Collective Classification of Spam Campaigners on Twitter: A Hierarchical Meta-Path Based Approach

Srishti Gupta  
IIIT-Delhi  
srishtig@iiitd.ac.in

Abhinav Khattar  
IIIT-Delhi  
abhinav15120@iiitd.ac.in

Arpit Gogia  
DTU  
aarpitgogia@gmail.com

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

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

ABSTRACT

Cybercriminals have leveraged the popularity of a large user base available on Online Social Networks (OSNs) to spread spam campaigns by propagating phishing URLs, attaching malicious contents, etc. However, another kind of spam attacks using phone numbers has recently become prevalent on OSNs, where spammers advertise phone numbers to attract users’ attention and convince them to make a call to these phone numbers. The dynamics of phone number based spam is different from URL-based spam due to an inherent trust associated with a phone number. While previous work has proposed strategies to mitigate URL-based spam attacks, phone number based spam attacks have received less attention.

In this paper, we aim to detect spammers that use phone numbers to promote campaigns on Twitter. To this end, we collected information (tweets, user meta-data, etc.) about 3,370 campaigns spread by 670,251 users. We model the Twitter dataset as a heterogeneous network by leveraging various interconnections between different types of nodes present in the dataset. In particular, we make the following contributions – (i) We propose a simple yet effective metric, called Hierarchical Meta-Path Score (HMPS) to measure the proximity of an unknown user to the other known pool of spammers. (ii) We design a feedback-based active learning strategy and show that it significantly outperforms three state-of-the-art baselines for the task of spam detection. Our method achieves 6.9% and 67.3% higher F1-score and AUC, respectively compared to the best baseline method. (iii) To overcome the problem of less training instances for supervised learning, we show that our proposed feedback strategy achieves 25.6% and 46% higher F1-score and AUC respectively than other oversampling strategies. Finally, we perform a case study to show how our method is capable of detecting those users as spammers who have not been suspended by Twitter (and other baselines) yet.

CCS CONCEPTS

• Information systems  • Security and privacy  • Applied computing

1 INTRODUCTION

Online Social Networks (OSNs) are becoming more and more popular in the recent years, used by millions of users. As a result, OSNs are being abused by spam campaigners to carry out phishing and spam attacks [18]. While attacks carried using URLs [8, 16, 18, 42, 47] has been extensively explored in the literature, attacks via a new action token, i.e., a phone number is mostly unexplored. Traditionally, spammers have been exploiting telephony system in carrying out social engineering attacks either by calling victims or sending SMS [45]. Recently, spammers have started abusing OSNs where they float phone numbers controlled by them. Besides exploiting trust associated with a phone number, spammers save efforts in reaching out their victims themselves.

Present Work: Problem definition. In this paper, we aim to detect spam campaigners (aka, spammers) spreading spam campaigns using phone numbers on Twitter. We here define spammers as user accounts that use phone numbers to aggressively promote products, disseminate pornography, entice victims for lotteries and discounts, or simply mislead victims by making false promises. Discovering the correspondence between the spammer accounts and the resources (such as URL or phone number) used for spam activities is a crucial task. As the phone numbers are being propagated by spammers, and their monetization revenue starts once people call them, it is fair to assume that these phone numbers would be under their control. As an added advantage of this, if we can identify the spammer accounts in Twitter and bring them down, the entire campaign would get disintegrated. To identify spammers, we model the Twitter dataset as a heterogeneous graph where there are different connections between heterogeneous type of entities: users, campaigns, and the action tokens (phone number or URL) as shown in Figure 1. Heterogeneous networks have been proposed for data representation in a variety of datasets like path similarity in scholar data [39], link prediction in social network data [22], etc. Objects of different types and links carry different
we introduce a meta-path to distinguish the semantics among paths connecting two nodes, from a pool of meta-paths linking various heterogeneous nodes. We then systematically extract most relevant meta-paths from the large heterogeneous network. We design various experimental setups to perform a thorough evaluation of our proposed method. We observe that our model outperforms the best baseline method by achieving 44.8%, 16.7%, 6.9%, 67.3% higher performance in terms of accuracy, precision, F1-score and AUC (Section 5.3). We further demonstrate how why one-class classifier (Section 5.4), active learning (Section 5.5) and feedback-based learning (Section 5.6) are better than 2-class classifier, general learning and other oversampling methods, respectively. Moreover, we conduct a case study and present an intuitive justification why our method is superior to the other methods (Section 5.3).

Present Work: Motivation of the work. The problem of identifying spammers on Twitter that use phone numbers is useful in many aspects. Attacks using phone numbers and URLs are different in some aspects: in URL based spam, the campaign propagates and spreads on the same medium, i.e., OSNs, while in case of phone number based spam, the attacking medium is a telephone and the propagating medium is OSNs. As a result, it is challenging for OSN service providers to track down the accounts spreading these spam campaigns. In addition, there is no meta-data available for phone numbers, unlike URLs where landing page information, length of URLs, obfuscation, etc. can be checked. Perhaps due to the challenges associated with finding spam phone numbers, there have been several attacks and financial losses caused by the phone-based attacks [31]. Using the collective classification approach proposed in the paper, Twitter will be able to find potential spammers and suspend the accounts, thereby restricting phone number based spam campaigns.

Present Work: A collective classification approach for detecting spam campaigns. In this work, we use the collective classification approach that exploits the dependencies of a group of linked nodes where some class labels are known and labels diffuse to other unknown nodes in the network. In our case, the known nodes are the already suspended Twitter users that were propagating campaigns with phone numbers. Here, we propose Hierarchical Meta-Path Score (HMPS), a simple yet effective similarity measure between a pair of nodes in the heterogeneous network. We first build campaign-specific hierarchical trees from the large heterogeneous network. We then systematically extract most relevant meta-paths from a pool of meta-paths linking various heterogeneous nodes.

We collected tweets and other meta-data information of users from April-October, 2016, and identified 3,370 campaigns, containing 670,251 users (Section 2). Each tweet carries a phone number. We consider user accounts suspended by Twitter as ground-truth spammers. However, due to the lack of enough training samples per campaign, we introduce a novel feedback-based active learning mechanism that uses a SVM-based one-class classifier for each campaign. Over multiple iterations, it keeps accumulating evidences from different campaigns to enrich the training set for each campaign. This, in turn, enhances the prediction performance of individual classifiers. The process terminates when there is no chance of finding the label of the unknown users across iterations.

