation corresponds to reviews posted over a long time period (and hence, lacking burstiness). Contrary to our hypothesis, burstiness of reviews is not a strong indicator to distinguish fraudulent and non-fraudulent users.

Self-similarity. Finally, we measure how much a reviewer's language mimics previous reviews they have written. Perhaps fraudulent reviewers write according to a simple "template", and so new reviews tend to repeat language used in previous ones. Here, we measure the lexical overlap between each two sequential reviews (r_i, r_j) written by the same reviewer using the Jaccard similarity (JS).

[1] We also checked several other crowdsourcing websites such as ShortTask, Microworkers and Amazon Mechanical Turk (AMT); however, tasks related to promoting products in Amazon are mainly announced on RapidWorkers.



(a) Rating Distribution **(b) Review Length** **(c) Review Burstiness in Days** **(d) Self-similarity**

Figure 4: Traditional features show some differences between fraudulent and non-fraudulent reviewers, but their distinguishing power is weak.

$$JS = \frac{|r_i \cap r_j|}{|r_i \cup r_j|}$$

Figure 4d shows that non-fraudulent tend not to repeat themselves (low Jaccard score); whereas fraudulent reviewers tend to rely on repeated keywords or phrases. Intuitively, reviewers engaged in crowd-launched manipulation tend to mimic themselves over time since they are not actually experienced with the actual product.

Based on this mix of somewhat encouraging features, we evaluate a variety of classifiers (see Section 4). We find that these traditional features do a poor job of distinguishing these two-faced reviewers. Our hypothesis is that traditional signals may fail since these reviewers engage in a mix of legitimate and deceptive reviews. Motivated by these observations, we turn to how we can propagate the suspiciousness of our original seeds for uncovering unknown fraudulent reviewers.

3.2 Propagating Suspiciousness

In this and the following section, we propose two complementary perspectives on propagating the suspiciousness of a known suspicious user. The first – based on a traditional random walk over the user-user graph – exploits the locality of suspiciousness within the graph. The main intuition is that suspicious users will tend to cluster in the graph. The second – based on recent advances in network embeddings – exploits the structure of the graph around suspicious users. So even if two users are distant in the graph, they may be considered similar in terms of their malicious activities. Intuitively, the campaign network structure around fraudulent reviewers in a site like Amazon may be similar even if the specific reviewers have not been encountered before.

Reviewer-Reviewer Graph. Our approach is built on reviewer-reviewer interactions. These interactions are mapped to a co-review graph in which each reviewer is represented as a node and if two reviewers write a review on the same product, then there exists an edge between them. Formally, we model the reviewer network as a graph $G = (V, E)$ wherein V is a set of reviewers and E is a set of undirected edges that connect reviewers. In our experiments (see Section 4), we consider two scenarios: un-weighted and weighted edges. In the first setting, if two users u_i and u_j $(i \neq j)$ have written reviews on multiple common products, this connection is represented as a single edge. In the second setting, we represent the number of common products as a weight value for the edge $w(u_i, u_j)$.

The number of nodes connected to user u is its degree $D(u)$. The co-review matrix corresponding to the un-weighted graph is calculated as:

$$m(u_i, u_j) = \begin{cases} 0 & \text{if } (u_i, u_j) \notin E \\ \frac{1}{D(u_i)} & \text{if} (u_i, u_j) \in E \end{cases}$$

Accordingly, the co-review matrix corresponding to weighted graph is calculated as:

$$m(u_i, u_j) = \begin{cases} 0 & \text{if } (u_i, u_j) \notin E \\ \frac{w(u_i, u_j)}{\sum_{k \in D(u_i)} w(u_i, u_k)} & \text{if} (u_i, u_j) \in E \end{cases}$$

Note that the values in each matrix are normalized transition probabilities.

Random Walk Propagation. Our approach for propagating suspiciousness is inspired by the *TrustRank* algorithm proposed in [6]. We aim to compute a suspiciousness score for each user based on their connectivity to other users in the co-review graph. The intuition is that fraudulent users write reviews on similar products, so they may form a dense sub-graph of suspicious reviewers.

We briefly describe the algorithm and report the results in Section 4. The input to the algorithm is the co-review matrix (m), a seed set (s), teleportation parameter (α), the number of iterations (n), and the number of nodes ($|V|$). The output is a suspiciousness vector (r).

