Pseudo-honeypot: Toward Efficient and Scalable Spam Sniffer

Yihe Zhang*, Hao Zhang†, Xu Yuan*, and Nian-Feng Tzeng*
* University of Louisiana at Lafayette, Lafayette, LA, USA
† IEEE Member, USA

***Abstract*—Honeypot-based spammer gathering solutions usually lack attribute variability, deployment flexibility, and network scalability, deemed as their common drawbacks. This paper explores pseudo-honeypot, a novel honeypot-like system to overcome such drawbacks, for efficient and scalable spammer sniffing. The pseudo-honeypot takes advantage of user diversity and selects normal accounts, with attributes that have the higher potential of attracting spammers, as the parasitic bodies. By harnessing such category of users, pseudo-honeypot can monitor their streaming posts and behavioral patterns transparently. When compared with its traditional honeypot counterpart, the proposed solution offers the substantial advantages of attribute variability, deployment flexibility, network scalability, and system portability. Meanwhile, it offers a novel method to collect the social network dataset that has a higher probability of including spams and spammers, without being noticed by advanced spammers. We take the Twitter social network as an example to exhibit its system design, including pseudo-honeypot nodes selection, monitoring, feature extraction, ground truth labeling, and learning-based classification. Through experiments, we demonstrate the efficiency of pseudo-honeypot in terms of spams and spammers gathering. In particular, we confirm our solution can garner spammers at least 19 times faster than the state-of-the-art honeypot-based counterpart.**

I. Introduction

The popularity of online social networks, such as Twitter, Facebook, and Instagram, has made them indispensable in our daily life. The success of online social networks not only has provided advanced tools for users to chat with friends, keep in touch with family, and share news, but also has the potentials of bringing economic profit and political influence. For example, in the 2016 U.S. presidential election, Twitter offered the largest source of breaking news with around 40 million election-related tweets [14]. Such amount of news has widely influenced voters' opinions to some extent.

However, spammers have been the adversaries since the inception of online social networks, pervasively annoying users. By creating fake accounts or compromising benign ones, spammers can initialize social relationships and send unsolicited requests or messages. Such requests or messages, containing malware URLs, phishing or deception information, can spread the harm to victims. The frequent appearance of these harms has adversely impacted the quality of users' experience, stolen private information, caused economic loss, and possibly changed victims' political opinions. In particular, a report [31] shows that the spam messages in Twitter have changed public opinions resulting from some 1.4 million affected users who inadvertently interacted with spammers during the 2016 U.S. presidential election.

To date, there has been extensive research [1], [7], [24], [18], [12], [21], [8], [29] on exploiting effective mechanisms to capture spammers. These mechanisms focus either on classifying the spam messages from large-scale network contents or analyzing the social relationships to identify fake/compromised accounts. Also, the graph-based approaches and statistical models [4], [35], [15], [3], [10] are employed to analyze the social relationships among a large number of accounts with the aim to identify the anomaly accounts as fake or compromised ones. Obviously, these efforts are time-consuming and inefficient since they filter spams (or spammers) from a large set of network contents (or accounts), but can only detect a small portion of spammers.

On the other hand, the honeypot was introduced as a promising solution by manually creating accounts as lures for spammers [27], [16], [22], [17], [38]. By imitating as a normal account with some specific attributes that suit spammers' tastes, a social honeypot can stay in the social network with an intent to attract spammers. Although honeypot can trap spammer accounts and only need to process on a small set of data, it suffers from major drawbacks, including deployment flexibility, attribute variability, and network scalability. The rationale behind is the manual setup which involves considerable human efforts. Thus, it incurs heavy deployment overhead and is impractical to scale up in a large-sized network to trap spammers. Moreover, even the honeypot acts as an ordinary account, it is still possible to be recognized by spammers with the evolution of smart spam techniques.

In this paper, we propose a novel spammer gathering and detection system, called pseudo-honeypot, by taking advantage of the diversity of online social users and utilizing it as the key resources to trap spammers. Instead of creating artificial accounts, we allow the pseudo-honeypot to take a normal user as the parasitic body and harness such a user to monitor spam activities. In particular, the pseudo-honeypot nodes are selected from a pool of normal users that are more vulnerable to spammers. From this point of view, pseudo-honeypot is envisioned to perform the similar function as honeypot in attracting and trapping spammers but has much rich diversity in user attributes. By harnessing normal users, the pseudo-honeypot can monitor the streaming posts and behavior patterns that have a higher probability of including spam messages. Furthermore, such pseudo-honeypot system can be easily scaled to the large size of networks for spammer capture by harnessing more users. As a result, it can be readily understood that the proposed pseudo-honeypot system offers the advantages of attribute variability, deployment flexibility, and network scalability, which will be further discussed in Section II.

To concretize our discussion, we take Twitter social network as an example to illustrate the design of pseudo-honeypot and show its advantages on spammers capturing. By carefully identifying a set of attributes that have high potentials in attracting spammers, we select the accounts that possess such attributes to serve as pseudo-honeypot nodes. Twitter's API enables us to select suitable pseudo-honeypot nodes periodically and to monitor the selected accounts transparently. It should be noted that the employed Twitter APIs (i.e., Streaming API and Restful) monitor only public information and the Twitter API's privacy policy terms are strictly followed by our pseudo-honeypot system. To classify spams/spammers from the monitored data with high accuracy, we design a pseudo-honeypot detector by employing a machine learning-based approach, which requires a ground truth dataset for training. To produce the quality training dataset, we first extract a rich set of features that can reflect spammer's characteristics. Then, we use diversified methods (i.e., checking suspended accounts, clustering, and the rule-based method) to pre-process a raw dataset so as to generate a roughly labeled one and then perform manual checking to refine it into a reliable ground truth dataset for training. In the end, we equip the machine learning algorithms in our detector to classify the collected tweets. To validate the performance of our system, we periodically create one pseudo-honeypot network with $2,400$ nodes in each hour and conduct experiments for a total of 700 hours.

Our main contributions are summarized as follows:

- We propose a novel spammer gathering framework, named pseudo-honeypot, for social networks. This framework systematically differs from the honeypot, as it takes advantage of the existing environment and offers the substantial advantages of attribute variability, deployment flexibility, network scalability, and system portability.
- We present the system design of the pseudo-honeypot in Twitter social networks and show its significant performance improvement on spammer capturing and better scalability by doing away with the manual setup for honeypot construction while enriching its attributes diversity.
- We implement the pseudo-honeypot system and conduct experiments to demonstrate its effectiveness. During a 700-hour experiment, we collect $5,618,476$ tweets posted by $2,785,815$ unique users, in which we classify a total of $1,208,375$ spams and $50,966$ spammers.
- We show the effectiveness of prevalent attributes and refine the top 10 most likely attributes that attract spammers for use to create an advanced pseudo-honeypot system. The experimental results show that such an advanced pseudo-honeypot system can garner spammers at least 19 times faster than the prominent honeypot systems.