Summary of the evaluation. We compare our model with three state-of-the-art baselines used for spam detection (Section 5.3). We design various experimental setups to perform a thorough evaluation of our proposed method. We observe that our model outperforms the best baseline method by achieving 44.8%, 16.7%, 6.9%, 67.3% higher performance in terms of accuracy, precision, F1-score and AUC (Section 5.3). We further demonstrate how why one-class classifier (Section 5.4), active learning (Section 5.5) and feedback-based learning (Section 5.6) are better than 2-class classifier, general learning and other oversampling methods, respectively. Moreover, we conduct a case study and present an intuitive justification why our method is superior to the other methods (Section 5.3).

2 DATASET

We collected tweets containing phone numbers from Twitter based on an exhaustive list of 400 keywords via Twitter streaming API. We chose Twitter due to easy availability of data. The data was collected from April - October, 2016. Since we intended to detect campaigns around phone numbers, the keywords we chose were specific to phone numbers such as ‘call’, ‘SMS’, ‘WA’, ‘ring’ etc. We accumulated ~ 22 million tweets, each of which containing at least one phone number. The reason behind collecting only tweets containing phone numbers is that they are found to be a stable resource, i.e., spammers use them for a long period due to attached cost. Moreover, the phone numbers are known to help in forming better user communities [9], which is the basis of the approach adopted in this work.

Campaign identification: We define a campaign as a group of similar posts shared by a set of users propagating multiple phone numbers. A phone number could be part of multiple campaigns; however, in this work, we restrict the phone number to be part of a single campaign (since our campaign detection approach is text-based, we want the campaigns to be coherent). Note that, multiple phone numbers could be part of a single campaign. The detailed approach for campaign identification is shown in Figure 2 using a toy example for three phone numbers as described below:

Step 1: Aggregating tweets. For every phone number, we aggregate all the tweets containing that phone number in a set. We do not find a single tweet containing two phone numbers in our dataset. This implies that every phone number $P_i$ has a set of unique tweets represented as $T_1, T_2, T_3, \cdots$. In Figure 2, $P_i$ is associated with $\{T_1, T_2, T_3\}$.
Step 2: Tweets to unigrams. We extract unigrams from tweets. Each tweet $T_i$ is now represented as $\{U_{i1}^1, U_{i2}^1, U_{i3}^1, \ldots\}$. In Figure 2, $T_1$ is represented as $\{U_{11}^1, U_{12}^1, U_{13}^1\}$.

Step 3: Extracting frequent unigram. We aggregate all tweets containing a certain phone number and extract top 30 unigrams that frequently appear in these tweets. This set of unigrams characterizes the document associated with the phone number. In Figure 2, the set $\{U_{11}^2, U_{12}^2\}$ represents the document associated with $P_1$.

Step 4: Selecting relevant tweets. From the set of tweets associated with a certain phone number, we choose those which have at least 5 unigrams common with the set of 30 unigrams representing the phone number. In Figure 2, we only choose $T_1$ and $T_2$ to form document $D_1$ for $P_1$ (note, in this example we only match at least one unigram in each tweet instead of 5 to be qualify as a part of the document).

Step 5: Jaccard similarity to find campaigns. Once we form the document corresponding to a phone number, we use Jaccard coefficient to find similarity between two documents and combine them as part of the same campaign. If the Jaccard coefficient is greater than 0.7 (experimentally calculated, as corresponding Silhouette score is 0.8), the documents are merged and thus the corresponding phone numbers become part of a single campaign. In Figure 2, $D_1$ and $D_2$ are merged together and form campaign $C_1$.

Using this approach, we identify 22,390 campaigns in our dataset. These account for 10,962,350 tweets posted by 670,257 users, containing 26,610 unique phone numbers, and 893,808 URLs. For collective classification to identify campaign-specific spammers, we need to have a set of labeled users. Therefore, we check the user accounts already suspended by Twitter. This process consists of a bulk query to Twitter’s API with the profile ID of the account. Twitter redirects to http://twitter.com/suspended, and returns ‘error 404’ in case the user account is suspended. Since Twitter suspension algorithm can have a delay in suspension, we made this query 6 months after the data collection. We find 5,593 user accounts to be already suspended by Twitter. These accounts are taken later as the training set to perform spam classification (see Section 4). Note that for further analysis, we take campaigns that have at least one suspended user – 3,370 out of 22,390 campaigns (670,251 user accounts) are seen to observe such behavior. We also observe 21% users to be part of multiple campaigns (see Figure 7(b)). Figure 3 shows the word cloud of top 2 campaigns containing the highest number of suspended users – first one is a Spanish campaign (Figure 3(a)) where people requested others to send WhatsApp invitation for receiving adult and pornographic videos. The second campaign (Figure 3(b)) offers people reservations for parties and clubs at discounted rates.

3 HETEROGENEOUS INFORMATION NETWORKS (HIN)

We model the entire Twitter data as a heterogeneous information network (HIN). A HIN is a special type of information network which contains multiple types of users or links between users [39]. Mathematically, it is defined as follows:

**Definition 3.1.** Heterogeneous network. A large number of OSNs are heterogeneous in nature, involving diverse relationship between the nodes. Heterogeneous network is represented as a graph, $G = (V, E, T)$ in which each node $v \in V$ and each link $e \in E$ are associated with their mapping functions:

$$\phi(v) : V \rightarrow T_V$$

$$\phi(e) : E \rightarrow T_E$$

deeply, $T_V \in T$ and $T_E \in T$ denote the sets of users and edge types.

Our heterogeneous network contains different types of nodes such as users, campaign, URLs, and phone numbers; edges connecting two nodes represent different types of relationships (see Figure 1). A user is linked to a campaign by promoting or promoted-by relation; a user is linked to a phone-number by sharing or shared-by relation; a campaign is linked to a phone-number by using or used-by relations. Two users can be connected via different paths viz. user-phone-user, user-url-user, user-phone-url-user, and so on. Formally, these paths are called meta-paths, as defined below.

**Definition 3.2.** Meta-path. A meta-path $\Pi_1, \ldots, k$ is a path defined on the graph of network schema $T_{\Pi_1} = (U, R)$, and is denoted in the form of $U_1 \xrightarrow{R_1} U_2 \xrightarrow{R_2} U_3 \ldots \xrightarrow{R_k} U_{k+1}$ which defines a composite relation $R = R_1 \circ R_2 \circ \ldots \circ R_k$ between type $U_1$ and $U_{k+1}$, where $\circ$ denotes the composition operator on relations. In our context, $U \in \{$user, campaign, phone number, URL$\}$ and $R \in \{$sharing, promoting, using$\}$.

