MalReG: Detecting and Analyzing Malicious Retweeter Groups

Sonu Gupta
IIIT-Delhi
sonug@iiitd.ac.in

Ponnurangam Kumaraguru
IIIT-Delhi
pk@iiitd.ac.in

Tanmoy Chakraborty
IIIT-Delhi
tanmoy@iiitd.ac.in

ABSTRACT

Given a retweeter network in Twitter for any event, how can we detect the group of users that collude to retweet together maliciously? A large number of retweets of a post often indicates the virality of the post. It also helps increase the visibility and volume of hashtags, topics or URLs, to promote the event associated with it. Our primary hunch is that there is synchronization or indicative pattern in the behavior of such users. In this paper, we propose (i) MalReG, a novel algorithm to detect retweeter groups, and (ii) a set of 23 group-based features (entropy-based and temporal-based) to train a supervised model to identify *malicious retweeter groups* (MRG). We present experiments on three real-world datasets with more than 10 million retweets crawled from Twitter. MalReG identifies 1, 017 retweeter groups present in our dataset. We train a supervised learning model to detect MRG which achieves 0.921 ROC AUC using Random Forest, outperforming the baseline by 7.97% higher AUC. Additionally, we perform geographical location-based and temporal analysis of these groups. Interestingly, we find the presence of the same group, retweeting different political events that took place in different continents at different times. We also discover masquerading techniques used by MRG to evade detection.

CCS CONCEPTS

• **Networks** → **Online social networks**; • **Information systems** → *Social networks*; • **Applied computing**;

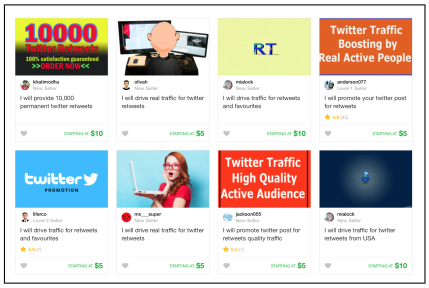
KEYWORDS

Twitter, Malicious Groups, Retweeters, Complex Networks

ACM Reference Format:
Sonu Gupta, Ponnurangam Kumaraguru, and Tanmoy Chakraborty. 2019. MalReG: Detecting and Analyzing Malicious Retweeter Groups. In *6th ACM IKDD CoDS and 24th COMAD (CoDS-COMAD '19), January 3–5, 2019, Kolkata, India.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3297001.3297009

1 INTRODUCTION

Nowadays, Online Social Networks (OSNs) are used not just to connect with people but also to acquire *social currency.*[1] Users explore multifarious opportunities to earn more and more of social currency



khatimodhu
New Seller
I will provide 10,000 permanent twitter retweets
STARTING AT $10

olivah
New Seller
I will drive real traffic for twitter retweets
STARTING AT $5

mialock
New Seller
I will drive traffic for retweets and favourites
STARTING AT $10

anderson077
Level 1 Seller
I will promote your twitter post for retweets
★ 4.8 (42)
STARTING AT $5

liferco
Level 2 Seller
I will drive traffic for retweets and favourites
★ 4.9 (7)
STARTING AT $5

mx___super
New Seller
I will drive real traffic for twitter retweets
STARTING AT $5

jackson055
New Seller
I will promote twitter post for retweets quality traffic
★ 5.0 (1)
STARTING AT $5

mialock
New Seller
I will drive traffic for twitter retweets from USA
STARTING AT $10

Figure 1: Fiverr, an online service to purchase retweets

[1, 16, 25]. For instance, retweet has become one such entity to gain influence on Twitter [14]. There is absolutely no limit to the number of times a post can be retweeted. As of May 2018, the most number of retweets for a tweet is 3.6M.[2] A retweet is now considered as a sign of its social currency. The quality tweet is retweeted numerous times, inferring that the author is a celebrated Twitter user. Consequently, a large number of retweets are associated with high influence and popularity. This has spawned a desire to gain popularity among users which is conducive to the emergence of several paid retweeting services. These paid services deploy both real-users and bots disguised as regular Twitter users which can be purchased at a minuscule price. There are many crowd-sourcing platforms which are involved in hiring retweeters. Figure 1 illustrates one such crowd-sourcing platform, Fiverr, where anyone can easily buy retweets. Since this activity is inorganic, we call such paid users and the bots as *malicious retweeters* and the users whose tweets get retweeted as *clients.* We define a *malicious retweeter group* (MRG) as a group of users that collude to retweet together for the same purpose. The purpose may vary to a large extent. A purpose can be monetary incentives or a political agenda to promote their own party and downgrade opponent's reputation. The members of an MRG together retweet their client's tweet as per the deal between the client and the service provider.

Motivation: The presence of bots in the Twitter network is not a new phenomenon [22]. Bots are often accused of fake activities and polarizing the discussions in the network.[3] It has been observed that such retweeting activity upsurges whenever a news-making incident takes place [15]. In the recent past, we have seen multiple scenarios where OSNs are also accused of influencing the elections.[4] In October 2017, an Indian politician was accused of using bots to increase his retweet count.[5] There are studies to detect the presence of individual fraud retweeters in the network [11, 14].

[1] https://www.entrepreneur.com/article/287702

[2] https://twitter.com/carterjwm/status/849813577770778624/photo/1

[3] http://time.com/5260832/malaysia-election-twitter-bots-social-media/

[4] https://www.cio.com/article/3137513/social-networking/twitters-impact-on-2016-presidential-election-is-unmistakable.html

[5] https://timesofindia.indiatimes.com/india/bots-behind-rise-in-rahul-gandhis-twitter-popularity/articleshow/61161857.cms

CoDS-COMAD '19, January 3–5, 2019, Kolkata, India

ACM ISBN 978-1-4503-6207-8/19/01…$15.00
https://doi.org/10.1145/3297001.3297009

However, the same techniques cannot be applied to detect the malicious retweeter groups as collective behavior is usually more subtle than individual behavior. At the individual level, activities might be normal; however, at the group level, they may substantively differ from other groups. Moreover, it is not possible to understand the actual dynamics of a group by aggregating the behavior of its members due to the complicated and multi-faceted nature of inter-personal dynamics. Besides this, individuals' behavior tend to be interdependent, influenced by the behavior of other members. This calls for designing a separate method for group level fraud detection.