The remainder of this paper is organized as follows. In Section II, we discuss our problem and propose the pseudo-honeypot as a novel framework for spammer gathering in online social networks. In Section III, we take Twitter social network as an example to present the system design of pseudo-honeypot. Section IV presents our pseudo-honeypot spam detector design, including feature extraction, ground truth labeling, and machine learning-oriented classification. Based on this design, we present the system implementation and use experimental results to demonstrate the effectiveness of our proposed system in Section V. Section VI discusses related work and Section VII concludes this paper.

II. Pseudo Honeypot Overview

In this paper, we study the spams collection and detection problem in online social networks. Our goal is to develop an efficient and effective mechanism to monitor the online social network contents that are likely to include spam messages and then identify them so as to remove them. We propose a novel system, named pseudo-honeypot, to achieve this goal. By taking advantage of user diversity, our pseudo-honeypot is constructed on top of normal users that may suffer spammy behaviors. These users are selected with attributes that meet spammer's tastes, thus likely to attract spammers. Instead of artificially creating fake accounts, pseudo-honeypot can still perform the similar function as honeypot in spammer attraction. Although the pseudo-honeypot takes normal users as its carrier, it can be constructed not to affect normal users' activities nor to make any change to their posts or behavioral patterns so as to meet the social network privacy policy. Thus, it is invisible to normal users and cannot be recognized by spammers. By harnessing these accounts, pseudo-honeypot can monitor streaming posts and behavioral patterns in a real-time manner. With monitored data, we can design suitable spam detector by extracting features that reflect spammer's characteristics and leveraging machine learning-based approaches for classification.

The pseudo-honeypot framework possesses salient advantages, as follows.

- **Nodes availability.** By taking normal user accounts as the parasitic body, pseudo-honeypot has a large number of available candidates in a social network. The involved construction cost is trivial when compared to placing the artificial honeypots. Meanwhile, in case some pseudo-honeypot nodes fail (i.e., accounts become dead or lose spammer's interests), our system can quickly select other candidates to serve as new pseudo-honeypot nodes. Thus, this approach provides abundant node availability with a small creation cost.
- **Deployment flexibility.** Its deployment flexibility manifests two aspects: candidates and attributes. From the candidate selection perspective, the pseudo-honeypot has the flexibility of switching its parasitic bodies among candidates in social networks. Such flexibility is critical to make the pseudo-honeypot always involving accounts that are attractive of spammers' interests. From the attribute

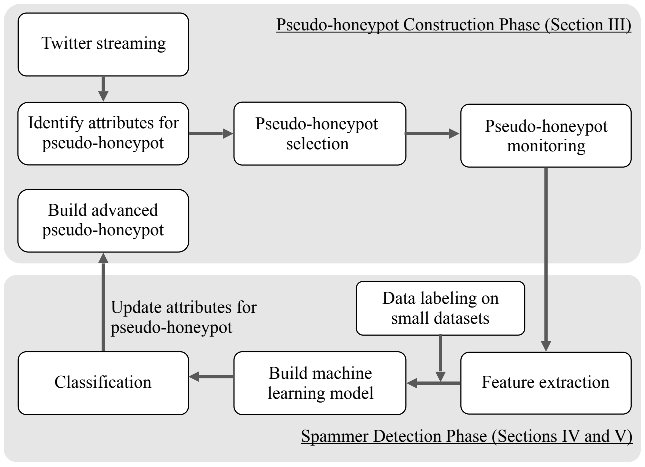

Pseudo-honeypot Construction Phase (Section III)
Twitter streaming
Identify attributes for pseudo-honeypot
Pseudo-honeypot selection
Pseudo-honeypot monitoring
Build advanced pseudo-honeypot
Update attributes for pseudo-honeypot
Data labeling on small datasets
Classification
Build machine learning model
Feature extraction
Spammer Detection Phase (Sections IV and V)

Fig. 1. An illustration of system design of the pseudo-honeypot.

selection perspective, the normal users provide a large variety of attributes for attracting spammers. This enables pseudo-honeypots to flexibly update their desirable attributes without manual construction.

- **Attributes variability.** The normal users offer abundant varieties of attributes that can be well utilized by pseudo-honeypots. Instead of manually setting up attributes in a honeypot, our framework can select desirable ones within existing attributes quickly and automatically so as to reduce the deployment cost. Moreover, some honeypot attributes are difficult, if not impossible, to manually set up in a short time, such as a long account age (e.g., 3 years) and plentiful friends/followers.
- **Network scalability.** The manual operations (of creating artificial accounts and configuring attributes) in a honeypot system are prohibitively involved in establishing a large-scale honeypot network. However, pseudo-honeypot can directly select the desired attributes and associate normal users with them. This avoids significant manual setup overhead, thus permitting to generate an arbitrarily sized pseudo-honeypot network.

Our pseudo-honeypot offers many possibilities and potential benefits to capture spammers in a social network. To be specific, we take the Twitter social network as an example to present its system design and fully explore its advantages. As shown in Figure 1, our system design consists of the following core components: *i)* Pseudo-honeypot selection, *ii)* Pseudo-honeypot monitoring, *iii)* Feature extraction, *iv)* Ground truth data labeling, and *v)* Spammer classification.

III. Pseudo-Honeypot System Design in Twitter Social Network

In this section, we offer an in-depth study of the pseudo-honeypot by designing efficient mechanisms for spam and spammer gathering in Twitter social networks.

A. Challenges of Selecting Pseudo-honeypot

One critical step in constructing the pseudo-honeypot network is to find suitable user accounts with attributes which meet spammers' interests. In this step, there are a number of technical challenges that one must address.