The length of a meta-path $\Pi$ is the number of relations that exist in $\Pi$ – e.g., user-phone-user is a 2-length meta-path between a pair of users, while a 3-length meta-path instance between two users is user-phone-url-user. Figure 4 depicts some example meta-paths in our heterogeneous network. For instance, a user
participating in a campaign and sharing a phone number can be represented by a 2-length meta-path User–Camp–Phno.

![Diagram of a network with nodes and edges]

**Figure 4: Examples of different meta-paths present in the network.**

Given a user-specific meta-path, \(\Pi = U_1, U_2, ..., U_l\), similarity measures can be defined for a pair of users \(x \in U_1\) and \(y \in U_2\) according to the path instances between them following the meta-path. Previous research has shown that including redundant meta-paths (i.e., a smaller meta-path that can be a part of a longer meta-path) in the collective classification may inject noise in the feature space, which can lead to over-fitting [21]. To minimize the risk, it is advisable to extract meta-paths that cannot be further disintegrated to shorter meta-paths. The major challenge in dealing with meta-paths is to find all and only relevant meta-paths. Sun et al. [39] showed that finding all possible meta-paths and picking the most relevant out of them is an NP-hard problem, and therefore many greedy approaches have been proposed to find relevant meta-paths [29]. To the best of our knowledge, this is the first work towards modeling Twitter as a heterogeneous network for spam campaigner detection by extracting relevant meta-paths. Therefore, there is no prior work suggesting possible and relevant meta-paths for our heterogeneous network. To deal with these challenges, we propose a simple yet efficient concept, called **Hierarchical Meta-Path Scores (HMPS)** to find similarity between a pair of users by picking shortest and relevant meta-paths (restricted to length 4) which can be used to calculate similarity between nodes. \(^1\) We also impose an additional constraint on the meta-path selection - we only consider intermediate nodes of type campaign, phone number, or URL when selecting meta-paths between two users.

**4 PROPOSED METHODOLOGY**

In this section, we describe the overall proposed methodology for collectively classifying users as spammers on Twitter (see Figure 5).

**Why collective classification?** Collective classification refers to the combined classification of nodes based on correlations between unknown and known labels [34]. Given the labels of the instances in training set \(Tr \subseteq All\), the task of collective classification in HIN is to infer the labels of the testing set \(Te = All − Tr\). We address collective classification problem using HMPS to find users (unknown labels) that are similar to spammers (known labels). In individual classification, nodes are classified individually without taking into account their interdependencies via the underlying network structure. However, in our heterogeneous networks, nodes are connected by same phone number or URL. Therefore, we employ collective classification approach. It has been shown to achieve better accuracy compared to independent classification [34].

**4.1 Hierarchical Meta-Path Scores (HMPS)**

After identifying campaigns (see Section 2), the next step is to measure HMPS for a user (Algorithm 1) to find the similarity of the user with other known spammers (suspended users). To this end, we propose an additive similarity score for a user with respect to all the spammers in that particular campaign. Although there are several other similarity measures available, they are biased towards underlying network structure and prior information about relevant meta-paths. For instance, PathSim [39] only works for symmetric relations, HeteSim [35] relies on the relevance of a single meta-path. Forward Stagewise Path Generation (FSPG) [29] generates the set of most relevant meta-paths under a given regression model, which is validated by a human expert. However, in the context of Twitter being modeled as a HIN, the relevant meta-paths are not known. Therefore, it is computationally intractable to find the relevance of a meta-path.

This motivates us to propose a novel meta-path based similarity measure, called **Hierarchical Meta-Path Scores (HMPS)** that captures the similarity between two users based on the function of distance through which they can be reached.

**HIN to hierarchical structure:** To measure HMPS, we model the Twitter heterogeneous network in the form of a multi-path hierarchical structure as shown in Figure 6. In this structure, nodes on a meta-path are connected with their Least Common Ancestor (LCA) node. LCA node for users is taken as a phone number or URL, and subsequently, campaign node is taken as the LCA node for a phone number / URL. The purpose of LCA node is to limit the range of operations that can be applied across two related nodes. We choose such a structure because if two users share the same phone number or URL for promoting campaigns, they should be more similar rather than two users who do not share any common phone number or URL but are still part of a single campaign. The intuition behind HMPS is that if two users are strongly connected to each other, the distance between them in the hierarchical structure would be less.

The similarity score between two entities \(x\) and \(y\) is a real number, computed by a function \(F\) of the similarity scores for each meta-path of a set \(\Theta \subseteq \{\Pi_1, \Pi_2, ..., \Pi_p\}\), where \(\phi(x, y | \Pi_i)\) is a similarity score between \(x\) and \(y\) given meta-path \(\Pi_i\), and \(F\) is the maximum similarity score over the `p` meta-paths. Then the HMPS of an entity \(x\) is defined as:

\[HMPS(x) = \sum_{y \in S} \Psi(x, y),\]

where \(S\) is the set of spammers in the campaign where \(x\) belongs to.
For every user, HMPS is calculated with respect to each spammer (suspended user) in the campaign, and the scores are finally added, as shown in Algorithm 1. Following are the weights used for each edge in the hierarchical structure.

- \( W(\text{User}_i, \text{Phone}_j) \): This is the weight of the edge connecting a user and a phone number, and is measured as the ratio of tweets propagated by \( \text{User}_i \) containing \( \text{Phone}_j \) over all the tweets containing \( \text{Phone}_j \).
- \( W(\text{User}_i, \text{URL}_j) \): This is the weight of the edge connecting a user and a URL, and is measured as the ratio of tweets propagated by \( \text{User}_i \) containing \( \text{URL}_j \) over all the tweets containing \( \text{URL}_j \).
- \( W(\text{Camp}_k, \text{Phone}_j) \): This is the weight of the edge connecting a campaign and a phone number, and is measured as the ratio of tweets containing \( \text{Phone}_j \) in \( \text{Camp}_k \) over cumulative frequency of URLs and phone numbers in \( \text{Camp}_k \).
- \( W(\text{Camp}_k, \text{URL}_j) \): This is the weight of the edge connecting a campaign and a URL, and is measured as the ratio of tweets containing \( \text{URL}_j \) in \( \text{Camp}_k \) over cumulative frequency of URLs and phone numbers in \( \text{Camp}_k \).