Algorithm 1 Suspiciousness Rank

1: $e = 0_{|V|}$ //Restart Vector
2: **for** $i = 1$ to $|V|$ **do**
3: **if** $u_i \in s$ **then**
4: $e(i) = 1$
5: $e = \frac{e}{|e|}$
6: $r = e$
7: **for** $j = 1$ to n **do**
8: $r = \alpha.m^T.r + (1 - \alpha).e$
9: **return** r

The restart vector e is initialized in steps 2-4 based on the seed set, i.e, the values are one in the corresponding indices of seed set items and zero in other places. Step 5 computes the $l1$ norm of vector e so that the aggregate sum of the vector is equal to 1. In step 6, the suspiciousness score vector r is initialized to e. Finally, the

scores are calculated in steps 7-8 using a biased PageRank with e as a restart vector referring to the seed set of fraudulent reviewers.

In each iteration, the suspiciousness score of a node propagates among its neighbors and is dampened by the teleportation factor α. The score vector r shows the probability of reaching each node when the random walk, rooted at one of the nodes in the seed set, is traversing the graph. In other words, the final scores measure how relatively close a node is to the initial seeds. This approach ranks reviewers based on their level of suspiciousness and suggests reviewers that should be examined further by an expert.

Seed Selection. A key question is how the seeds for the random walk propagation are selected in the first place. First, we need to identify a small set of users as seeds that we certainly know are fraudulent and then propagate their suspiciousness to other users using iterative approaches. In some works, e.g., [6], initial seeds are selected based on human judgment. Here, we consider three different scenarios for seed selection that could arise in practice:

- If we find a user with tens of reviews on targeted products and notice that all of his purchase statuses are unverified, then we conclude that this user is a critical player working on behalf of crowdsourcing websites. We call such reviewers *highly malicious users*. Therefore, we pick reviewers with the maximum number of such reviews as seeds. The intuition is that such reviewers may be connected directly to other potential fraudulent reviewers in the co-review graph. Therefore, propagating their suspiciousness would lead to identifying fraudulent reviewers more accurately. We call this approach the "best" choice of seeds.
- The second approach is to pick a few number of fraudulent reviewers randomly and propagate their suspiciousness. This approach is more indicative of real-world scenarios since we are not always guaranteed to have found the best (most connected) reviewers. For example, malicious users might use different accounts to write fake reviews in order to avoid detection models. We call this approach the "random" choice of seeds.
- The third approach is to pick a few number of fraudulent reviewers randomly among reviewers with only a few reviews on target products. The intuition here is that a system operator at a user review site like Amazon may have discovered some fraudulent reviewers, but only the weakest connected ones. How well does the suspiciousness propagation work in this case? We call this approach the "worst" choice of seeds.

In practice we find that there is a great variance in the quality of seed selection approaches. To illustrate, we show in Figure 5 the Precision@k over the top-k ranked reviewers when we initialize the suspiciousness propagation algorithm with seeds from the best, random, and worst cases. In the real-world cases of random and worst, we see that the fraction of reviewers that are actually fraudulent drops precipitously with an increase in k. That is, while there may be some localness in the co-reviewer graph which helps identify nearby fraudulent reviewers, many fraudulent reviewers are not closely connected to the seeds (hence, the low precision@k as k grows). We do see that in the extreme case of picking the most prolific seeds (which we argue is rare in practice), there is good locality and good precision@k. Even so, for large choices of k, the quality drops off too.

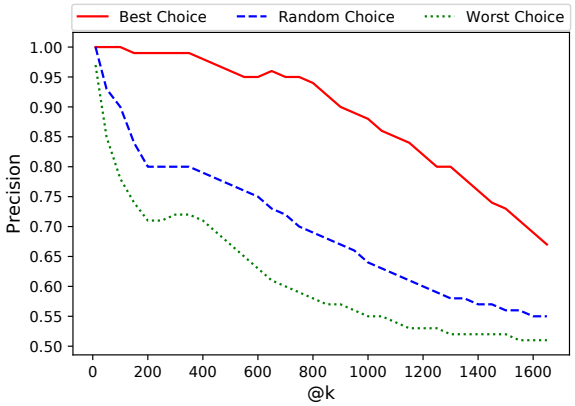


Figure 5: Precision@k for different seed selection approaches.