- *Transparent to normal users.* Since pseudo-honeypot takes normal user accounts as the parasitic body, one key criterion is that pseudo-honeypot's activities have to be utterly transparent to users so as to observe Twitter's security and privacy policy. In sharp contrast, the honeypot relies on the operators to have the privilege to access and manage target user accounts, whose activities may be visible to affected accounts. It is a big challenge in the design of pseudo-honeypot to meet such transparency criterion.
- *Pseudo-honeypot selection.* There are billions of user accounts in the Twitter social network, but not all of them are suitable to serve as the pseudo-honeypot nodes. Carefully and strictly screening user accounts is required in order to keep high efficiency in spammer capturing. The attributes are key to guide our selection of pseudo-honeypot nodes. That is, we target the type of user accounts that possess similar attributes as those of the honeypot so that those target users are highly likely to attract spammers. Unfortunately, the wide variety of attributes from billions of users provides a rich attribute set but also comes with difficulties in determining the top ones that best meet spammers' tastes. Therefore, how to extract the suitable attributes and select accounts which possess them from billions of Twitter users, are challenging.
- *Pseudo-honeypot portability.* Since there is a large pool of available Twitter accounts, it is essential to let the pseudo-honeypot shift across accounts dynamically. Such portability allows pseudo-honeypot to smartly drop the ineffective ones, always keeping those that attract spammers' interests the most. This is in contrast to the static, passive, and immutable properties of the honeypot while exhibiting improved spammer capturing efficiency. But the implementation of such a dynamical property raises the issue of deciding the lifetime of each pseudo-honeypot node. Due to the variety of user account activities, it is extremely challenging in deciding when to shift to new accounts.

B. Attribute-based Pseudo-honeypot Selection

In order to select the effective attributes for pseudo-honeypot, we take the reverse engineering strategy. That is, we first take a large pool of prevalent attributes that have been widely used in previous research and then use them as the criteria to select pseudo-honeypot nodes for creating a sample pseudo-honeypot system. After running it for a period, we will examine these attributes and refine the top ones that have the highest probability of capturing spammers. These refined attributes will govern our design of a highly effective pseudo-honeypot system. As the account screening is extremely fast (to be shown in Section V), pseudo-honeypot makes it easy to select any sized attributes pool in the design. Here, we just take into account the prevalent attributes from [34], [37],

TABLE I
ATTRIBUTE CATEGORIES OF PSEUDO-HONEYPOT SELECTION.

Category	Attributes
C1: Profile-based	friends count, followers count, total friends and followers, ratio of friend and followers, account age (days), lists count, favorites count, statuses count, average of lists per day, average of favorites per day, average of statues per day
C2: Hashtag-based	entertainment, general, business, tech, education, environment, social, astrology, no hashtag
C3: Trending-based	trending-up topics, trending-down topics, popular tweets, no-trending topics

[24], [38] as the sample examples to demonstrate pseudo-honeypot's feasibility and advantages, by grouping them into the following three categories.

- *Profile-based attributes* contain personal information presented by concrete labels in Twitter. As shown in Category C1 of Table I, we extract 11 most critical social attributes to characterize user's profiles in Twitter.
- *Hashtag-based attributes* identify messages on some specific topics that cluster tweets into different groups. In our system design, we mainly consider eight types of hashtags as outlined in C2 of Table I, namely, *entertainment, general, business, tech, education, environment, social, astrology* and *no hashtag*.
- *Trending-based attributes* represent the trending of discussed topics in tweets. They reflect the changes of users' interests in some topics with the attributes of trending up or down (in C3 of Table I), which can highly impact some spammer's strategies.

C. Constructing Pseudo-honeypot

We use the attribute set as shown in Table I to select a collection of accounts, with each account satisfying at least one attribute. The pseudo-honeypot takes selected accounts as parasitic bodies and monitor the posted tweets and behavioral patterns crossing these accounts. Note that the operation of pseudo-honeypot should always stay transparent to the harnessed user accounts. This can be achieved by leveraging the popular Twitter APIs (e.g., RESTful or Streaming API) that are available to developers.[1] Such APIs can help pseudo-honeypot to monitor account activities (i.e., posted tweets and behavioral patterns) while staying invisible to them. Til now, our pseudo-honeypot has been constructed with similar spammer attraction as that of the honeypot, while offering the advantages of flexible attribute selections and lowered chances to be recognized by spammers.

[1]Most social networks provide such APIs, for example, Reddit API in Reddit, Tumblr API in Tumblr, Search API in Facebook/Instagram/Google+, and so on.

D. Improving Pseudo-honeypot Performance

In Twitter, one account may not keep attracting spammers' interests for an extended time period. According to our observation, if one account has more recent activities, it has a higher potential in attracting spammers. Thus, it is essential for our system to have high portability for improving its efficiency of spammers capturing by building pseudo-honeypot only over active accounts. To explore the portability property, we define a user's status as either *Active* or *Dormant*. If a user account posts new tweets and brings lots of mentions/replies in a certain time interval, it is referred to as in the *Active Status*. On the other hand, if a user does not post new tweets for a certain time duration, or it posts new tweets but brings few or no mentions/replies, it is called in the *Dormant Status*. To maintain high efficiency, we aim to make pseudo-honeypot stay only over the *Active* users.

E. Pseudo-honeypot Monitoring

The pseudo-honeypot network has been constructed with aforementioned steps, able to start monitoring tweets and users' behavioral patterns. Currently, we collect only the direct interactive behaviors, instead of all streaming passing through the associated accounts, to reduce the processing workload of the pseudo-honeypot network. Many direct interactive behaviors can be explored and utilized by pseudo-honeypot if needed. In this design, we only take "mentions" behaviors as an example. The advantages of this design are twofold. First, most information in streaming tweets is benign, so it is expensive to remove them among all streams. Second, the streams with "mention" may include the most severe spammer behaviors and provide much valuable information to garner spammers.

The tweets we collected can be classified into three categories: (1) pseudo-honeypot accounts' activities, including posts, retweets, and quotes; (2) other normal accounts mentioning the pseudo-honeypot accounts; (3) spammers mentioning pseudo-honeypot accounts. Category (1) belongs to normal behaviors if the users are not spammers. But, based on our pseudo-honeypot selection approach, the selected accounts can be spammers. Thus, the tweets in Category (1) may be spam messages. Category (2) reflects the normal behaviors, while Category (3) contains anomaly network behaviors. Our goal is to identify the spams under Category (1) and differentiate Category (3) from Category (2).

F. Ethical Considerations

Since pseudo-honeypot employs normal accounts as the carriers, we should put strict restrictions on its operation. One restriction is that it is not allowed to mine secret information of users. All monitored and collected information should be visible to the public. Another restriction is that it is disallowed to perform social activities or conduct any interaction with the carriers or other accounts, so as not to interrupt Twitter users. As we employ Twitter APIs (i.e., Streaming and Restful), these two restrictions can be readily met as Twitter APIs apply the Developer Agreement and Policy of Twitter [32] to regulate

API use by developers. The Twitter API's privacy policy is fully observed by our pseudo-honeypot system. As a result, the pseudo-honeypot mechanism is considered to be both ethical and legal.