Let us assume that we want to calculate the HMPS for \( \text{User}_1 \) (unknown) shown in Figure 6. The campaign contains two suspended users, \( \text{User}_2 \) and \( \text{User}_3 \). So the HMPS score of \( \text{User}_1 \) is calculated w.r.t. \( \text{User}_2 \) and \( \text{User}_3 \) as follows:

- Weight between \( \text{User}_1 \) and \( \text{User}_2 \), \( W(\text{User}_1, \text{Phone}_2) = W(\text{User}_2, \text{Phone}_2) \).
- Weight between \( \text{User}_1 \) and \( \text{User}_3 \), \( W(\text{User}_1, \text{Phone}_3) = W(\text{User}_3, \text{Phone}_3) \).
- Weight between \( \text{User}_1 \) and \( \text{User}_4 \), \( W(\text{User}_1, \text{URL}_4) = W(\text{User}_4, \text{URL}_4) \).

The final HMPS of \( \text{User}_1 \), \( \text{HMPS}(\text{User}_1) = W_1 + W_2 \).

Note that in order to measure the HMPS for each user from the hierarchical structure, we build the hierarchical structure for individual campaigns separately instead of combining all the campaigns due to the following two reasons: (i) it is computationally expensive to find meta-paths for all the connections of users across campaigns from a large hierarchical structure, and (ii) HMPS is an absolute value; global HMPS can result in wrong labeling. Specifically, if a spammer (S) has HMPS value \( X \) in campaign \( C_1 \) and other unknown user (U) has same value \( X \) in another campaign \( C_2 \), then U will be wrongly labeled as a spammer. It might not be a spammer based on HMPS calculated within that campaign.

### Algorithm 1 HMPS for Collective Classification

```plaintext
1: for \( \text{Camp}_j \) \in \text{Campaigns} do
2: \( S = \) Set of known spammers in \( \text{Camp}_j \) (m = \(|S|\)); \( U = \) Set of unknown users in \( \text{Camp}_j \); \( n = \) Total number of users in \( \text{Camp}_j \);
3: \( \text{score}_i = \sum_{j=1}^{m} \text{HMPS}(U, S, \text{Camp}_j) \forall i \in [1, n] \)
4: end for
5: procedure HMPS(u, s, camp)
6: \( res = 0 \)
7: for \( i \in \text{Parent}(u) \) do \( \text{Parent}(u) = \) Immediate antecedent of \( u \)
8: for \( j \in \text{Parent}(u) \) do
9: if \( i == j \) then \( \Rightarrow W(s, j) = \) weight of the edge \((s, j)\) in the hierarchical structure
10: if \( W(u, i), W(s, j) > res \) then
11: \( res \leftarrow W(u, i), W(s, j) \)
12: end if
13: else
14: if \( W(u, i), W(s, j), W(i, \text{camp}), W(j, \text{camp}) > res \) then
15: \( res \leftarrow W(u, i), W(s, j), W(i, \text{camp}), W(j, \text{camp}) \)
16: end if
17: end if
18: end for
19: end for
20: return \( res \)
21: end procedure
```

### 4.2 Active Learning with Feedback

As we consider only those campaigns which contain more than one suspended user (spammer), the classes (spammers and non-spammers) present in our dataset would be highly imbalanced. Exiting research has shown that one-class classification (OCC) achieves much better performance than two-class classification if: (i) there is highly imbalanced dataset [33] and the target class is prevalent in the training set, (ii) the unknown instances do not belong to any known class, or (iii) the unknown instances are difficult to be categorized into a known class due to several reasons such as lack of annotators, lack of enough evidences etc. OCC is trained only on the target class (which is spam in our case), and its task is to define a classification boundary around the target class, such that it accepts as many instances as possible from the target class, while it minimizes the chance of accepting the outlier instances. In OCC, since only one side of the boundary can be determined, it is hard to decide from just one-class how tightly the boundary should fit in each of the directions around the data. It is also hard to decide which features should be used to find the best separation of the target and outlier class instances.

**Learning with feedback:** We would like to reiterate that we picked individual campaigns and not the entire dataset together since the HMPS local to a campaign helps in finding similar users better (see Section 4.1). Each campaign is associated with a supervised classifier (one-class classifier in our case). Out of 3,370 campaigns in the dataset that have at least one suspended user, not all campaigns have sufficient training samples to train the models, as shown in Figure 7(a). However, the process of human annotation to enrich the training set can be costly. To reduce the effort of human labeling, one can obtain meaningful shards of evidence from some external sources and incorporate them into the training...
set. For instance, in ensemble learning, one can leverage the output class of unknown objects obtained from one classifier and feed them into the other classifiers. This might be related to active learning, where given a pool of unlabeled data, one can try to select a set of training examples actively to reach a minimum classification error.

Since individual campaigns may not have significant training instances, we propose an active learning approach with feedback to collect cues about unknown users from multiple campaigns to enlarge the individual training set associated with each campaign-specific model. We further notice that campaigns have significant user overlap – 21% users belong to multiple campaigns (see Figure 7(b) for the distribution of overlapping users). Presence of user overlap further motivates us to incorporate the feedback-based model as follows.

![Diagram showing active learning with feedback amongst campaign-specific classifiers.](image)

**Figure 8:** A schematic diagram of active learning with feedback amongst campaign-specific classifiers.

**5 EXPERIMENTAL RESULTS**

In this section, we start by presenting the baseline methods used to compare with our method, followed by a detailed comparative evaluation.

**5.1 Baseline Methods**

We compare our method with three state-of-art methods proposed in the literature for spam detection in general. However, none of them focused on phone number specific spammers whose dynamics are different (as mentioned in Section 2). Since we did not obtain the source code, we implemented the methods on our own. Note that all the baselines originally used 2-class classifiers. However, in this paper, we show the results of the baselines both for one-class and 2-class classifications after suitable hyper-parameter optimization.

**Baseline 1:** We consider the spam detection method proposed by Benevenuto et al. [2] as our first baseline. They proposed the following OSN-based features (referred as OSN1) per user: fraction of tweets with URLs, age of the user account, average number of URLs per tweet, fraction of followers per followee, fraction of tweets the user replied, number of tweets the user replied, number of tweets the user receives a reply, number of friends and followers,
average number of hashtags per tweet. They showed that the SVM-based classifier performs best.

**Baseline 2:** We consider the method proposed by Khan et al. [19] to segregate spammers from genuine experts on Twitter as our second baseline. They suggested the following features (referred as OSN2): authority and hub scores of users in the follower-followee network, fraction of the user’s tweets that contain the URLs, average number of URLs in a tweet, average number of URLs per number of words in a tweet of the user, average number of hashtags per number of words in a tweet, and average number of hashtags in a tweet. They showed that Logistic Regression performs best.