3.3 Uncovering Distant Users

While traditional random walk methods exploit the locality of suspiciousness within the graph, they may miss reviewers who are not closely connected to the original seeds. In particular, in crowdsourcing scenarios, different fraudulent users may participate in different crowdsourcing tasks, so there would not be a direct link between them. Therefore, we need more sophisticated models to capture this relationship. In this section, we adopt a framework for learning feature representations of nodes in the network. Concretely, we explore using network embeddings to classify similar nodes even if two users are not directly connected but may have the same structural role in the graph. For example, Figure 6 shows that even though node 5 and node 10 act in two distinct communities, they play the same structural role in their own community.

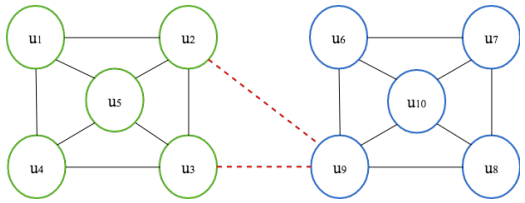


Figure 6: Same structural role in distinct communities. Here nodes 5 and 10 serve similar roles as we surmise different crowd campaigns may also be organized.

Reviewer Graph Embeddings. In node classification, the goal is to predict the most accurate labels for the nodes [29]. On the other hand, in supervised machine learning approaches, a set of informative features is required. When it comes to the problem of node classification in networks, it means feature representation of the nodes is required. The idea of network feature learning is inspired by the recent advances in natural language processing [16] such as the Skip-gram model. In summary, the Skip-gram algorithm goes over the words in a document, and builds a feature vector – the *word embedding*– for every word in the vocabulary such that it can predict its nearby words (i.e., words within a window). The continuous feature representation of words are learned by

optimizing the likelihood objective function – Stochastic Gradient Descent (SGD) – and is based on the distributional hypothesis which declares words in similar contexts tend to be semantically similar [7]. In essence, similar words tend to have similar word neighborhoods.

Inspired by the Skip-gram model, a few recent works have proposed models for feature learning from networks where a network is "document" and the nodes are treated as "words" [4, 22, 28]. Similar to a document that is an ordered sequence of words, a network can be turned into an ordered sequence of nodes.

In this paper, we adopt the recent learning approach proposed in [4] in which feature learning in networks is formulated as a maximum likelihood optimization problem. Function $f :\rightarrow R^d$ maps nodes to their feature representation which are used in classification tasks later on. d indicates the number of dimensions of the feature vector, so f is a matrix of size $|V| \times d$. For each node $u \in V$, $N(u) \subset V$ is the set of its neighborhoods. Depending on the neighborhood sampling strategy, $N(u)$ includes either immediate, embedding, or mixture neighbors. The goal is to optimize the objective function given by f:

$$max_f \sum_{u \in V} log\, p_r(N(u)|f(u))$$

This function maximizes the log-probability of observing $N(u)$ as the neighborhood of node u given its feature representation $f(u)$. In our case, each node is a reviewer. Assuming the probability of observing n_i in neighborhood of u is independent from observing any other node in the neighborhood of u, the probability function can be treated as:

$$pr(N(u)|f(u)) = \prod_{n_i \in N(u)} pr(n_i|f(u))$$

Moreover, node u and its neighbors n_i have equal effect over each other in the feature space. Therefore, their conditional probability is modeled as a *softmax* function with dot product of their feature vectors:

$$pr(n_i|f(u)) = \frac{exp(f(u).f(n_i))}{\sum_{v \in V} exp(f(u).f(v))}$$

Putting it altogether, the objective function in Equation 3.3 can be re-written as follows wherein $Z_u = \sum_{v \in V} exp(f(u).f(v))$:

$$max_f \sum_{u \in V} \Big\{ - log\, Z_u + \sum_{n_i \in N(u)} f(u).f(n_i) \Big\}$$