IV. Pseudo-Honeypot Spam Detector

To evaluate pseudo-honeypot performance, it is infeasible to check all collected tweets manually. Instead, we adopt the machine learning-based approach to design a pseudo-honeypot spam detector so as to analyze spam messages from Categories (1), (2), and (3). Our detector includes three components. First, we extract a rich set of features that can reflect tweets' characteristics. Next, we use diversified methods to label a ground truth data for training to cover the major types of spams. In the end, we employ the machine learning algorithms to classify the collected tweets.

A. Feature Extraction

To understand tweets' characteristics that reflect spams or spammers' attributes, we extract a total of 58 features categorized as follows.

Account Profile. We aim to extract a rich set of accounts information that is included in tweets. A user is defined as the sender if posting tweets to (or retweeting from) others, and as the receiver if mentioned by others. We extract 16 profile features from sender accounts. These features include friends count, followers count, age, status count, average statuses, list count, average lists, average favourites, favorites count, verified status, default profile image, screen name length, name length, description length, description emoji count, and description digits count. Similarly, the same 16 features are extracted from receiver accounts.[2] Note here, these profile features can be extracted from tweet's JavaScript Object Notation (JSON).

Tweet Contents. By analyzing tweet contents, we extract 8 statistic features, including if the tweet is repeated, tweet status (i.e., tweet, retweet, quote), tweet source (web, mobile, third-party, others), hashtag count, mention count, content length, content emoji count, and content digits count.

User Behaviors. To reflect user's behavioral patterns, we extract a total of 18 features, which are described next.

- *Reciprocity count:* The number of conversations between a sender and a receiver.
- *Sender (or receiver) tweet distribution:* The percentages of the tweet, retweet, and quote from the same sender (or to the same receiver). This includes a total of 6 features at the sender and receiver.
- *Sender (or receiver) tweet source distribution:* The percentage of tweets under each tweet source (web, mobile, third party, and others) associated with a sender (or a receiver). This includes a total of 8 features at the sender and receiver.

[2]We consider only those receivers who are pseudo-honeypot nodes or have mentioned the pseudo-honeypot, since all other account information cannot be singled out.

- *Mention time:* Once a user updates a new post, assuming at T_{post}, it will take a certain time for other users to see this update (assuming at $T_{mention}$) and then react to this update after a period. We define mention time f_m as the time interval between these two activities, i.e.,

$$f_m = T_{mention} - T_{post}.$$

  The importance of this feature stems from considering the reaction time differences of normal users and of spammers. For normal users, the *mention time* is relatively longer since users need time to read the post and reply to it. For spammers, however, they target the victims and start spam behaviors with little consideration to tweet contents, thus incurring a short time for reaction.
- *Average tweet intervals:* This feature calculates the average intervals of any two neighboring tweets. The average intervals of tweets sending (or received) from a sender (or by a receiver) are expressed by:

$$t = \frac{\sum(T_{i+1} - T_i)}{N_u},$$

  where T_i denotes the time when a sender (or receiver) sends (or receives) a tweet and N_u denotes the total number of tweets that have been sent (or received).
- *Environment score:* Inspired by spammers' interests in various attributes, the *group likelihood score* p_i is used to denote the probability of an attribute i in attracting spammer's interests. With such score, we define a new feature, named environment score f_{score}, to express the contribution of an attribute to spam identification, which can be calculated as follows:

$$f_{score} = \begin{cases} \max(p_i), & \forall p_i \in P_{attr}, \\ \tau, & otherwise, \end{cases}$$

  where P_{attr} denotes the set of probability values with respect to all attributes. That is, the highest group likelihood score in this set will be selected as the environment score, i.e., $f_{score} = \max(p_i), \forall p_i \in P_{attr}$. If $P_{attr} = \emptyset$, there is no spam found yet within a group of attributes. Then, we set the score as a small constant value τ, i.e., $f_{score} = \tau$. Note that both P_{attr} and f_{score} will be updated once new spams are found by any attribute.

B. Ground Truth Labeling

Ground truth labeling is used for training data based on selected features. However, the lack of reliable ground truth is known as a challenge for real-world data. To acquire a reliable ground truth dataset, we first rely on preprocessing below to obtain the roughly labeled data: 1) checking suspended accounts, 2) clustering-based approach, and 3) the rule-based method. After that, we perform manual checking on such roughly labeled data to refine a reliable ground truth dataset. This method can significantly reduce the labeling cost while maintaining good reliability.

Suspended Account. Twitter suspends the user accounts that violate Twitter rules [33] to maintain a clean social environment. The flagged twitter accounts can help us to label a portion of data in the ground truth dataset. Notably, a suspended account is not necessary a spam account, but our manual checking in the last step can further filter these accounts.

Clustering Based Method. To label the remaining dataset, we employ the clustering method, which has been widely applied [5], [21], [24], [25] to detect the spammer campaigns and near-duplicate tweets. Our clustering is based on four types of data, i.e., profile image, screen name, user description, and tweet contents, where the first three are from the user profiles [30], [13] and the last one is from tweets [24].

First, we cluster users with similar images into the same group. This method is plausible since a spam campaign typically uses similar images in the profile even they have different URLs [13]. The dHash (Different Hash) algorithms [28] can be utilized to dig the similar images as follows:

- Reduce the original image into a constant size (i.e., 9*9) by removing high frequencies and detailed information of the image and then transforming color into grayscale.
- Compare the adjacent pixels in the image. If one pixel is greater (or smaller) than the next one (considering both horizontal and vertical directions), we set it to 1 (or 0). We then transform these binary values into hexadecimal values and concatenate the two 64-bit values together to get a 128-bit hash.
- Calculate the difference of any two images (128-bit hash) by using Hamming distance, i.e., $d(h1, h2) = \sum XOR(h1, h2)$.

For any two images, if their Hamming distance is smaller than a threshold (i.e., 5), we put them into the same group.

Second, we group together the users with specific patterns in their screen names. A spam campaign typically registers its accounts with automatic naming patterns which have relatively limited variability [30]. We may adopt the similar regular expression method [19] that has a low false-positive rate in URL pattern [36], text template [23], or merchant patterns [30] to extract user screen names. Then, we match each screen name to a sequence Σ-Seq by a pre-defined character classes $\Sigma = \{p\{Lu\}, p\{Ll\}, p\{N\}, p\{P\}\}$, where $p\{Lu\}$ $p\{Ll\}$, $p\{N\}$, and $p\{P\}$ indicate the uppercase, lowercase, numeric characters, and punctuation characters, respectively. We keep those groups that have 5 or more members in a sequence.