Our Approach: In this paper, we propose MalReG, a novel algorithm to detect retweeter groups present in the Twitter network. Here, we focus on three political events – (i) Delhi Legislative Assembly Election (2013), (ii) Indian Banknote Demonetization (2016), and (iii) UK General Election (2017). We create an undirected weighted retweeter network for each dataset. We apply a well-established community detection algorithm, Louvain [5], on the retweeter network to extract the groups. To improve on the groups obtained from Louvain, we apply MalReG on each extracted group. MalReG is efficient enough to capture the groups that have used several techniques to evade detection. We propose a set of 23 group-based features; both entropy-based and temporal-based, to train a supervised learning model and detect MRG. Our method achieves ROC-AUC of 0.921 using Random Forest classifier, outperforming Attractor+ [28], the only available baseline with 7.97% higher AUC. In our prefatory examination, we notice that each MRG has disparate behavior w.r.t the retweeting time-interval and the frequency. Few MRGs retweet the post as soon as it is created while others wait for some time before retweeting. However, the behavior of the latter demonstrates a deliberate practice of the groups to evade detection. We also observe that not every member of the MRG retweets together. Instead, they often form subsets of the group and retweet together. This indicates the intentional camouflage exercised by the expert MRGs. Giatsoglou et al. [13] observed that despite earlier studies that showed followers-to-followees ratio is a good indicator of fraudulent behavior, it is uninformative for several fraudsters. It also applies to MRGs.

To the best of our knowledge, this is the second attempt after [28] to detect MRGs present in the Twitter network. Our main contributions are four-fold:

- **Methodology:** We propose MalReG, a novel algorithm to detect retweeter groups.
- **Feature engineering:** We investigate the synchronous behavior of the members of MRGs w.r.t tweet content, retweeting activities and temporal properties. By this analysis, we carefully curate 23 group-based features.
- **Classifier:** We train a binary classification model which achieves 0.921 AUC using Random Forest algorithm and beats the baseline significantly.
- **Dataset:** As a by-product of this study, we collected and annotated MRGs from three datasets, which to our knowledge are the first publicly available datasets of this kind.

Reproducibility: The anonymized and labeled dataset of the retweeter groups and as well as the codes are available at https://tinyurl.com/ybgdkz2r.

2 RELATED WORK

Various works have addressed the detection of individual fraudsters in the Twitter network [16, 24]. Here we discuss anomaly detection in Twitter and assess both individual and group retweeting fraud detection techniques.

Anomaly Detection: Das et al. [9] used local anomaly detectors to discern records with anomalous values. Yu et al. [30] performed a survey on OSM anomaly detection. Chan and Mahoney [7] introduced two algorithms to create models from multiple time series for anomaly detection. The models produced anomaly scores for real-life tracking of the tasks. Yu et al. [29] studied group anomaly detection. NetProbe spots anomalies and online auction fraud by applying belief propagation [23]. It can also predict which users might perform frauds in the future. Akoglu et al. [2] proposed Oddball to exploit ego-nets to spot numerous patterns in a weighted graph. Beutel et al. [4] proposed CopyCatch, to find suspicious lockstep behavior. In a similar work, Jiang et al. [18] studied who-follows-whom networks. Shah et al. [26] proposed fBox, an adversarial strategy to catch suspicious entities in the large online networks that suffer from link fraud. Mao et al. [21] proposed MalSpot that apply multi-linear probation with disparate time resolutions to detect malicious network patterns. Com2 [3] uses an incremental tensor analysis technique to identify ephemeral and recurrent communities. Jiang et al. [17] proposed CatchSync that utilizes synchronized and rare behavior to find fraudulent patterns. Giatsoglou et al. [13] curated features to detect synchronous frauds. The proposed algorithm is called ND-Sync which is utilized to detect anomalies. Kaminska et al. [19] studied the use of automated means to disseminate enormous amount of misinformation about politics over OSNs. Liu et al. [20] proposed contrast suspiciousness metric to detect fraudulent users. However, there are only a few studies that analyze the retweeting patterns in the Twitter network.

Retweet Fraud Detection: *Retweet* is a powerful function to propagate information rapidly across OSNs. It has been observed that cyber-criminals exploit this function to achieve their malignant goals. Such incidents often occur during news-making events like elections [15]. However, there are only a few works that deal with retweet fraud detection or retweet fraudsters. [12] identified five discrete classes of retweeting activity on Twitter using the entropy of the time interval distribution and the user. Cherepnalkoski and Mozetic [8] studied the retweet network of users that belong to the European Parliament and detected groups of influence when the ground-truth is known. Giatsoglou et al. [14] identified some triangular and homogeneity related patterns to detect fake retweets. They developed RTGen that generates both inorganic and organic retweets by applying the weighted cascade model. Recently, Dutta et al. [11] designed ScoRe, a supervised method to detect collusive retweeters. However, all these works attempted to detect individual fraud activities, not the group-level activities.

Group Retweet Fraud Detection: Here we focus on identifying *malicious retweeter groups*. There is not much research done to find such retweeter groups. Our work is inspired by the only study done by Vo et al. [28] to address the same problem. They proposed Attractor+ to detect retweeter groups such that the retweeting behavior of each member is similar. In our work, we propose a novel algorithm called MalReG to extract and prune the retweeter groups.

Besides, the technique we use for pruning the groups is different and efficient from the one devised in [28]. They decomposed a component on the basis of pair-wise screen name similarity. However, in our preliminary study, we found that a significant fraction of retweeter groups do not have similar screen names. Therefore, decomposing on the basis of just screen names might not be a good practice. MalReG is robust enough to uncover the groups that exhibit various complex retweeting patterns.

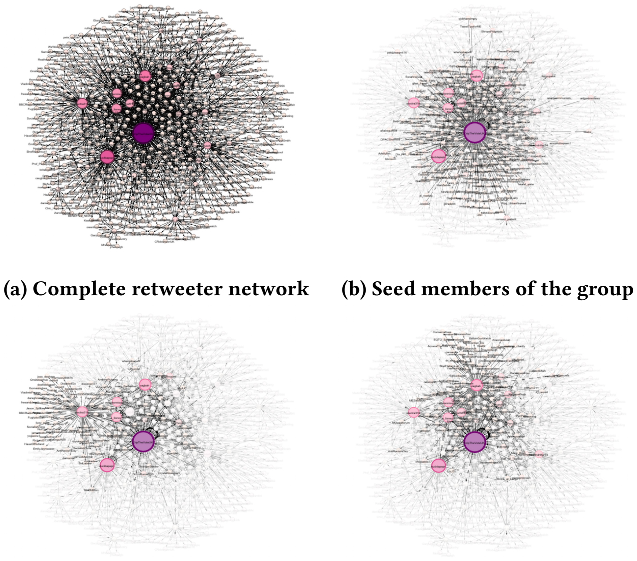
3 PROBLEM DEFINITION

Our goal is to find *malicious retweeter groups* in Twitter network.
Given: *A set of users $\mathcal{U}$ of size $\mathcal{N}$.*
Identify: *Groups of users involved in collective retweeting activities.*
To determine: *Whether the retweeter group is malicious or benign.*