The remainder is to wisely determine neighborhood nodes i.e., $N(u)$. The notion of neighborhood in a text document is defined by a sliding window over sequential words. However, due to the nature of networks, they do not have such a linear representation and a new notion of neighborhood is needed. Grover et. al in [4] proposed an efficient sampling strategy known as *node2vec*. Traditional neighborhood sampling approaches are Breadth-first Sampling (BFS) and Depth-first Sampling (DFS). BFS samples nodes which are in immediate neighborhood of node u, while DFS samples ones which are in increasing distance from u. *node2vec* enables interpolating between BFS and DFS. Briefly, a biased random walk explores the neighborhood of a node in a mixture of BFS and DFS ways by tuning two parameters – Return and In-out – parameters. In our case, we transform each reviewer (node) into an embedding that captures its neighborhood structure.

Putting it all Together. So far, we have suggested two complementary approaches to address the problem of uncovering fraudulent reviewers. The first one is based on traditional ranking algorithms which scores users based on their suspiciousness and is completely unsupervised. The other is to adopt an embedding feature representation of the reviewers. Here, we propose to combine these two approaches into a supervised classification framework:

- First, we take as input our seeds sampled from a crowdsourcing platform. Typically, we may have one to dozens of seeds.
- Then we propagate the suspiciousness of these reviewers via the random walk.
- After transforming every reviewer into its graph embedding representation, we then train a classifier where the positive examples (the fraudulent reviewers) are drawn from the ranked list of suspicious users. The negative examples (the legitimate reviewers) are drawn randomly from a held out set of reviewers.

4 EXPERIMENTS

In this section, we first introduce our ground truth. Then, we report the results of evaluating the complementary methods on our dataset to identify active users in review manipulation tasks. Finally, we compare our approach with a number of alternatives.

4.1 Data Preparation

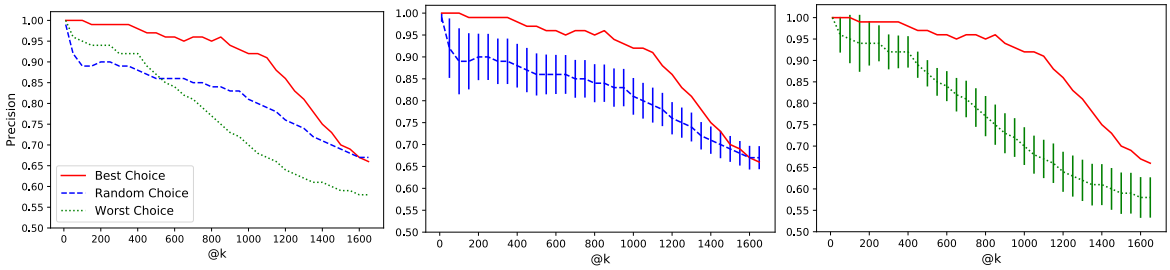
As we mentioned above, there are 12,212 reviewers who wrote a review on target products. We sampled 50% of them randomly. To go one step further, considering the products those sampled reviewers are associated with – 238,000 products – we sampled their reviewers from an Amazon dataset introduced in [8, 15]. Finally, our expanded dataset contains 38,590 reviewers over whom the co-review graph is built.

Ground Truth. In total, Table 1 shows the number of reviewers who wrote a review on a specific number of target products in our dataset (which include 12,212 reviewers in total). For example, 87 reviewers wrote reviews on more than 20 products targeted by the crowdsourcing site. Our dataset naturally contains a mix of reviewers and their reviews: some are legitimate reviews, some are the result of targeted crowdsourced efforts, while others may also be fraudulent but outside the purview of our sampling method (e.g., launched via an unobservable channel like private email). Hence, we create two datasets – one based on a conservative assumption, and one that is a slight relaxation:

# Targets	1	2	3-5	6-8	9-20	>20
# Reviewers	9,096	1,669	1,093	126	141	87

Table 1: Distribution of reviewers based on number of target products they are associated with.

Conservative definition of fraudulent reviewers. We consider a reviewer to be a *fraudulent reviewer* if they have reviewed **two or more products** that have been targeted by a crowdsourcing effort.



(a) All three seed selection approaches. **(b) Random selection with error bars.** **(c) Worst selection with error bars.**

Figure 7: Precision@k for un-weighted graph with different seed selection approaches and 5 initial seeds.