Third, we cluster users by analyzing their descriptions. We process a user description by removing its URL, emoji, stop words, and special characters, before employing the MinHash algorithm [26] to find near-duplicate descriptions among all users. We consider two descriptions identical if their minimum hash values of the tri-grams shinglings are the same.

In the end, we set a 1-day time window to check near-duplicated tweets. We filter out tweet contents that are less than 20 characters to check the duplication.

After we group the user accounts and tweets, we label spammers and spams via the following criteria: If a user in one group is suspended by Twitter, we label all users in this group as spammers; If a tweet in one group is labeled as a spammer, we label its users and all tweets in this group as spammers and spams, respectively.

Rule-Based Method. This is a complementary step to label the remaining dataset. Specifically, we set the following rules/policies to label the spams and non-spams.

- A tweet is labeled as spam if it falls into one of the following conditions: 1) has malicious URL; 2) includes repetitive information; 3) includes deceptive information; 4) has pertinence purpose; 5) includes many meaningless tweets; 6) has relevant information on free or quick money gain; 7) includes adult content; 8) is an automatic tweet from bots/app with the malicious purpose; 9) is from malicious promoters; 10) is friend infiltrators. 11) includes sensitive or offensive contents.
- We label non-spam tweets by defining seed accounts. A large set of truthful accounts (from governments, famous companies, organizations, and well-known persons) are considered as the seed and their tweets are labeled as non-spams.
- We label a tweet as spam if a certain symbol exists and it is from a group of users with the same affiliation while the users in this group perform spamming behaviors.

Lastly, we perform manual checking both in the labeled dataset obtained from aforementioned steps and the remaining unlabeled dataset to refine a reliable ground truth dataset. With the combination of these approaches, our labeled ground truth dataset covers a broad range of spams and spammers that can reflect their characteristics of diversity.

C. *Machine Learning Based Spammer Detector*

Our detector relies on the machine learning-based methods to perform intelligent spams/spammers classification. There are a variety of prominent machine learning methods that may be used to classify spammers, such as Decision Tree (DT) [2], Support Vector Machine (SVM)[39], Gradient Boosting (GB)/Extreme Gradient Boosting (EGB) [20], [11], k-Nearest Neighbors Algorithm (kNN) [7], and Random Forest (RF) [38]. We test all these machine learning classifiers on our labeled ground truth dataset with 10-fold cross-validation and choose the most accurate one for use in our detector.

Since spammers' taste may change over time in practice, the Twitter spammer drift problem [6] is challenging in the design of pseudo-honeypot. One strategy is to apply the reverse engineering strategy by keeping track of the spammers' tastes in real time. The pseudo-honeypot can update its spam features automatically in a real-time manner once there are new spams captured. Meanwhile, the ground truth training dataset also keeps updating. As spammer drift problem is out of the scope of this paper, we omit its discussion here to conserve space.

V. Evaluation and Results Discussion

In this section, we describe the implementation of our pseudo-honeypot system and present experimental results to demonstrate its superior capability in capturing spams and spammers.

TABLE II
THE LISTS OF PROFILE-BASED ATTRIBUTES AND THEIR SAMPLING VALUES.

Index	**Attribute**	**Sample value**	**Total selected accounts**
1	friends count	10 50 100 200 300 500 1k 3k 5k 10k	100
2	follower count	10 50 100 200 300 500 1k 3k 5k 10k	100
3	total friends and follower	20 100 200 500 1k 2k 3k 5k 10k 30k	100
4	ratio of friend and follower	1/10 1/8 1/4 1/2 1 2 4 6 8 10	100
5	account age (days)	10 50 100 300 500 1k 1.5k 2k 2.5k 3k	100
6	lists count	10 20 30 40 50 70 100 200 300 500	100
7	favorites count	10 50 100 500 1k 5k 10k 50k 100k 200k	100
8	status count	10 50 100 500 1k 5k 10k 50k 100k 200k	100
9	average of list per day	1/100 1/50 1/20 1/10 1/8 1/6 1/4 1/2 1 2	100
10	average of favorites per day	1/50 1/10 1/5 1/2 1 2 3 5 10 50	100
11	average of statues per day	1/50 1/10 1/5 1/2 1 2 3 4 10 50	100

A. System Implementation

Our system is implemented by selecting a set of user accounts that include a total of 24 attributes, as discussed in Section III-B. For profile-based category listed in Table I, we consider each attribute with 10 different sample values while selecting 10 user accounts having each sample attribute value to serve as the pseudo-honeypot nodes. Consider the attribute of *friends count*, for example, we have 10 sample values, namely, with friend amounts equal to 10, 50, 100, 200, 300, 500, 1000, 3000, 5000, and 10000. Each sample value is then to include 10 individual accounts selected as the pseudo-honeypot nodes, for a total of 100 pseudo-honeypot nodes under the *friends count* attribute. The sample values of each attribute in profile-based category are listed in Table II, indicating the construction of 1100 pseudo-honeypot nodes totally. In the hashtag-based category, we identify the top 10 hashtags (from [9]) in each attribute and select 10 accounts possessing each hashtag to serve as pseudo-honeypot nodes. That is, we have 100 pseudo-honeypot nodes each for *entertainment, general, business, tech, education, environment, social,* and *astrology*. For the *no hashtag* attribute, we randomly select 100 accounts that are posting tweets without any hashtag. In total, we have 900 pseudo-honeypot nodes in the hashtag-based category. For the trending-based category, each attribute in [9] identifies its top 10 topics and each of which determines 10 user accounts to serve as the pseudo-honeypot nodes. Hence, there are 100 pseudo-honeypot nodes each with the *trending up, trending down,* and *popular* topics. For the *non-trending* topic, we randomly select 100 accounts that do not post tweets with any topic in [9]. Hence, there are 400 pseudo-honeypot nodes totally under the trending-based category.

Overall, we have created a pseudo-honeypot network with 2400 pseudo-honeypot nodes. The time to create such a pseudo-honeypot network is less than 1 min, substantially shorter than the traditional honeypot-based solution. Notably, the nodes in a pseudo-honeypot are not static and are migrated to another group of accounts after a specific time duration. The new group of accounts is selected with the same criteria. In our implementation, we set the time duration of our pseudo-honeypot network on a group of accounts to be 1 hour.