**Baseline 3:** We consider the method proposed by Adewole et al. [1] to detect spam messages and spam user accounts as our third baseline. They proposed the following list of profile and content-based features (referred as OSN3): length of the screen name based on characters, the presence or absence of profile location, whether the user includes URL or not in his profile, age of the account in days, number of followers of the user, number of friends/followers of the user, total statuses of the account, number of tweets the user has favorited, indicating presence or absence of profile description, whether the user has not modified the default profile egg avatar, number of the public lists the user is a member of, whether or not the user has enabled the possibility of geo-tagging their tweets, normalized ratio of followers to friends, ratio of the number of follower to friends, ratio of the number of friends to followers, (total, unique, and mean) number of tweets, hashtags, URLs, mentions, favorite count, and retweets, ratio of (hashtags, URLs, mentions, retweets) to total number of tweets, (hashtag, URLs, mention, retweet, tweet-length) deviation, average number of daily tweets, average tweet length, popularity ratio, number of duplicate tweets, and maximum value of hashtag frequency. They showed that Random Forest performs best for the classification task.

Note that previous work considered only those campaigns which involve only URLs [1, 2, 19]. In our work, a phone number, being a stable resource, helped in forming campaigns better. Besides, most of the OSN features used in the baselines are easy to evade by spammers, whereas HMPS-based feature is difficult to manipulate.

### 5.2 Experimental Setup

Our proposed classification method is run separately with different features (HMPS, OSN1, OSN2, and OSN3) and their combinations. We use the standard grid-search technique to tune the hyper-parameters. For evaluation, we design two experimental settings:

(i) **Setting 1:** Our primary goal is to detect user accounts which are suspended by Twitter because they are spam accounts. Therefore, the set of suspended accounts constitutes the ground-truth for the classifiers. Out of all suspended accounts present in our dataset (mentioned in Section 2), we adopt leave-one-out cross-validation technique (due to the very limited number of suspended accounts) and report the average accuracy of the classifiers. Note that in this setting, we use one-class classifier for all the competing methods.

(ii) **Setting 2:** We believe that our method is capable of detecting those accounts which are spammers, but not suspended by Twitter yet. Therefore, we further invited human annotators to annotate some non-suspended accounts as spammers or non-spammers. This will further help us to run the baseline methods which originally used binary classifiers (see Section 5.4). Since it is not possible to label all non-suspended users, we adopt a convenient sampling approach. We define user bins according to the number of campaigns the non-suspended users exist (see the distribution in Figure 7(b)). Our sampling approach preferentially chooses users who are part of multiple campaigns to maximize the evidence per campaign – the probability of choosing a user belonging to multiple campaigns is higher than that for a user who is a part of a single campaign. Following this approach, we picked 700 users from 3,370 campaigns. Each user was labeled by three human annotators as spammers or non-spammers, and then the majority vote was considered as the final class. The inter-annotator agreement was 0.82 according to Cohen’s kappa measure.

Out of 700 manually annotated accounts, we hold out 20% of the dataset to be used as the test set in Setting 2. We repeat this experiment 50 times and report the average accuracy. Here also, we use one-class classifier for all the competing methods and consider ‘spammer’ as our target class.

**Evaluation metrics:** For comparative evaluation, we use the standard information retrieval metrics – Precision, Recall, F1-score, Area under the ROC curve (AUC).

### 5.3 Comparative Evaluation

Table 1 shows the performance of the competing methods for both settings. We report the results of our active-learning based one-class classifier with different feature combinations.  For setting 1 (leave-one-out), we report the performance w.r.t the accuracy (fraction of known spammers identified by the method) and observe that our method performs significantly well with only HMPS feature – it achieves an accuracy of 0.77, outperforming all baseline methods. However, incorporating OSN2 features along with HMPS further enhances 9.1% performance of our classifier, achieving an accuracy of 0.84.

A similar pattern is observed for setting 2. However, here our model with only HMPS turns out to be even stronger classifier, outperforming all others in terms of precision (0.99), F1-score (0.93) and AUC (0.88). Here also, incorporating most of the OSN features with HMPS does not enhance the performance of our method (or sometimes deteriorates the performance), except OSN2 which seems to be quite competitive. However, baseline 2 seems to be the best method w.r.t recall (0.92), but it significantly sacrifices the performance w.r.t. precision, F1-score, and AUC.

Nevertheless, we consider the following setting as our default method since it outperforms other methods in almost all experimental setup: HMPS + OSN2 + one-class classifier + active learning. Baseline 2 is considered as the best baseline method in the rest of the paper.

---

2All annotators were security researchers between the age group of 25 - 35 years.

3We tried with other combinations as well such as HMPS+OSN1+OSN2, HMPS+OSN2+OSN3 etc. The results were not encouraging enough to be reported in the paper.
Table 1: Comparative evaluation of the competing methods on two different experimental settings. For all the methods, one-class classifier is used. The colored row shows the performance (P: Precision, R: Recall, F1: F1-score) of our default method. The last row shows the results of our default method without active learning (see Section 5.5).

| Method | Feature | Setting 1 | Setting 2 |
|--------|---------|-----------|-----------|
|        |         | Accuracy  | P         | R         | F1        | AUC       |
| Baseline 1 | OSN1 | 0.62 | 0.86 | 0.71 | 0.77 | 0.48 |
| Baseline 2 | OSN2 | 0.58 | 0.84 | 0.92 | 0.87 | 0.52 |
| Baseline 3 | OSN3 | 0.62 | 0.86 | 0.66 | 0.74 | 0.47 |
| Our    | HMPS   | 0.77 | 0.99 | 0.87 | 0.93 | 0.88 |
|        | HMPS + OSN1 | 0.76 | 0.89 | 0.90 | 0.89 | 0.72 |
|        | HMPS + OSN2 | 0.84 | 0.98 | 0.88 | 0.93 | 0.87 |
|        | HMPS + OSN3 | 0.76 | 0.88 | 0.73 | 0.80 | 0.59 |
| Our    | HMPS + OSN2 - Active Learning | – | 0.42 | 0.98 | 0.55 | 0.51 |

Justification behind superior performance of HMPS: All of the baseline methods rely on the features that can be changed over time. These methods either consider URL attributes (baselines 1 and 3) within the tweets or changes in profile characteristics between a legitimate and spam user account (baselines 2 and 3). Given these specificities, it is easy for a spammer to manipulate these features. In contrast, HMPS relies on the monetization infrastructure (phone numbers) to identify campaigns and spammers. As discussed earlier, we aggregate tweets as part of the same campaign when they use multiple phone numbers wrapped around similar text. As a result, our method is resilient to spammers’ manipulation. Furthermore, to understand how HMPS helps in improving the detection of spammers over the baselines, we manually analyze a sample of ‘spammers’. Some of the users not identified by baselines 1 and 3 as spammers have a balanced number of friends and followers and a low number of tweets. In addition, users were not using URLs to spread the campaign. Therefore, all URL-based features do not aid in the detection task.