Intuitively, workers may aim to maximize their income by participating in many tasks (and hence, targeting many products). On the other hand, it is unlikely that a random user will write a legitimate review on two different crowdsourcing products in a short period of time, considering Amazon's selection of millions of products [3]. Making this conservative assumption, in our sampled dataset, we identify 1,650 of 38,590 reviewers as *fraudulent reviewers* and label the rest (36,940 reviewers) as *non-fraudulent*. Note that, 4,565 reviewers labeled as non-fraudulent still wrote one review on a target product. Of course, there may still be some unknown fraudulent reviewers in this set of 4,565 reviewers, but it gives us a baseline to compare against the clearly prolific fraudulent reviewers.

Relaxed definition of fraudulent reviewers. In an alternative way to identify fraudulent users, we can relax our conservative assumption and instead label all the reviewers associated with crowdsourcing products (i.e., 6,215 of 38,590) as *fraudulent reviewers* and label the rest as *non-fraudulent reviewers*. In this way, any reviewer who has reviewed a targeted product is labeled as fraudulent. While certainly overstating the number of fraudulent reviewers, this relaxed definition may give us more insights into the capability of our approach.

4.2 Propagating suspiciousness

Revisiting our initial approach to identify fraudulent reviewers by propagating suspiciousness, we report here an additional experiment where we consider five seed users. Again, we identify the best, random, and worst choice of seeds. We repeat this selection 20 times (since users are randomly chosen either from the entire set or from the least connected reviewers) and report in Figure 7 the variation of precision@k for different approaches in picking initial seeds. Here, we report all three approaches in (a), then we show the random approach with error bars in (b), and the worst approach with error bars in (c). The variability suggests that seed selection alone cannot identify fraudulent reviewers. This echoes our previous figure (see Figure 5), yet here we see that the "worst" approach can sometimes do better than random as we increase k. We attribute this result to the fact that some of the fraudulent reviewers have been more active in crowdsourcing manipulation in the past and so they are uncovered as actual fraudulent reviewers at $k <= 500$.

4.3 TwoFace detection

Given these results, we now turn to evaluating the quality of the end-to-end TwoFace system. Recall that TwoFace takes as input the seeds, the suspiciousness propagated scores, and the graph embeddings for the co-review graph.

Choice of Classifier. The first question is what classification algorithm to apply for distinguishing between fraudulent and non-fraudulent reviewers? Here, we consider six alternatives: logistic regression, SVM, Naive Bayes, Decision Tree, Random Forest, and a one-class version of SVM. One-class SVM builds a classifier using instances from only one class – fraudulent in our scenario – and then classifies new instances from both classes. We include this alternative to validate our use of legitimate reviewers in the training, even though our sample of legitimate reviewers is taken from a random sample of Amazon (and so, may erroneously include some fraudulent reviewers).

Figure 8 shows the performance of different classifiers in the presence of 10% labeled data, where the results are averaged over 20 runs. We report the Precision and Recall for the Fraudulent class as well as F1-macro for the whole dataset. It should be noted that the other class which is abundant (95% of the dataset) always has high Precision and Recall. We observe that Logistic Regression and one-class SVM give us higher Recall, 91% and 88% respectively, but Logistic Regression performs better in Precision. While Random Forest and SVM give higher Precision, 84% and 73% respectively, they have a poor Recall i.e., many of fraudulent users remain unidentified. Therefore, we do our further analysis using Logistic Regression.

How Many Seeds Do We Need? In practice, the TwoFace system may have access to only a small fraction of all fraudulent reviewers. Hence, we study here the impact of the number of fraudulent reviewers on the quality of classification. Specifically, we consider samples of our entire dataset ranging from 1% to 10% of all reviewers; in this case, 1% corresponds to about 16 fraudulent reviewers and 370 non-fraudulent ones. We additionally consider both the weighted and unweighted graph for suspiciousness propagation as input to the method. Table 2 shows their performance in the presence of 1% to 10% of labeled data. As we increase the number of seeds, Recall increases from 76% to 92% and 83% to 93% in unweighted and weighted cases respectively while Precision remains relatively constant – 30% to 35%. This encouraging result shows that TwoFace can obtain high Recall even in the presence of very