The implementation of our system is written in Python with the Tweepy library. We rely on a streaming API in Tweepy to retrieve tweets and filter Twitter user accounts. The streaming API used in the pseudo-honeypot implementation enables real-time tweet monitoring. An account selected as a pseudo-honeypot node is denoted by a filter, represented in the form of: the mention notation '@' followed with the user account name in Streaming API (i.e., @user_account_name). For example, if the account of $Mykhaylo\_bowning$ is taken as a pseudo-honeypot, it is denoted by $@Mykhaylo\_bowning$ in the Streaming API, signifying that all tweets crossing $Mykhaylo\_bowning$ are acquired.

TABLE III
THE LABELED SPAMS, SPAMMERS AND THEIR PERCENTAGES IN THE GROUND TRUTH DATA BY EACH METHOD. THE TOTAL NUMBER OF TWEETS: 161,633. THE TOTAL NUMBER OF USERS: 73,487.

Categories	**# of spams**	**% of tweets**	**# of spammers**	**% of users**
Suspended	10,858	6.72	3697	5.03
Clustering	4121	2.55	1281	1.74
Rule Based	3221	1.99	862	1.17
Human Labeling	1096	0.68	256	0.35

B. Experiments

We conduct extensive experiments to evaluate the proposed pseudo-honeypot system by running it for a total of 700 hours. The pseudo-honeypot nodes are switched to different user accounts once in an hour. In the 700-hour experiment, we collected a total of $5,618,476$ mention behavioral tweets, with a total of $2,785,815$ unique accounts involved. The collected tweets are analyzed by a computer with one Intel Core i5-6600K CPU (quad-core) and 32 GB RAM.

C. Ground Truth Labeling and Model Selection

To obtain the ground truth of training dataset, we create a 100-node pseudo-honeypot network with attributes randomly selected from Table I and run it for 300 hours (from March 10 to March $25, 2018$) to gather tweets. We use the approaches (i.e., suspended account, cluster-based, and rule-based methods) mentioned in Section IV-B to roughly label this dataset on September $1, 2018$. It then takes two weeks to manually check the roughly labeled dataset for refinement to obtain a reliable dataset for training use. The spams, spammers, and their percentages that are finally refined by different methods

TABLE IV
THE ACCURACY, PRECISION, RECALL, AND FALSE POSITIVE RATE OF MACHINE LEARNING CLASSIFIERS ON GROUND TRUTH DATA.

Method	Accuracy	Precision	Recall	False Positive
DT	0.912	0.801	0.788	0.249
kNN	0.955	0.813	0.869	0.193
SVM	0.877	0.912	0.762	0.026
EGB	0.965	0.952	0.811	0.033
RF	0.962	0.974	0.744	0.002

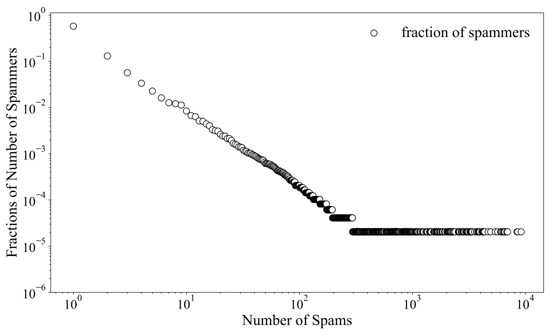


Fig. 2. The fractions of spammers respects to the number spam messages.

are listed in Table III. In the end, we have labeled a total of $6,096$ spammers and $19,296$ spams, which involve 8.30% of the total user accounts and 11.94% of the total tweets, respectively, as the ground truth dataset.

To determine the best suitable machine learning classifier for use in our pseudo-honeypot detector, we examine various algorithms, including Decision Tree (DT), k-Nearest Neighbors (kNN), Support Vector Machine (SVM), Extreme Gradient Boosting (EGB), and Random Forest (RF). The 10-fold cross-validation is used to show their accuracy, precision, recall, and false positive levels, with the results shown in Table IV. Results show that DT, kNN, SVM, EGB, and RF achieve the precision levels of 0.801, 0.813, 0.877, 0.952, and 0.974, respectively. The false positive rates are 0.249, 0.193, 0.026, 0.033, and 0.002, respectively. From these 10-fold cross-validation results, we conclude that RF outperforms other classifiers. Hence, we equip RF as the classifier in our pseudo-honeypot detector for the following experiments. In our experiments, RF is configured with 70 trees as estimators while each tree has a maximum depth of 700.

D. *Effectiveness of Pseudo-honeypot Attributes*

We use the pseudo-honeypot detector to perform classification on all 700-hour collected tweets. There are a total of $1,208,375$ tweets that are identified as spams and all remaining are classified as non-spams. The $1,208,375$ spams are associated with $50,966$ unique accounts, so we have classified a total of $50,966$ spammers. Table V shows the top 10 attributes that capture the most number of spammers. Figure 2 shows the fraction of spammers respecting to the number of spam messages. From this figure, we can see 70.26% of spammers post only one spam message while less than 0.03% of spammers post more than 10 spam messages.

TABLE V
THE TOTAL NUMBER OF COLLECTED TWEETS AND CAPTURED SPAMS AND SPAMMERS UNDER THE TOP 10 ATTRIBUTES.

Index	Attributes	Tweets	Spams	Spammers
1	Average of lists	1025330	112555	40662
2	Lists count	489332	71999	20940
3	Friends&followers	383111	66466	15882
4	Followers count	296273	69648	15155
5	Favorites count	326865	90641	13519
6	Trending up	385977	140877	13314
7	Friends count	220175	33879	11308
8	Hashtag: Social	219908	20701	10444
9	Hashtag: General	145934	40889	9400
10	Popular tweets	305773	122844	9336

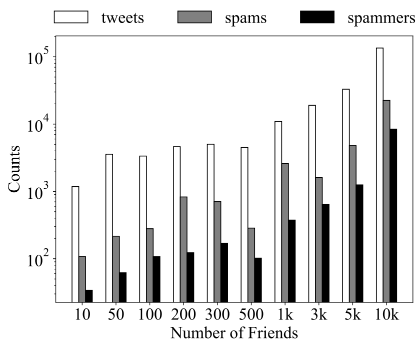
We next give a deep observation of the number of spams (or spammers) captured under various attributes.