During our preliminary analysis, we discover that often retweeter groups have two types of members. The first type of the members are the most frequent retweeters. They form the core-part of the group. The second type of retweeters are the less-active members. The other significant difference lies in the users whose tweet they retweet. To understand it, let us assume a user A and a retweeter group $\mathcal{G}$ that has 20 members – $m_1, m_2, m_3, .., m_{20}$. User A may or may not be a part of $\mathcal{G}$. Let us consider a scenario where user A posts a tweet t_1. Five members, $m_1, m_2, .., m_5$ retweet t_1 that we call $rt_1, rt_2, ..., rt_5$. The remaining 15 members form four subgroups, sg_1, sg_2, sg_3, sg_4 of size 3, 4, 5, and 3 respectively, and sg_1 retweets rt_1, sg_2 retweets rt_2, and sg_3 retweets rt_3. Now, sg_4 retweets one of the retweets by a member of sg_2. In this way, not all the members directly retweet t_1 but form a sort of retweeting cascade. Again, user A tweets t_2, and like the previous case, $m_1, m_2, .., m_5$ retweet it. However, unlike before, not all the remaining members form identical subgroups or some members may not even retweet. Therefore, we call members $m_1, m_2, .., m_5$ as *seed members* of the group and the remaining users as *guest members*, as they retweet occasionally. We can see this behavior in the Figure 2. The network represents one the retweeter groups from our UK dataset. Figure 2a shows the complete retweeter network, Figure 2b shows the seed members, while Figures 2c and 2d show some guest users that are connected to primary source user via few seed members. *Most the existing algorithms fail to capture this behavior of a retweeter group. They focus on the detection of only seed members of the group. The beauty of our algorithm, MalReG, lies in its capability to detect the seed members along with the guest members that lends a few extra hands to the seed members.* In this paper, we focus on detecting the *malicious retweeter groups* using the features based on the network connections, the temporal behavior of retweeters, the content of the retweet, and the retweeter's profile that distinguish them from the *benign retweeter groups* (BRG).

4 METHODOLOGY

In this Section, we describe our proposed methodology to detect the presence of MRGs in the Twitter network. *An MRG can be defined as a set of real users or bots that collude together to retweet either a post of a Twitter user or all the posts related to a particular event.* For this study, we aim to find MRGs of size at least 3. To extract MRGs from the network, we propose an algorithm called MalReG. It follows a 5-step process: (i) create an undirected weighted network, (ii) extract candidate groups, (iii) decompose the candidate group on the basis of connectedness and common retweets, (iv) extract features from each retweeter group, and (v) train a supervised model to classify it as malicious or benign.


(a) Complete retweeter network **(b) Seed members of the group**

(c) Set of guest members (I) **(d) Set of guest members (II)**

Figure 2: Network graphs of a retweeter group extracted from our UK dataset.

4.1 Creating an Undirected Weighted Network

For each user u_i, $RT(u_i)$ denotes the set of all the retweets of u_i that belong to set U. We calculate weight w_{ij} of each pair of u_i and u_j such that w_{ij} is the number of common tweets of u_i and u_j i.e., $w_{ij} = \mid RT(u_i) \bigcap RT(u_j) \mid$. We create an undirected weighted network $G(V, E)$ such that V is the set of users and E is the set of edges, $E = \{(u_i, u_j) \mid u_i, u_j \in V, w_{ij} > r\}$, where r is the threshold (set as 3 in this case).

4.2 Extracting and Pruning Retweeter Groups

Here, we discuss our technique to extract the groups and then prune them using the steps presented in Algorithm 1.

Extraction: In order to extract groups from an undirected weighted graph, $G(V, E)$, we apply the Louvain community detection algorithm [5]. The Louvain algorithm is a heuristic method that is based on modularity optimization. It is extremely fast, and it outperforms many existing baselines [6]. On manual inspection of the extracted group, we find that the Louvain gives coarse grained groups. As a result, many false positives are also present within the group. This calls for pruning of the groups to remove the false positives. Therefore, the extracted groups form an initial set of candidate groups (ζ). From now onwards, we consider one candidate group at a time and prune it. We repeat the entire procedure for all the candidate groups one by one. To prune a candidate group, we need to create an undirected weighted retweeter network. Note that, here we create $G_i(V, E)$ for a candidate group only, and not for the entire network. To create a $G_i(V, E)$, first we generate a bipartite graph, $\mathcal{BG}(v', e')$, for a candidate group such that one set of nodes represent users and another set represents all the retweets of those users. We place an edge from a user node to retweet node if u_i has retweeted t_i. We apply one mode projection on the user set of $\mathcal{BG}(v', e')$. It results in an undirected weighted retweeter network for a candidate group. Now, our candidate group is ready for pruning. But before that, we drop all the edges from $G_i(V, E)$ that has weight less than r (set to 3). By removing the edge, we intuitively break the bond of an inactive user from the entire group.

Pruning: For pruning, our input is $G_i(V, E)$. The first step is to check if $G_i(V, E)$ is a connected component or not. If there are some disjoint nodes, intuitively, such nodes cannot be part of the same group. Therefore, we further decompose $G_i(V, E)$ into n-subgraphs where n is the number of connected components present in it. Now, each component is considered as a separate candidate group. To further prune the candidate groups, we find the maximal cliques ($\mathcal{MC}$) from each group such that the intersection of the nodes of all the maximal cliques is null. It means that there should be no overlap between the nodes of two maximal cliques. The idea behind it is to obtain all the close-knit users together as $\mathcal{MC}$. We compute the frequency of retweets for each $\mathcal{MC}$ and sort them in descending order of frequency of retweets. We discover that in some cases, there is a huge difference between the retweeting frequency of $\mathcal{MC}$. As expected, this is a result of cascading retweeting behavior mentioned in Section 3. In order to capture this behavior, we compute a threshold, α which denotes the point from where there is a drastic fall in retweeting frequency. We divide the set of $\mathcal{MC}$ by α, such that all $\mathcal{MC}$ with the number of retweets greater than α form seed groups, and all the remaining $\mathcal{MC}$ are combined to form a set of candidate nodes, $\mathcal{CN}$. Let us understand this with the help of an example. Let mc_1, mc_2,...,mc_n be a set of $\mathcal{MC}$ with f_1, f_2,...,f_n be their respective retweeting frequency sorted in descending order. α is the threshold here, and it divides $\mathcal{MC}$ into two parts. Let f_1, f_2,...f_q be greater than α. Therefore, mc_1, mc_2,...,mc_q would form a set of q seed groups and members of mc_{q+1}, mc_{q+2},...,mc_n are combined to form a set of $\mathcal{CN}$.