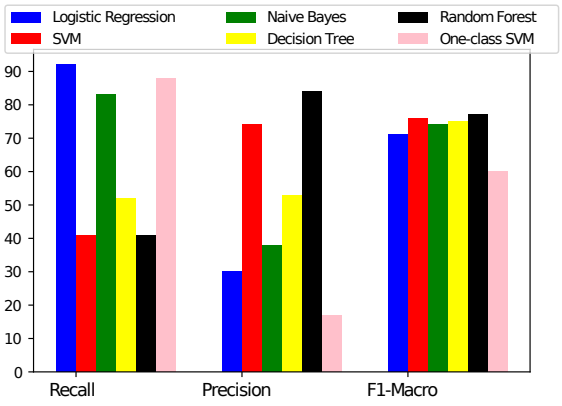


Figure 8: Comparing classifiers.

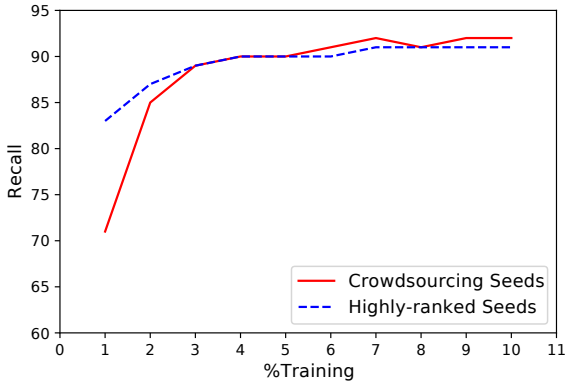


Figure 9: Impact of suspiciousness propagation in identifying fraudulent users

few seeds. Since the weighted and un-weighted cases perform relatively close, we focus our discussion on the un-weighted graph going forward.

Training	1%	2%	3%	4%	5%	6%	7%	8%	9%	10%
un-Weighted Graph										
Recall	76	85	88	90	91	91	92	92	92	92
Precision	34	33	32	30	30	30	30	30	30	30
F1-macro	72	71	70	70	70	70	70	70	70	70
Weighted Graph										
Recall	83	88	89	89	92	92	92	93	93	93
Precision	35	33	34	34	32	32	32	32	32	32
F1-macro	73	72	72	72	71	71	72	71	71	72

Table 2: Increasing the number of seed users available to TwoFace.

The Impact of Suspiciousness Propagation. In an earlier experiment, we saw that suspiciousness propagation alone is insufficient for uncovering fraudulent reviewers. Here, we investigate the impact of using the highly-ranked reviewers that are output from the suspiciousness ranking approach as input to our TwoFace classification. That is, do we really need to propagate suspiciousness? Or can we just use the initial seeds we sample from the crowdsourcing platform alone? In Figure 9, we show the impact of suspiciousness propagation on feature embedding classification. For example, if we rely only on the original seeds from the crowdsourcing platform we achieve around 71% for Recall of fraudulent reviewers; in contrast, if we feed a small number of highly-ranked users from the suspiciousness propagation algorithm into classifier, we see that Recall jumps to 83% with just 1% of seeds. This finding demonstrates the importance of propagating the suspiciousness of these seed users through the random walk over the co-review graph.

Relaxing the Ground Truth. As we can see from Table 2 Precision is in the 30% range which may seem surprising (even though Recall is quite high). This means that many of the non-fraudulent users have been misclassified as fraudulent, which could burden resources that are required to further examine these reviewers. To further explore this issue, we evaluate the TwoFace approach on the alternative "relaxed" ground truth in which every reviewer with at least one review on a target product is considered as fraudulent. In this alternative scenario, we find in Table 3 that precision jumps to 77%, while Recall increases even more to around 90%. This encouraging result suggests that many of the reviewers who only reviewed one target product historically are connected to other fraudulent reviewers and may need further examination by an expert.

	Recall	Precision	F1-macro
Conservative	83	35	72
Relaxed	91	77	89

Table 3: Relaxing the ground truth definition.