Baseline 2 measures the authority and hub scores based on the tweets with hashtags. As a result, it wrongly detects some benign users as spammers that were retweeting posts related to (say,) blood donation campaigns. When baseline 2 is combined with HMPS, the false positive rate is reduced since these users are not found in the spammer network.

In addition, HMPS can find spammers that are not suspended by Twitter yet. For instance, Figure 9 shows a spammer account that clearly violates the Twitter policy by promoting and posting repeated, pornographic content.4 Surprisingly, this account has not been suspended by Twitter yet. However, we found similar such accounts suspended by Twitter. Interestingly, our system was able to identify this account as a spammer.

These examples show that HMPS can identify spammers that use phone numbers, which are not detected by the baseline systems and / or Twitter, and is, therefore, more effective in detecting spammers that spread phone numbers to promote campaigns.

4https://support.twitter.com/articles/18311

5.4 One-class vs. 2-class Classifier

One may argue that the results reported in Table 1 may not reflect the original performance of the baseline methods since all the baseline methods originally used 2-class classifiers. Moreover, there was no empirical justification for adopting one-class classifier over 2-class classifier. To address these arguments, here we exactly replicate the baseline methods by considering the best 2-class classifier per baseline reported in the papers. We choose a balanced dataset of 150 suspended and 150 non-suspended users randomly sampled from our manually labeled dataset (see setting 2 in Section 5.2). For comparative evaluation, we consider several state-of-the-art 2-class classifiers (Logistic regression (LR), Latent Dirichlet Allocation (LDA), K-nearest neighbors (KNN), Decision Tree (DT), Naive Bayes (NB), Random Forest (RF), and Support Vector Machine (SVM)) and adopt them into our active learning framework. Table 2 shows that none of the baselines and our adopted 2-class classifiers outperform our default one-class classifier (last row of Table 2). Our default method is 12.6%, 7.7%, 10.7% and 9.7% higher than the second-ranked method (Decision Tree) in terms of precision, recall, F1-score, and AUC respectively. This result indicates that one-class classification is always helpful for the application where there is limited labeled data, and the label of most of the instances is unknown.

Table 2: Results of 2-class classifiers and comparison with our default one-class classifier. Here, the best 2-class classifiers reported in the papers are considered for the baselines.

| Method                      | Precision | Recall | F1-score | AUC  |
|-----------------------------|-----------|--------|----------|------|
| Baseline 1                  | 0.68      | 0.69   | 0.65     | 0.50 |
| Baseline 2                  | 0.47      | 0.57   | 0.51     | 0.50 |
| Baseline 3                  | 0.79      | 0.78   | 0.78     | 0.57 |
| Our 2-class classifier      |           |        |          |      |
| LR                          | 0.61      | 0.58   | 0.55     | 0.58 |
| LDA                         | 0.61      | 0.58   | 0.55     | 0.58 |
| KNN                         | 0.75      | 0.74   | 0.74     | 0.74 |
| DT                          | 0.83      | 0.83   | 0.83     | 0.83 |
| NB                          | 0.60      | 0.58   | 0.57     | 0.58 |
| SVM                         | 0.65      | 0.63   | 0.62     | 0.63 |
| RF                          | 0.83      | 0.82   | 0.82     | 0.82 |
| Our default one-class       |           |        |          |      |
| classifier                  | 0.95      | 0.90   | 0.93     | 0.92 |

Figure 9: An example spammer account (bio shown in (a), timeline shown in (b)) that has not been suspended by Twitter yet, but our system could detect it as spammer.
We argue that since the number of training samples is not sufficient, would increase the training set that in turn enhances the performance of the classifier. To verify our argument, we run our default method without active learning (without feedback) and observe that although recall increases significantly (0.98), it degrades the performance w.r.t. other performance measures – 57.1%, 40.8% and 41.3% degradation of precision, F1-score and AUC respectively (see the last row of Table 1). High recall with low precision indicates that most of the unknown users are classified as spammers by the general classifier. It happens due to the limited labeled data, which active learning can efficiently handle.

5.5 General vs. Active Learning

We argue that since the number of training samples is not sufficient for individual classifiers, feedback from one classifier to another would increase the training set that in turn enhances the performance of the classifier. To verify our argument, we run our default method without active learning (without feedback) and observe that although recall increases significantly (0.98), it degrades the performance w.r.t. other performance measures – 57.1%, 40.8% and 41.3% degradation of precision, F1-score and AUC respectively (see the last row of Table 1). High recall with low precision indicates that most of the unknown users are classified as spammers by the general classifier. It happens due to the limited labeled data, which active learning can efficiently handle.

5.6 Feedback vs. Oversampling

Since the size of the training set is small for each campaign-specific classifier, we use feedback across campaigns to increase the training set. As an alternative, one can also use other state-of-the-art oversampling techniques such as SMOTE [7]. Here, we adapt SMOTE for increasing the size of the training data (i.e., target class: spammers). The training set is oversampled by taking each training sample and introducing synthetic examples along the line segments joining any / all k neighbors of the training sample. Depending on the amount of oversampling required, k-nearest neighbors are randomly chosen. Table 3 shows the results for different values of oversampling ratio, i.e., the fraction of training set taken as the number of synthetic samples. In addition, we perform the oversampling technique before dividing the data into training and validation to ensure that the information from the training set is used in building the classifier. Table 3 shows that even after varying the ratio for oversampling, none of the cases can achieve the accuracy obtained from our feedback-based learning approach. This indicates that our feedback-based learning strategy is superior to the other oversampling strategy.