Profile-based attributes. Figures 3(a) to (d) show the number of collected tweets, classified spams and spammers corresponding to the *friend/follower* associated attributes. From Figure 3(a) to Figure 3(d), we can see accounts that have more friends, more followers, more total number of friends and followers, or low ratios of friends over followers are more likely to attract spammers. This can be easily understood as follows. If an account has more friends and followers, it is likely that more spammers exist in her friends and followers. And also the associated spam messages can be spread to their neighbors. Thus, the spammers are more likely to target these users with large amounts of friends.

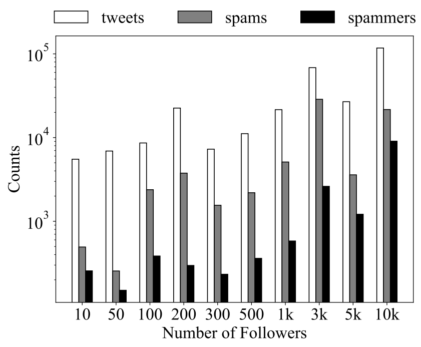
Figure 3(e) shows that accounts with age at around $1,000$ days are more likely to attract spammers. Figures 3(f) to (k) show the collected tweets and the number of classified spams and spammers corresponding to the *list count, favorites count,* and *status count,* respectively. From these figures, we can see the accounts that joining more users groups (Figures 3(f) and (i)), having more favorites (Figures 3(g) and 3(j)), and updating more frequently (Figures 3(h) and 3(k)), have higher probabilities to attract spammers. The reason is that these attributes represent the activities of associated accounts, so if more activities of a user, the more likely such user to be exposed to spammers. As a result, the user is more vulnerable to be attacked.

Hashtag-based attributes. Figure 4 shows the total number of collected tweets, classified spams, and spammers under various hashtag-based attributes. From this figure, we can see *social, general, technology and business* are the top four attributes in our pseudo-honeypot system capturing most spammers (i.e., 10444, 9400, 9251, and 7133, respectively). In this figure, the solid line represents the spammer's ratio (i.e., garnered spammers over total user accounts) associated with each feature. That is, the spammer ratios in the user accounts related to *technology, entertainment, business, and general* are 23.22%, 18.06%. 16.53%, and 12.62% respectively.

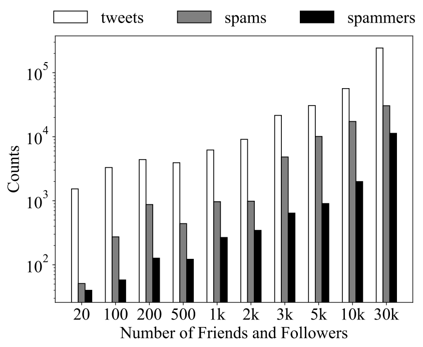
Trending-based attributes. Figure 5 shows the total number of collected tweets and the classified spams and spammers from various trending-based attributes. From this figure, we can see the number of spammers captured by our pseudo-honeypots with the attributes of *trending up, popular, trending*



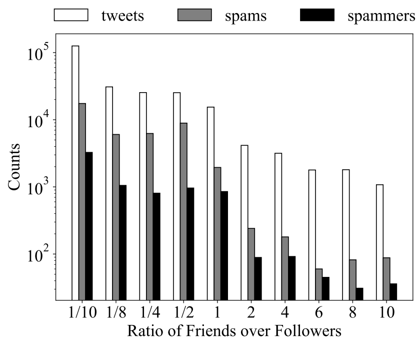
(a) The total number of friends.



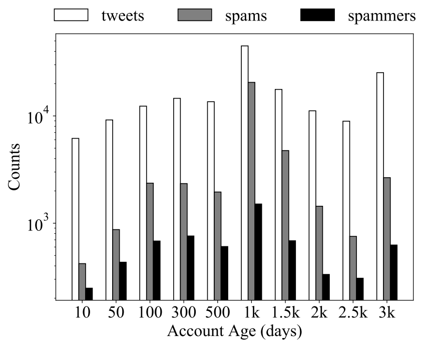
(b) The total number of follwers.



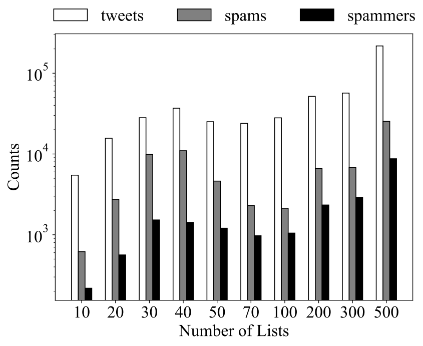
(c) The total number of friends and followers.



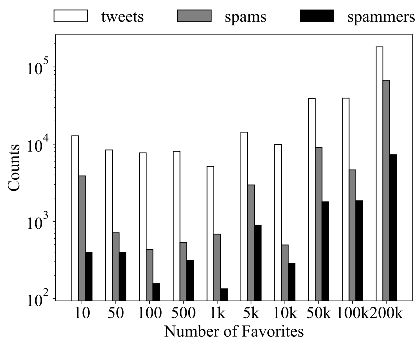
(d) The ratio of friends over follwers.



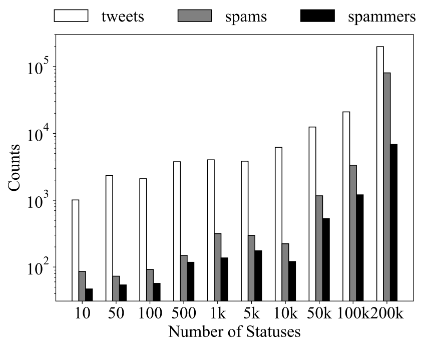
(e) The ages of acounts.



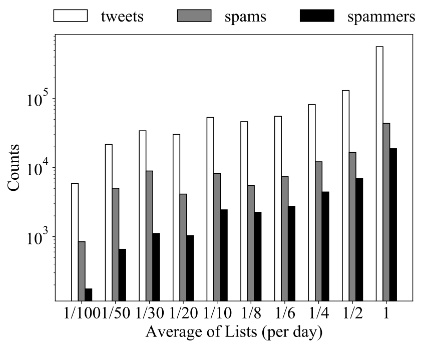
(f) The number of lists.



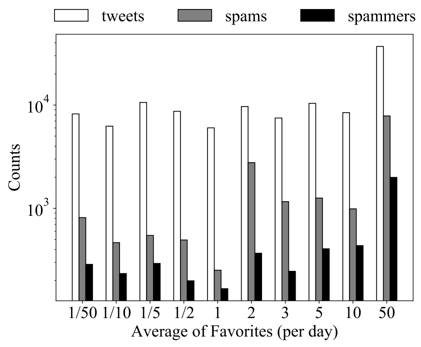
(g) The number of favorites.