Comparing with Alternatives. Finally, we compare the proposed TwoFace system with alternative methods including classification over traditional features and the state-of-the-art D-cube method [27]. For traditional features, we adopt the standard features described in our previous analysis of fraudulent reviewers, including rating, burstiness of reviews, review length, and self-similarity. We evaluate a variety of classifiers and report the best results which are from Logistic Regression. For D-cube – a dense block detection approach – we try many different number of dense blocks and report its best results when the number of blocks is 30 and 40. For a fair comparison, we also consider the relaxed ground truth for these methods. For TwoFace, we adopt Logistic Regression and use the highly-ranked users from our suspiciousness propagation as seeds.

	Recall	Precision	F1-macro
TwoFace System	**91**	**77**	**89**
Traditional Features	61	24	54
D-cube [27]/ 30 blks	69	34	50
D-cube [27]/ 40 blks	82	24	64

Table 4: Comparison of TwoFace with alternatives.

Table 4 shows that TwoFace system outperforms the two other approaches. The low performance of traditional features – 61% Recall and 24% Precision – indicates that fraudulent reviewers

do not always behave abnormally and their rating distribution or burstiness might be similar to non-fraudulent reviewers. On the other hand the D-cube approach [27] which aims to detect dense blocks, i.e., a group of reviewers who write a review on a specific number of products in a short time, is the most similar scenario to crowdsourcing manipulation. It takes number of blocks as input and returns the most dense blocks. As a result, by increasing the number of blocks it returns more fraudulent reviewers – 82% versus 69% Recall with 40 and 30 blocks respectively. However, by increasing the number of blocks, we see that many non-fraudulent reviewers will be misclassified as fraudulent – 24% and 34% Precision. Since crowdsourcing campaigns do not form dense blocks, we see that TwoFace provides the best overall performance with 91% Recall and 77% Precision.

5 CONCLUSION AND FUTURE WORK

We have explored how monitoring tasks on sites like RapidWorkers can uncover fraudulent reviewers on sites like Amazon. The proposed TwoFace framework complements previous efforts by providing a new approach for identifying these types of reviewers. The main intuition is to: (i) exploit the locality of suspiciousness within the graph through a random walk to find suspicious users who tend to cluster; and (ii) exploit the structure of the graph around suspicious users to uncover campaign network structures for identifying fraudulent reviewers who are distant in the graph. Our results are encouraging, indicating that TwoFace can indeed uncover many fraudulent reviewers. In our ongoing work, we are expanding our coverage both in terms of crowdsourcing sites and targets of manipulation (e.g., App Store, Play Store, Yelp). We are also eager to further explore how linguistic evolution may provide new insights into the strategies of review manipulation to complement our focus in this paper on the network properties of the reviewers.

Acknowledgement. This work was supported in part by AFOSR grant FA9550-15-1-0149.