Now, we compute the number of common retweets between a candidate node and a seed group. We perform this computation for all the candidate nodes and seed groups. We add a candidate node to the seed group with which it has maximum common retweets. Hence, we add all the candidate nodes to the appropriate seed groups. The resultant seed groups are the final retweeter groups. From the algorithm, it is evident that we perform a fine-grained detection by improving upon the results of the Louvain algorithm and are able to obtain several retweeter groups from a single candidate group. We apply this entire procedure on all the candidate groups and get a set of retweeter groups as a result. Further, it is important to remember that the threshold α is different for each candidate group. However, if a candidate group consists of n members such that all the members have the same retweeting frequency, then α would be 0, and the entire candidate group becomes a retweeter group.

4.3 Feature Selection

In order to detect MRGs, we identify two types of group-based features. Table 1 gives a list of all the features which we use for our experiments. We curate a set of 23 group-based features.

Algorithm 1: Pruning Algorithm

```
Input  : Candidate Groups, $\zeta$
Output : Retweeter Groups $\mathcal{G}(v, e)$
foreach $c \in \zeta$ do
    Generate a bipartite graph, $\mathcal{BG}(v', e')$
                        ▷ Set1 ⟵ users, Set2 ⟵ retweets
    One mode projection of $\mathcal{BG}(v', e')$ on the user set
        ▷ Computes weighted undirected retweeter graph
    Find all the connected components, $CC(u)$
    if size of $CC(u) < \tau$ then
        Discard $CC(u)$            ▷ CC of size less than 3
    end
    foreach $\omega \in CC(u)$ do
        Extract maximal cliques, $\mathcal{MC}$
        foreach $\mu \in \mathcal{MC}$ do
            Compute drastic fall of $\alpha$ values     ▷ steepest
              slope
            $\mathcal{SG}(v'', e'')$, $\mathcal{CN} = \mathcal{MC}(\alpha)$
            foreach $k \in \mathcal{CN}$ do
                $\mathcal{G}(v, e)$ = GetRetweeterGroup($\mathcal{SG}(v'', e'')$, $k$)
                return $\mathcal{G}(v, e)$
            end
        end
    end
end
```

Procedure GetRetweeterGroup($\mathcal{SG}$, k)

```
Input  : $\mathcal{SG}(v'', e'')$, $k$
Output : Updated $\mathcal{SG}(v, e)$
if $k \geq \tau$ then
    foreach $\delta \in \mathcal{SG}(v'', e'')$ do
        $\mathcal{I}$ = | $rt(\delta) \bigcap rt(k)$ |        ▷ finding common RTs
    end
    Add $k$ to $\delta$ with max $\mathcal{I}$
    return $\delta(v, e)$
end
```

Entropy-based features: In [12], entropy-based features were reported to be a good measure to classify retweeting behavior. Here, our goal is to look for synchronous behavior of such groups. The lower the entropy more is the synchronicity in the group behavior. Thus, we carefully curate a set of 15 entropy-based group features to identify malicious retweeting behavior. Let X_a be a feature, e.g., favorites count and $\{x_1, x_2, x_3, ..., x_j, ..., x_m\}$ are the favorites count of each member of $\mathcal{G}_i$. If there are n occurrences of x_j, then $p(x_j)$ denotes the probability of observing x_j:

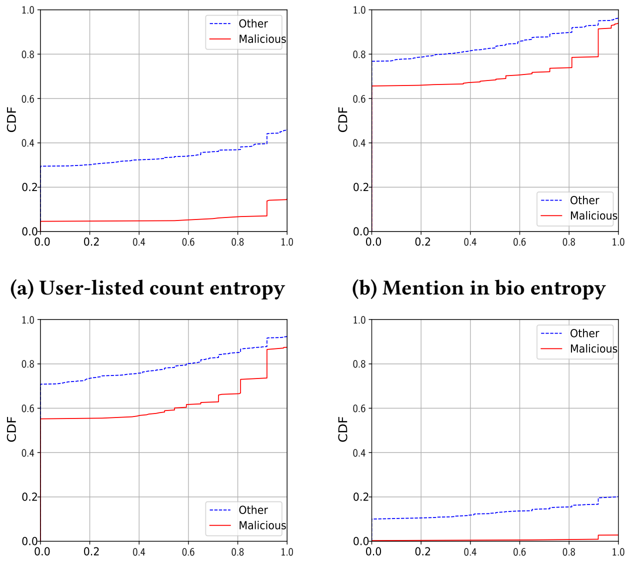
$$p(x_\mathrm{j}) = \frac{n}{m} \tag{1}$$

where m is the number of members in a group $\mathcal{G}_i$. The entropy $\mathcal{H}_i$ of the distribution of the favorites count of $\mathcal{G}_i$ is:

$$\mathcal{H}_i(X_a) = -\sum_{j=1}^{m} p(x_j) \log p(x_j) \tag{2}$$

Table 1: Features used for supervised learning experiments. We extracted features of two types, viz. Entropy-based, and Temporal features.

Source	Feature
Entropy based	retweeters (favorite count, listed count, status count), digits in screen names, hashtags in username, eccentricity, average degree connectivity, average neighbour degree, number of special characters in (screen names, username), no. of URLs in bio, no. of mentions in bio, no. of hashtags in bio, screen name length, username length
Temporal based	inter posting time compactness, retweeting time distribution (standard deviation, mean, coefficient of variance), coefficient of variance of response times, User creation time distribution (standard deviation, mean, and coefficient of variance)



(a) User-listed count entropy

(b) Mention in bio entropy

(c) Hashtags in bio entropy

(d) User favorites count entropy

Figure 3: Cumulative distribution function of the entropy-based features for malicious and other groups.

Similarly, we calculate the entropy of other group-based features as mentioned in Table 1. The entropy of MRG is always less than that of benign groups. As we can see in Figure (3a), the entropy of the user-listed count for MRG groups is quite low. It has been observed that often such malicious accounts are created during any high-impact event; thus the number of lists of which it is part of would be eventually less, decreasing the overall entropy of the group [15]. The same pattern can be seen in the Figure (3b-3d), the overall entropy of the MRGs is low in comparison with the other groups in the network.