Table 3: Comparison of our feedback-based learning approach with standard oversampling approach (SMOTE). The term ‘Ratio’ indicates the fraction of training set taken as the number of synthetic samples generated by the oversampling technique.

| Oversampling + default one-class classifier | Precision | Recall | F1-Score | AUC  |
|---------------------------------------------|-----------|--------|----------|------|
| Ratio = 0.20                               | 0.90      | 0.64   | 0.64     | 0.59 |
| Ratio = 0.30                               | 0.88      | 0.74   | 0.74     | 0.63 |
| Ratio = 0.50                               | 0.81      | 0.71   | 0.68     | 0.58 |
| Ratio = 0.75                               | 0.91      | 0.68   | 0.69     | 0.56 |
| Ratio = 1                                 | 0.91      | 0.68   | 0.70     | 0.57 |

Feedback + default one-class classifier

| Precision | Recall | F1-Score | AUC  |
|-----------|--------|----------|------|
| 0.95      | 0.90   | 0.93     | 0.92 |

5.5 General vs. Active Learning

We argue that since the number of training samples is not sufficient for individual classifiers, feedback from one classifier to another would increase the training set that in turn enhances the performance of the classifier. To verify our argument, we run our default method without active learning (without feedback) and observe that although recall increases significantly (0.98), it degrades the performance w.r.t. other performance measures – 57.1%, 40.8% and 41.3% degradation of precision, F1-score and AUC respectively (see the last row of Table 1). High recall with low precision indicates that most of the unknown users are classified as spammers by the general classifier. It happens due to the limited labeled data, which active learning can efficiently handle.

6 RELATED WORK

In this section, we highlight the prior work on classification, clustering, and similarity tasks carried out using meta-paths. We also discuss several existing techniques to identify spammers.

Meta-path classification: Sun et al. [39] first proposed the idea of meta-path in heterogeneous network. Since then, it has been used extensively in various applications such as classification [20, 26], clustering [40], and similarity measures [36, 39]. Sun et al. proposed a measure called “PathSim” which outperformed Path Constrained Random Walk (PCRW) proposed by [23]. Meng et al. introduced biased constraint random walk to handle both symmetric and non-symmetric meta-paths [29]. They proposed Forward Stagewise Path Generation algorithm (or FSPG), which derives meta-paths that best predict the similarity between a node pair. Shi et al. proposed “HeteSim” to measure the relevance of any node pair in a meta-path [35]. To overcome the computational and memory complexity of HeteSim, Meng et al. proposed “AvgSim” that measures similarity score through two random walk processes along the given meta-path and the reverse meta-path [30]. Besides these similarity measures, Zhang et al. found node similarity based on connections between centers in X-star network [46]. While previous work considered meta-paths for classification assuming relevant meta-paths are known or finding greedy approaches to identify relevant meta-paths, in this work, we model Twitter as a heterogeneous network where relevant meta-paths are unknown. We further propose Hierarchical Meta-Path Scores to predict an unknown user as a spammer based on its neighboring spammers.

Spammer classification: Previous literature has addressed the problem of spam and spammers on Twitter and other OSNs [4–6, 8, 10, 13, 14, 16, 25, 27, 38, 41, 42, 44]. Benevenuto et al. used OSN based features to detect spammers on YouTube, with an accuracy of 98% [2, 3]. Researchers have identified spammers on Twitter [24, 28, 43] and blogging websites [19] using user-based features. Previous literature has also looked into identifying fake accounts on OSNs by examining characteristics of user profiles [37], by learning typical behavior of an account and flagging an account as suspicious in case of deviation [12], similarity to social bots [11, 15], or building a system that considers differences in which legitimate and malicious messages propagate through the network [32]. Link farming in Twitter where spammers acquire a large number of follower links has been investigated by Ghosh et al. [17]. These works treat OSN as a homogeneous network. Modeling OSNs as heterogeneous networks provide us several ways to capture structural similarity between a pair of nodes via meta-paths. In this work, we also demonstrated that our model works better than the baseline models that use only OSN-based features for classification.

7 CONCLUSION

In this paper, we detected spammers spreading spam campaigns using phone numbers on Twitter. We modeled Twitter as a heterogeneous network and proposed a collective classification approach that leverages heterogeneous nodes and their interconnections to identify unknown users as spammers. The significant contributions of our method are three-fold: (i) our proposed Hierarchical Meta-Path Score (HMPS) can efficiently measure how close an unknown user is w.r.t other known spammers; (ii) our proposed feedback-based active learning strategy is effective over three other baselines that use 2-class classifiers for spam detection; (iii) in case of small number of training instances, our proposed feedback strategy performs significantly better than other oversampling strategies. Through a case study and human-annotated dataset, we also showed that our method could find spammer accounts that are not suspended by Twitter yet.
REFERENCES

[1] Kayode Sakariyao Adewole, Nor Badrul Anuar, Amirrudin Kamsin, and Arun Kumar Sangaiya. 2017. SMSAD: a framework for spam message and spam account detection. Multimedia Tools and Applications (2017), 1–30.

[2] Fabrizio Benevenuto, Gabriel Magno, Tiago Rodrigues, and Virgilio Almeida. 2010. Detecting spammers on twitter. In Collaboration, electronic messaging, anti-abuse and spam conference (CEAS), Vol. 6. 1–12.

[3] Fabrizio Benevenuto, Tiago Rodrigues, Virgilio Almeida, Jussara Almeida, and Marco Gonçalves. 2009. Detecting spammers and content promoters in online video social networks. In Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval. ACM, 620–627.

[4] Yanzo Rishal, Dmytro Logothetis, Georgos Siganos, Jorge Leria, Jose Lorenzo, Mati Ripeana, and Konstantin Benesov: 2015. Intego: Leveraging Victim Prediction for Robust Fake Account Detection in OSNs. In NDSS, Vol. 15. 8–11.

[5] Qiang Cao, Michael Sirivivzon, Xiaowei Yang, and Tiago Pregueiro. 2012. Aiding the detection of fake accounts in large scale social online services. In Proceedings of the 9th USENIX conference on Networked Systems Design and Implementation. USENIX Association, 15–15.

[6] Qiang Cao, Xiaowei Yang, Jieqi Yu, and Christopher Palow. 2014. Uncovering large groups of active malicious accounts in online social networks. In Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security. ACM, 477–488.

[7] Nitesh V Chawla, Kevin W Bowyer, Lawrence O Hall, and W Philip Kegelmeyer. 2002. SMOTE: synthetic minority over-sampling technique. Journal of artificial intelligence research 16 (2002), 321–357.

[8] Zi Chi, Indra Widjaja, and Haining Wang. 2012. Detecting social spam campaigns on twitter. In International Conference on Applied Cryptography and Network Security. Springer, 455–472.

[9] Andrei Costin, Jelena Isacenkova, Marco Balduzzi, Aurélien Francillon, and Daniel Vidal Balzarotti. 2013. The role of phone numbers in understanding cyber-crime. IEEE Transactions on Dependable and Secure Computing, 12, 2–15.

[10] Kayode Sakariyah Adewole, Nor Badrul Anuar, Mohammed Kamarudin, and Arun Kumar Sangaiya. 2017. SMSAD: a framework for spam message and spam account detection. Multimedia Tools and Applications (2017), 1–30.