REFERENCES

[1] Leman Akoglu, Rishi Chandy, and Christos Faloutsos. 2013. Opinion Fraud Detection in Online Reviews by Network Effects.. In *ICWSM*.
[2] Alex Beutel, Wanhong Xu, Venkatesan Guruswami, Christopher Palow, and Christos Faloutsos. 2013. Copycatch: stopping group attacks by spotting lockstep behavior in social networks. In *WWW*.
[3] Paul Grey. 2015. How Many Products Does Amazon Sell?, https://export-x.com, Last Access: 01/10/2017.
[4] Aditya Grover and Jure Leskovec. 2016. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 855–864.
[5] Stephan Günnemann, Nikou Günnemann, and Christos Faloutsos. 2014. Detecting anomalies in dynamic rating data: A robust probabilistic model for rating evolution. In *SIGKDD*.
[6] Zoltán Gyöngyi, Hector Garcia-Molina, and Jan Pedersen. 2004. Combating web spam with trustrank. In *Proceedings of the Thirtieth international conference on Very large data bases-Volume 30*. VLDB Endowment, 576–587.
[7] Zellig S Harris. 1954. Distributional structure. *Word* 10, 2-3 (1954), 146–162.
[8] Ruining He and Julian McAuley. 2016. Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering. In *Proceedings of the 25th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 507–517.
[9] Bryan Hooi, Neil Shah, Alex Beutel, Stephan Gunneman, Leman Akoglu, Mohit Kumar, Disha Makhija, and Christos Faloutsos. 2015. BIRDNEST: Bayesian Inference for Ratings-Fraud Detection. *arXiv* (2015).
[10] Bryan Hooi, Neil Shah, Alex Beutel, Stephan Günnemann, Leman Akoglu, Mohit Kumar, Disha Makhija, and Christos Faloutsos. 2016. Birdnest: Bayesian inference for ratings-fraud detection. In *Proceedings of the 2016 SIAM International Conference on Data Mining*. SIAM, 495–503.
[11] 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*. ACM, 895–904.
[12] Meng Jiang, Peng Cui, Alex Beutel, Christos Faloutsos, and Shiqiang Yang. 2014. Inferring strange behavior from connectivity pattern in social networks. In *PAKDD*.
[13] Parisa Kaghazgaran, James Caverlee, and Majid Alfifi. 2017. Behavioral Analysis of Review Fraud: Linking Malicious Crowdsourcing to Amazon and Beyond. In *ICWSM*.
[14] 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*. International World Wide Web Conferences Steering Committee, 1063–1072.
[15] Julian McAuley, Christopher Targett, Qinfeng Shi, and Anton Van Den Hengel. 2015. Image-based recommendations on styles and substitutes. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, 43–52.
[16] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*. 3111–3119.
[17] Arjun Mukherjee, Vivek Venkataraman, Bing Liu, and Natalie S Glance. 2013. What yelp fake review filter might be doing?. In *ICWSM*.
[18] Myle Ott, Claire Cardie, and Jeff Hancock. 2012. Estimating the prevalence of deception in online review communities. In *Proceedings of the 21st international conference on World Wide Web*. ACM, 201–210.
[19] Myle Ott, Claire Cardie, and Jeffrey T Hancock. 2013. Negative Deceptive Opinion Spam.. In *HLT-NAACL*.
[20] Myle Ott, Yejin Choi, Claire Cardie, and Jeffrey T Hancock. 2011. Finding deceptive opinion spam by any stretch of the imagination. In *ACL*.
[21] 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 *WWW*.
[22] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. 2014. Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 701–710.
[23] B Aditya Prakash, Ashwin Sridharan, Mukund Seshadri, Sridhar Machiraju, and Christos Faloutsos. 2010. Eigenspokes: Surprising patterns and scalable community chipping in large graphs. In *PAKDD*.
[24] Shebuti Rayana and Leman Akoglu. 2015. Collective opinion spam detection: Bridging review networks and metadata. In *SIGKDD*.
[25] Neil Shah, Alex Beutel, Brian Gallagher, and Christos Faloutsos. 2014. Spotting suspicious link behavior with fbox: An adversarial perspective. In *ICDM*.
[26] Kijung Shin, Bryan Hooi, and Christos Faloutsos. 2016. M-zoom: Fast dense-block detection in tensors with quality guarantees. In *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, 264–280.
[27] Kijung Shin, Bryan Hooi, Jisu Kim, and Christos Faloutsos. 2017. D-cube: Dense-block detection in terabyte-scale tensors. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining*. ACM, 681–689.
[28] Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. 2015. Line: Large-scale information network embedding. In *Proceedings of the 24th International Conference on World Wide Web*. ACM, 1067–1077.
[29] Grigorios Tsoumakas and Ioannis Katakis. 2006. Multi-label classification: An overview. *International Journal of Data Warehousing and Mining* 3, 3 (2006).
[30] Bimal Viswanath, Muhammad Ahmad Bashir, Muhammad Bilal Zafar, Simon Bouget, Saikat Guha, Krishna P Gummadi, Aniket Kate, and Alan Mislove. 2015. Strength in numbers: Robust tamper detection in crowd computations. In *Proceedings of the 2015 ACM on Conference on Online Social Networks*. ACM, 113–124.
[31] Guan Wang, Sihong Xie, Bing Liu, and S Yu Philip. 2011. Review graph based online store review spammer detection. In *ICDM*.
[32] Elizabeth Weise. 2016. Amazon bans 'incentivized' reviews, goo.gl/K8Woqd, Last Access: 01/10/2017. *USATODAY*.
[33] Sihong Xie, Guan Wang, Shuyang Lin, and Philip S Yu. 2012. Review spam detection via temporal pattern discovery. In *SIGKDD*.
[34] Junting Ye and Leman Akoglu. 2015. Discovering opinion spammer groups by network footprints. In *ECML-PKDD*.