Temporal features: Here we discuss the temporal features that we use to train our learning algorithm. Vo et al. [28] showed that the temporal features are a good indicator of malicious group behavior. Therefore, we use 8 temporal features to train our model as explained below.

(a) Inter-posting time compactness (ITC): For each retweeter $u_i \in \mathcal{G}$, we create a list of time when a post was retweeted by u_i. We merge all the lists $T = [t_1, t_2, t_3, ..., t_q]$ and sort it to calculate inter-posting time Δ of T (e.g., $t_2 - t_1, t_3 - t_2, ..., t_q - t_{q-1}$). Then we pair each consecutive inter-posting time and logarithmically bin into a grid in two-dimensional space. In order to make a grid, we calculate the number of vertical lines and the horizontal lines and then use the maximum value to create a grid $\mathbf{M}^{s \times s}$. To find the value of s, for each pair of coordinates $(\log_2 \Delta_i, \log_2 \Delta_{i+1})$, we round it off to nearest integer. Let $\mathrm{int}(\log_2 \Delta_i)$ be f and $\mathrm{int}(\log_2 \Delta_{i+1})$ be g such that $F = [f_1, f_2, f_3, ..., f_{q-1}]$ and $G = [g_1, g_2, g_3, ..., g_{q-1}]$. We compute the value of f_{max} and f_{min}. Similarly, we determine g_{max} and g_{min}. The maximum of $f_{max} - f_{min} + 1$ and $g_{max} - g_{min} + 1$ yields the value of s. We count the number of pairs in each grid cell and save it in $\mathbf{C} \in \mathbf{M}^{s \times s}$. We finally calculate inter-posting time compactness as the maximum of the ratio of number of pair in a cell to sum of pairs in the grid as shown in Equation 3. This is worth mentioning that the value of s changes for every group due the variation in the number of members and the number of retweets by each group.

$$ITC(\mathcal{G}) = \max_{ij} \left(\frac{C_{ij}}{\sum_{i=1,j=1}^{s} C_{ij}} \right) \tag{3}$$

(b) Retweeting time distribution: For each retweeter $u_i \in \mathcal{G}$, we extract a list of retweeting time for each retweet thread[6] and compute the standard deviation of this list. From this list of standard deviations, we measure three features - standard deviation, mean, and coefficient of variance.

(c) Coefficient of variance of response times: For each retweeter $u_i \in \mathcal{G}$, we calculate the median of response time. Response time is the time difference between the actual time when the tweet was posted and the retweeting time. We then calculate the coefficient of variance of this list.

(d) User creation time distribution: In our analysis we observe that within a malicious group, several users are created within a small time span. In order to capture this, we calculate the mean, standard deviation and coefficient of variance of the time difference of the user creation.

5 DATA COLLECTION AND ANNOTATION

We now discuss our technique to collect data and to annotate the retweeter group as malicious or benign. The dataset curation is a three-step process; (i) collecting data from Twitter, (ii) extracting and pruning retweeter group, and (iii) labeling the retweeter group as malicious or benign.

5.1 Data Collection

It has been observed that the cyber-criminal activities increase during the high impact events [15]. Therefore, we select three political events for this study; (1) UK General Election, 2017, (2) Indian

[6] Given a tweet tw_i, a retweet thread is the set of all the retweets of tw_i

banknote demonetization, 2016, and (3) Delhi Legislative Assembly Election, 2013. All these events created quite a buzz globally. Note that this is the first work to use the Twitter data of Indian banknote demonetization (2016) and Delhi Legislative Assembly Election (2013). To collect tweets for each of the event, firstly, we curated a list of trending hashtags and then collected the tweets using Twitter's streaming API. [7]. We filtered out all the tweets that were retweeted less than 3 times. Table 2 shows the descriptive statistics of the Twitter data.

Table 2: Descriptive statistics of Twitter data

Event	# Retweets	# Retweeters
UK General Election	1,459,205	443,913
Indian Banknote Demonetization	2,015,101	288,487
Delhi Legislative Assembly Election	6,800,687	297,793

5.2 Extracting and Pruning Retweeter Groups

As described in Section 4, for each dataset, we create a weighted undirected retweeter network, $G(V, E)$. We extract an initial set of candidate groups by applying the Louvain community detection algorithm. We prune the groups using Algorithm 1. After pruning, we get a set of 1,017 retweeter groups including all the three datasets. The number of members in a retweeter group varies from 3 to 402. However, it is important to mention here, since there is no ground truth available for the retweeter groups we could not evaluate the detected groups.

5.3 Annotating Retweeter Groups

Annotating a retweeter group as benign or malicious requires a rigorous assessment than annotating an individual user. Every group was labeled by exactly three people[8]. We provide Twitter Rules[9] to all the annotators as a reference. The inter-annotator agreement based on Cronbach α for all the 1,017 groups is 0.73. If the value of α is greater than 0.7, it implies high agreement between annotators [27]. Finally, out of 1,017 groups, 690 groups are labeled as malicious and 327 as benign. Table 3 shows the description of retweeter groups for each dataset.

Table 3: Description of retweeter groups

Event	# Groups	# MRG	# BRG
UK General Election	196	89	107
Indian Banknote Demonetization	458	313	145
Delhi Legislative Assembly Election	363	288	75

6 EXPERIMENTAL RESULTS

In this Section, we elaborate the results based on the classification mechanism using the features described in Section 4.3. We also present a baseline algorithm and compare our results with it. All results are reported after 10-fold cross-validation.

6.1 Baseline Algorithm

Vo et al. [28] proposed Attractor+ to extract the retweeter groups from a retweeter graph such that the members of each of group have similar retweeting behavior. It measures various types of interactions such as common interaction, direct linked interaction, and exclusive interaction. It measures pair-wise screen name similarity and the density of a connected component to determine if further decomposition is required. It examines the characteristics of malicious and legitimate retweeter groups and uses group-based features to detect synchronized behavior and to build a model to predict if a group is malicious. However, Attractor+ does not take into consideration the cascading retweeting effects. Moreover, MalReG is efficient enough to detect groups that have used deliberate techniques to evade detection.

6.2 Evaluation Metrics

In order to assess the efficacy of our classification method based on the group-based features, we use two measures – accuracy and ROC-AUC value. Accuracy can be defined as the ratio of the correctly classified elements of either class to the total number of elements. AUC measures the performance of a two-class classifier system as its discrimination threshold is varied.