[11] George Danezis and Prateek Mittal. 2009. SybilInfer: Detecting Sybil Nodes using Collective Classification in Heterogeneous Information Networks. In Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining. ACM, 54–62.

[12] Manuel Egele, Gianluca Stringhini, Christopher Kruegel, and Giovanni Vigna. 2010. Detecting Criminal Organizations in Mobile Phone Networks. In Proceedings of the 24th Annual Computer Security Applications Conference. ACM, 1–9.

[13] Sangho Lee and Jong Kim. 2012. WarningBird: Detecting Suspicious URLs in Twitter Stream. In Proceedings of the 21st ACM international conference on Information and knowledge management. ACM, 614–622.

[14] Nazia Majeed, Pierre Mourlanne, Gregoire Jacob, Manuel Egele, Christopher Kruegel, and Giovanni Vigna. 2015. EVILCOHORT: Detecting Communities of Criminal Organizations. In Proceedings of the 34th Annual Computer Security Applications Conference. ACM, 1–9.

[15] Sharan Niladregh, Francois Labrecque, Alireza Sedighian, Ali Zend, Jose Fernandez, Christopher Kruegel, Gianluca Stringhini, and Giovanni Vigna. 2017. POSED: Spotting Twitter Spam Off the Beaten Paths. arXiv preprint arXiv:1708.00958 (2017).

[16] Bhavani Raskutti and Adam Kowalczyk. 2004. Extreme re-balancing for SVMs: a case study. ACM SIGKDD Explorations Newsletter, 6, 1 (2004), 60–69.

[17] Prathviraj Sen, Galileo Namata, Mustafa Bilgic, Lise Getoor, Brian Gallagher, and Tania Eliassi-Rad. 2008. Collective classification in network data. AI magazine, 29, 3 (2008), 93.

[18] Chuan Shi, Xiangan Kong, Philip S Yu, Sihong Xie, and Bin Wu. 2012. Relevance measure in large-scale heterogeneous networks. In Asia-Paciﬁc Web Conference. Springer, 636–643.

[19] Najmeh Miramirkhani, Oleksii Starov, and Nick Nikolaiakis. 2017. Dial one for scam: A large-scale analysis of technical support scams. In Proceedings of the 24th Network and Distributed System Security Symposium (NDSS). 1–15.

[20] Sharan Niladregh, Francois Labrecque, Alireza Sedighian, Ali Zend, Jose Fernandez, Christopher Kruegel, Gianluca Stringhini, and Giovanni Vigna. 2017. POSED: Spotting Twitter Spam Off the Beaten Paths. arXiv preprint arXiv:1708.00958 (2017).

[21] Bhatia Raskutti and Adam Kowalczyk. 2004. Extreme re-balancing for SVMs: a case study. ACM SIGKDD Explorations Newsletter, 6, 1 (2004), 60–69.

[22] Prathviraj Sen, Galileo Namata, Mustafa Bilgic, Lise Getoor, Brian Gallagher, and Tania Eliassi-Rad. 2008. Collective classification in network data. AI magazine, 29, 3 (2008), 93.

[23] Chuan Shi, Xiangan Kong, Philip S Yu, Sihong Xie, and Bin Wu. 2014. Hetsim: A general framework for relevance measure in heterogeneous networks. IEEE Transactions on Knowledge and Data Engineering, 26, 10 (2014), 2479–2492.

[24] Kurt Thomas, Chris Grier, Justin Ma, Vern Paxson, and Dawn Song. 2011. Designing and evaluation of a real-time url spam filtering service. In IEEE Symposium on Security and Privacy. IEEE, 447–462.

[25] Kurt Thomas, Chris Grier, Dawn Song, and Vern Paxson. 2011. Suspended accounts in retrospect: an analysis of twitter spam. In Proceedings of the 2011 ACM SIGCOMM conference on Internet measurement conference. ACM, 243–258.

[26] Bimal Viswanath, Muhammad Ahmad Bashir, Mark Crovella, Saikat Guha, Krishna P Gummadi, Balachander Krishnamurthy, and Alan Mislove. 2014. Towards Detecting Anomalous User Behavior in Online Social Networks. In Proceedings of the 21st international conference on Computer and communications security. ACM, 27–37.

[27] Usman Us Khan, Madhar Ali, Sameed Khan, and Albert Zomaya. 2016. Segregating Spammers and Unsolicited Bloggers from Genuine Experts on Twitter. IEEE Transactions on Dependable and Secure Computing (2016).

[28] Xiangan Kong, Bokai Cao, and Philip S Yu. 2013. Multi-label classiﬁcation by mining label and instance correlations from heterogeneous information networks. In Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining. ACM, 614–622.

[29] Shayne P Gummadi, Balachander Krishnamurthy, and Alan Mislove. 2014. Towards Detecting Anomalous User Behavior in Online Social Networks. In Proceedings of the 21st international conference on Computer and communications security. ACM, 614–622.

[30] Yuan Zhuang, Bokai Cao, and Philip S Yu. 2013. Multi-label classiﬁcation by mining label and instance correlations from heterogeneous information networks. In Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining. ACM, 614–622.

[31] Xiangan Kong, Bokai Cao, and Philip S Yu. 2013. Inferring anchor links across multiple heterogeneous social networks. In Proceedings of the 22nd ACM international conference on Information & Knowledge Management. ACM, 179–188.

[32] Xiangan Kong, Francesco Bonaventura, Gabriel Magno, Tiago Rodrigues, and Virgilio Almeida. 2010. Detecting spammers on twitter. In Collaboration, electronic messaging, anti-abuse and spam conference (CEAS), Vol. 6. 1–12.

[33] Xiangan Kong, Jiewei Zhang, and Philip S Yu. 2013. Inferring anchor links across multiple heterogeneous social networks. In Proceedings of the 22nd ACM international conference on Information & Knowledge Management. ACM, 179–188.

[34] Xiangan Kong, Francesco Bonaventura, Gabriel Magno, Tiago Rodrigues, and Virgilio Almeida. 2010. Detecting spammers on twitter. In Collaboration, electronic messaging, anti-abuse and spam conference (CEAS), Vol. 6. 1–12.

[35] Mingzai Zhang, Hao Hu, Zhenying He, and Wei Wang. 2015. Top-k similarity search in heterogeneous information networks with x-star network schema. Expert Systems with Applications, 42, 2 (2015), 699–712.

[36] Xinhao Zhang, Zhaoxing Li, Zhiqiong Zhu, and Wenzin Liang. 2016. Detecting Spam and Promoting Campaigns in Twitter. ACM Trans. Web, 10, 1 (2016), 41–428.