6.3 Comparative Evaluation

To evaluate MalReG to identify MRGs on Twitter, we use four classification methods viz. Logistic Regression, Random Forest, Gradient Boosting, and Linear Discriminant Analysis. We present the results of the classification task for the above mentioned algorithms. Since our dataset is imbalanced, we downsample to the minority class. The results from the four classification algorithms are described in Table 4. Random Forest turns out to be the best classifier with both types of features. We, therefore, use it as the default classifier in our experiments.

Table 4: Results of individual classifiers with different feature sets.

Classifier	Feature set	Acc. (%)	ROC AUC
Logistic Regression	Entropy	76.22	0.854
	Temporal	78.05	0.857
	Both	71.34	0.775
Random Forest	Entropy	74.39	0.824
	Temporal	81.54	0.884
	Both	82.88	**0.921**
Gradient Boosting	Entropy	76.83	0854
	Temporal	80.49	0.874
	Both	78.05	0.864
Linear Discriminant Analysis	Entropy	73.17	0.843
	Temporal	79.88	0.854
	Both	81.10	0.884

Now, we compare the results of our classifier with the baseline model. Table 5 shows the performance of our best classifier and the baseline. MalReG outperforms the baseline model by 7.97% higher ROC AUC value and 8.45% higher accuracy (relative).

[7] https://developer.twitter.com/en/docs/tweets/filter-realtime/overview

[8] The annotators were experts on social media, and their age ranged between 20 to 30.

[9] https://help.twitter.com/en/rules-and-policies/twitter-rules

Table 5: Performance of MalReG and the baseline [28].

Method	Accuracy (%)	ROC AUC
Attractor+	76.42	0.853
MalReG	82.88	0.921

7 ANALYSIS

In this Section, we discuss several interesting characteristics of MRGs based on geographical location and temporal pattern.

7.1 Geographical Locations

We collect and study the geographical location of each group. We observe that 20.43% of the MRGs have not mentioned the locations in their profiles while for benign retweeter groups (BRG) this behavior is observed in only 3% of the total (Figure 4). Moreover, a significant number of MRG have mentioned phoney locations; e.g., *Follow me! and retweet New's*, *#ButtBounceSec #Bucket*, *MSG Fan*. We also try to ascertain the locations of the MRG in our dataset. We filter out all the phoney locations and plot the remaining ones on a map (Figure 5). Since the political events considered in this work happened in India and the UK, the density of the MRG is higher in these regions. However, we can see that a large count of MRG are indeed from the USA.

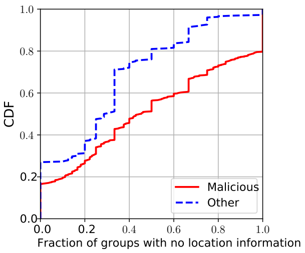


Figure 4: Cumulative distribution function of the geographical location pattern of the retweeter group.

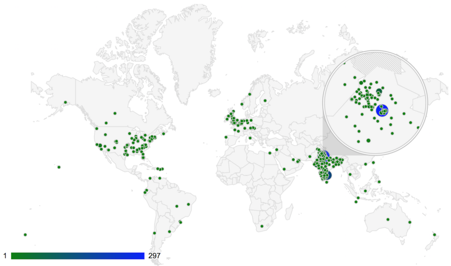
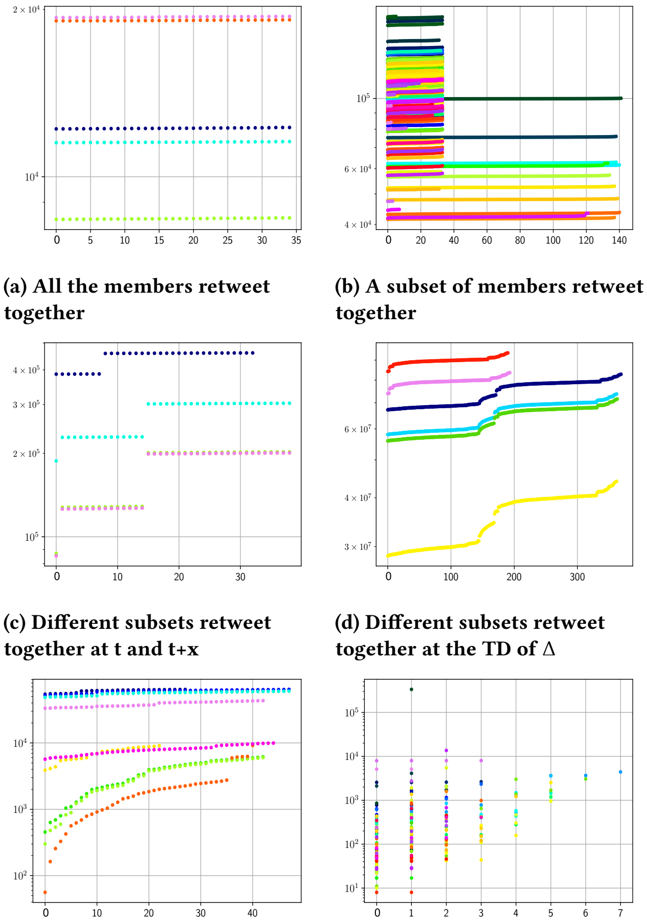


Figure 5: Geographical locations of the malicious retweeters.

7.2 Temporal Analysis

We analyze the temporal behavior of retweeter groups and discover some interesting patterns. One of the key observations is that the temporal behavior of each MRG is different. Therefore, we cannot generalize the behavior of all such groups. However, in some cases, there is a pattern within the group itself. For example, all the members of the group might retweet together always (Figure 6a). In our dataset, nearly 3.34% groups fall into this category and are bots (detected by Botometer [10]); while in some instances, a subset of members retweet together at the same time (Figure 6b) or at different time (Figure 6c). However, the subsets may differ in size and members. There are some groups that never retweet together but with a same time difference, Δ (Figure 6d). There are some other complex cases in which the group follow different retweeting technique for different retweet threads (Figure 6e). In contrast with MRG, there are no such visible patterns in the behavior of BRG (Figure 6f). It is worth noticing that in the examples given in Figure 6, the tweets were not retweeted as soon as they were posted. Some MRGs start retweeting even after several days from the time of creation of the post. This is possibly one of the techniques used by MRGs to evade detection. However, this is not the case always. There are some naive MRGs that start retweeting instantly. To analyze this, we take some MRGs and observe their retweeting threads



(a) All the members retweet together

(b) A subset of members retweet together

(c) Different subsets retweet together at t and t+x

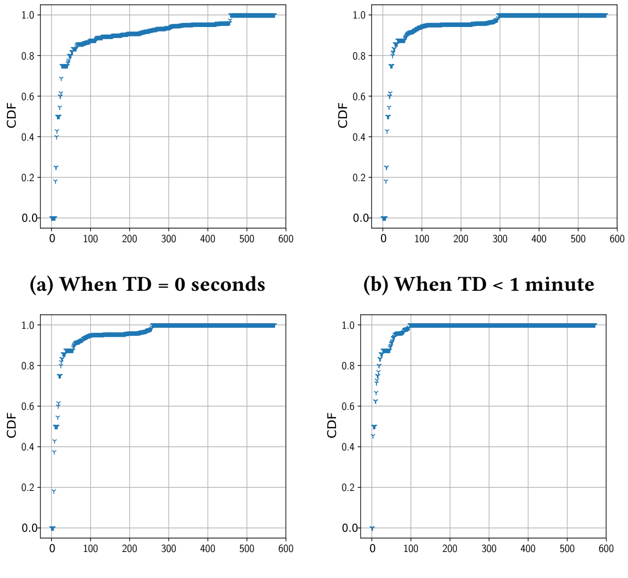
(d) Different subsets retweet together at the TD of Δ

(e) Combination of patterns of figure (6b - 6d)

(f) Random retweeting behavior of a benign retweeter group

Figure 6: Difference in the temporal behavior of the retweeter groups. X axis represents the number of the retweets. Each color indicates a different retweet thread while Y axis is the time difference (in seconds) between actual tweet and retweet. (Best viewed in color)

for two days. Interestingly, for a specific group, 18.3% of retweets are done with no time difference (TD). It is a clear indicator that bots are utilized by MRGs to boost retweet count. We also find that the TD for over 50% retweets is less than a minute (Figure 7) and 98.79% retweeting is done within a day. Such MRGs retweet rapidly but their rate of retweeting drops within an hour. Another interesting observation is the presence of identical MRGs in different political events that took place in different continents at different times. However, not every MRG remains intact over the years. Over 5.5% of MRGs re-form the group for different events. This seems like an intentional step to escape detection.



(a) When TD = 0 seconds

(b) When TD < 1 minute

(c) When TD < 5 minute

(d) When TD < 1 hr

Figure 7: Cumulative distribution of TD between post creation time and retweeting time.

8 KEY FINDINGS

In this Section, we discuss some of the interesting and critical findings of this work. Before that, we would like to mention that here we only report the findings and refrain from mentioning the exact Twitter handles involved in the malicious activities for the obvious reasons.

We studied each MRG and discovered that in the context of the politics, malicious retweeting takes place only for two reasons; to promote one's political affiliations and to downgrade opponents reputation. To our surprise, we found that even the big-shot candidates of the elections with verified Twitter handles use MRG for campaigning. They use not only one but multiple MRGs at a time. We also discover some characteristic differences among MRGs. A set of MRGs are solely created for a political party. Let us call them MRG Type-I (MT-I). A quick analysis of their Twitter timeline reveals that all the retweets are linked to one topic. Additionally, they are often created at the time of the event. During our analysis, we found an MRG of 52 members that were created within 2 days to promote one of the candidates during elections. Whereas there are another set of MRGs that retweet a variety of users and versatile topics. Let us call them MRG Type-II (MT-II). We found another striking difference between MT-I and MT-II. We observed that the members of former often retweet each other whereas the members of the latter never do that. A simple explanation of this behavior is the common purpose of the MT-I. On the other hand, MT-II has no real purpose. They just retweet on the basis of the deal between the client and the service provider and keep retweeting for the new set of clients. Another fascinating finding is the re-use of MT-I. We discovered that a set of MRGs used in 2013 Delhi Election, were re-used in 2015 Delhi Elections. We also spotted that over 7.2% MRGs deleted all the tweets from previous elections and again became active only at the time of the next elections. This could possibly help them evade suspension and maintain their presence on the network for a long time. Also, not only at the time of the elections but even after being elected, politicians use MRGs to maintain their popularity. We found Twitter handles of the Union Ministers being retweeted by bots on a regular basis.

Twitter has a strict user-agreement and suspends the account if found violating any policy. We observed that after some time t many users from MRGs were suspended. They were not suspended at once, but few at a time. This shows that Twitter suspension mechanism works for individual users and not for groups. That is why they are not able to detect such groups and suspend all the members at once. Therefore, there is a need for algorithms like MalReG to address this issue.

9 CONCLUSION

In this paper, we addressed a novel problem of detecting and analyzing malicious retweeter groups present in the Twitter network.We used three political event-based datasets; (i) Delhi Legislative Assembly Election (2013), (ii) Indian Banknote Demonetization (2016), and (iii) UK General Election (2017). This is the first work to study the Twitter dataset of 2013 Delhi Legislative Assembly Election and 2016 Indian Banknote Demonetization. We created undirected weighted retweeter network for each dataset. We applied the Louvain community detection algorithm to extract an initial set of candidate groups. We proposed a novel algorithm, called MalReG, to detect and prune the candidate retweeter groups. We were able to identify 1,017 retweeter groups in our datasets. We proposed a set of 23 features – entropy-based features and temporal features, to identify MRGs from all the identified 1,017 groups. We trained a supervised model to detect MRG which achieved 0.921 AUC using Random Forest. This is 7.97% higher than the baseline. Furthermore, we performed geographical location based and temporal based analysis. We observed that 20.43% MRG have not publicly disclosed their locations. Besides, there are a significant number of MRGs that use phoney locations. In our temporal analysis, we discovered that MRG use multiple techniques to evade detection, e.g., retweeting not immediately after the original tweet was posted, using different subgroups to retweet different tweets, etc. In the future, we would like to model user behavior. Besides, we also intend to study the behavior of such groups across different social media platforms.

ACKNOWLEDGEMENT

We would like to thank Hridoy Sankar Dutta for his comments. This work was partially supported by Indo-U.K. Collaborative Project under Grant (DST/INT/UKP-158/2017) and Early Career Faculty Award (SERB, DST, India).

REFERENCES

[1] Anupama Aggarwal, Saravana Kumar, Kushagra Bhargava, and Ponnurangam Kumaraguru. 2018. The Follower Count Fallacy: Detecting Twitter Users with Manipulated Follower Count. *arXiv preprint arXiv:1802.03625* (2018).

[2] Leman Akoglu, Mary McGlohon, and Christos Faloutsos. 2010. Oddball: Spotting anomalies in weighted graphs. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*. Springer, 410–421.

[3] Miguel Araujo, Spiros Papadimitriou, Stephan Günnemann, Christos Faloutsos, Prithwish Basu, Ananthram Swami, Evangelos E Papalexakis, and Danai Koutra. 2014. Com2: fast automatic discovery of temporal (âĂŸcometâĂŹ) communities. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*. Springer, 271–283.

[4] 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 22nd international conference on World Wide Web*. ACM, 119–130.

[5] Vincent D Blondel, Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. 2008. Fast unfolding of communities in large networks. *Journal of statistical mechanics: theory and experiment* 2008, 10 (2008), P10008.

[6] Tanmoy Chakraborty, Ayushi Dalmia, Animesh Mukherjee, and Niloy Ganguly. 2017. Metrics for Community Analysis: A Survey. *ACM Comput. Surv.* 50, 4, Article 54 (Aug. 2017), 37 pages. https://doi.org/10.1145/3091106

[7] Philip K Chan and Matthew V Mahoney. 2005. Modeling multiple time series for anomaly detection. In *Data Mining, Fifth IEEE International Conference on*. IEEE, 8–pp.

[8] Darko Cherepnalkoski and Igor Mozetic. 2015. A retweet network analysis of the European Parliament. In *Signal-Image Technology & Internet-Based Systems (SITIS), 2015 11th International Conference on*. IEEE, 350–357.

[9] Kaustav Das, Jeff Schneider, and Daniel B Neill. 2008. Anomaly pattern detection in categorical datasets. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 169–176.

[10] Clayton Allen Davis, Onur Varol, Emilio Ferrara, Alessandro Flammini, and Filippo Menczer. 2016. Botornot: A system to evaluate social bots. In *Proceedings of the 25th International Conference Companion on World Wide Web*. International World Wide Web Conferences Steering Committee, 273–274.

[11] Hridoy Sankar Dutta, Aditya Chetan, Brihi Joshi, and Tanmoy Chakraborty. 2018. Retweet Us, We Will Retweet You: Spotting Collusive Retweeters Involved in Blackmarket Services. *CoRR* abs/1806.08979 (2018). arXiv:1806.08979 http://arxiv.org/abs/1806.08979

[12] Rumi Ghosh, Tawan Surachawala, and Kristina Lerman. 2011. Entropy-based classification of’retweeting’activity on twitter. *arXiv preprint arXiv:1106.0346* (2011).

[13] Maria Giatsoglou, Despoina Chatzakou, Neil Shah, Alex Beutel, Christos Faloutsos, and Athena Vakali. 2015. Nd-sync: Detecting synchronized fraud activities. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*. Springer, 201–214.

[14] Maria Giatsoglou, Despoina Chatzakou, Neil Shah, Christos Faloutsos, and Athena Vakali. 2015. Retweeting activity on twitter: Signs of deception. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*. Springer, 122–134.

[15] Aditi Gupta and Ponnurangam Kumaraguru. 2012. Credibility ranking of tweets during high impact events. In *Proceedings of the 1st workshop on privacy and security in online social media*. ACM, 2.

[16] Aditi Gupta, Hemank Lamba, Ponnurangam Kumaraguru, and Anupam Joshi. 2013. Faking sandy: characterizing and identifying fake images on twitter during hurricane sandy. In *Proceedings of the 22nd international conference on World Wide Web*. ACM, 729–736.

[17] Meng Jiang, Peng Cui, Alex Beutel, Christos Faloutsos, and Shiqiang Yang. 2014. Catchsync: catching synchronized behavior in large directed graphs. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 941–950.

[18] Meng Jiang, Peng Cui, Alex Beutel, Christos Faloutsos, and Shiqiang Yang. 2014. Inferring strange behavior from connectivity pattern in social networks. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*. Springer, 126–138.

[19] Monica Kaminska, Bence Kollanyi, and Philip N Howard. [n. d.]. Junk News and Bots during the 2017 UK General Election: What Are UK Voters Sharing Over Twitter? ([n. d.]).

[20] Shenghua Liu, Bryan Hooi, and Christos Faloutsos. 2017. HoloScope: Topology-and-Spike Aware Fraud Detection. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*. ACM, 1539–1548.

[21] Hing-Hao Mao, Chung-Jung Wu, Evangelos E Papalexakis, Christos Faloutsos, Kuo-Chen Lee, and Tien-Cheu Kao. 2014. MalSpot: Multi 2 malicious network behavior patterns analysis. In *Pacific-Asia Conference on Knowledge Discovery and Data Mining*. Springer, 1–14.

[22] Amanda Minnich, Nikan Chavoshi, Danai Koutra, and Abdullah Mueen. 2017. BotWalk: Efficient adaptive exploration of Twitter bot networks. In *Proceedings of the 2017 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining 2017*. ACM, 467–474.

[23] 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*. ACM, 201–210.

[24] Charles Perez, Marc Lemercier, Babiga Birregah, and Alain Corpel. 2011. Spot 1.0: Scoring suspicious profiles on twitter. In *Advances in Social Networks Analysis and Mining (ASONAM), 2011 International Conference on*. IEEE, 377–381.

[25] Indira Sen, Anupama Aggarwal, Shiven Mian, Siddharth Singh, Ponnurangam Kumaraguru, and Anwitaman Datta. 2018. Worth its Weight in Likes: Towards Detecting Fake Likes on Instagram. In *Proceedings of the 10th ACM Conference on Web Science*. ACM, 205–209.

[26] Neil Shah, Alex Beutel, Brian Gallagher, and Christos Faloutsos. 2014. Spotting suspicious link behavior with fbox: An adversarial perspective. In *Data Mining (ICDM), 2014 IEEE International Conference on*. IEEE, 959–964.

[27] Keith S Taber. 2017. The use of cronbachâĂŹs alpha when developing and reporting research instruments in science education. *Research in Science Education* (2017), 1–24.

[28] Nguyen Vo, Kyumin Lee, Cheng Cao, Thanh Tran, and Hongkyu Choi. 2017. Revealing and detecting malicious retweeter groups. In *Proceedings of the 2017 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining 2017*. ACM, 363–368.

[29] Rose Yu, Xinran He, and Yan Liu. 2015. Glad: group anomaly detection in social media analysis. *ACM Transactions on Knowledge Discovery from Data (TKDD)* 10, 2 (2015), 18.

[30] Rose Yu, Huida Qiu, Zhen Wen, ChingYung Lin, and Yan Liu. 2016. A survey on social media anomaly detection. *ACM SIGKDD Explorations Newsletter* 18, 1 (2016), 1–14.