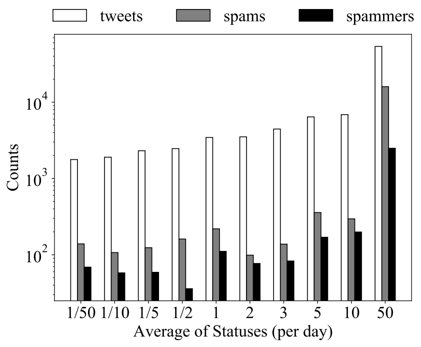
(h) The number of statuses.



(i) The average of lists per day.



(j) The average of favorites per day.



(k) The average of status per day.

Fig. 3. The number of collected tweets, spams, and spammers under various sample values of each attribute.

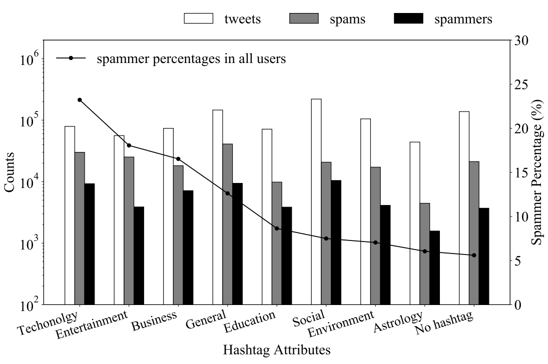


Fig. 4. The number of collected tweets, classified spams and spammers as well as the ratio of spammers over total users in each attribute of Hashtag-based category.

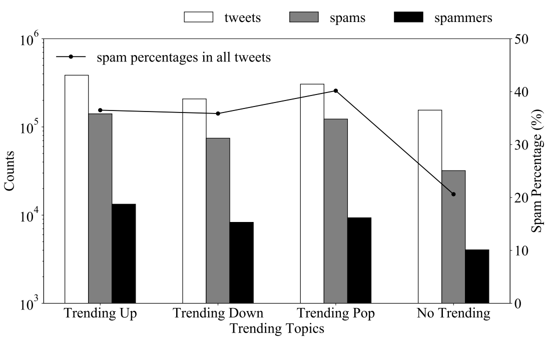


Fig. 5. The number of collected tweets, classified spams and spammers as well as the ratio of spams over tweets in each attribute of Trending-based category.

down, and no trending topic are 13314, 9336, 8292, and 4043, respectively. The spam ratios of these four trending features are 36.50%, 40.17%, 35.87%, and 20.61%, respectively. Thus, the tweets with *trending up* and *popular* attributes have much more potentials of attracting spam messages. The spammer ratios of *trending up, popular,* and *trending down* are all located between 12.4% to 12.6%.

E. Guidelines to build advanced pseudo-honeypot

The results in Section V-D provide a guideline to design a more efficient pseudo-honeypot system by taking into account of attributes that have the highest potentials of capturing spams and spammers. We define a new performance measurement metric *Pseudo-honeypot Garner Efficiency (PGE)* to reflect the effectiveness of each attribute on spammers garnering. The PGE can be expressed as follows:

$$PGE_i = \frac{N_i}{G_i T_i},$$

where N_i denotes the number of spammers garnered by pseudo-honeypots with the attribute i in a total time window T_i and G_i denotes the number of pseudo-honeypot nodes with attribute i. Note here, the switching time of pseudo-honeypot is set to 1 hour. Thus, PGE represents the number of spammers garnered per pseudo-honeypot per hour.

Based on the results in Section V-D, we refine the top 10 sampling attributes that have the highest PGE, which are shown in Table VI. We use such top 10 attributes to build an advanced pseudo-honeypot system, where we select 10 users accounts possessing each attribute. Hence, we have a total of 100 pseudo-honeypot nodes in the system.

TABLE VI
PGEs OF THE TOP 10 SAMPLING ATTRIBUTES.

Rank	Attribute Description	PGE
1	Joining 1 lists per day	2.6894
2	Having 30k friends and followers	1.6150
3	Having 10k followers	1.2943
4	Joining 500 lists	1.2477
5	Having 10k friends	1.2054
6	Having 200k favourites	1.0433
7	Joining 0.5 lists per day	0.9926
8	Having 200k statuses	0.9799
9	Joining 0.25 lists per day	0.6341
10	Friend follower ratio 1:10	0.4667

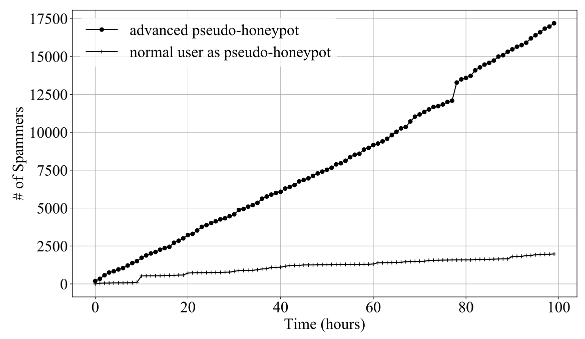


Fig. 6. Spammer captured by advanced pseudo-honeypots and non pseudo-honeypots in 100 hours.

Comparing with non pseudo-honeypot system. For comparison, we randomly select 100 user accounts as one group and name it as *non pseudo-honeypot system*. For both pseudo-honeypot and non pseudo-honeypot systems, we set the switching time as one hour and run a total of 100 hours. Figure 6 shows the amounts of garnered spammers are $17,336$ and $1,850$, respectively, under pseudo-honeypot and *non pseudo-honeypot* systems within 100 hours. Thus, the number of spammers captured under the pseudo-honeypot system is 9.37 times more than that in the *non pseudo-honeypot* system.

Comparison with honeypot-based solutions. We continue to compare the performance of pseudo-honeypot system with the state-of-the-art honeypot-based solutions. As it is difficulty to deploy honeypot system with such network size and associated attributes, we select to compare the results from the prominent research (i.e., Stringhini *et al.* [27], Lee *et al.* [17], Yang *et al.* [38]). Table VII lists the experimental time, the running time, the number of deployed nodes, garnered spams, garnered spammers, and PGE values of our pseudo-honeypot and existing honeypot systems. By comparing PGEs, we can see our method can garner spammers at least 19 times faster than the state-of-art honeypot based solutions.

VI. RELATED WORK

Spammer Detection in Online Social Networks. Extensive research efforts have aimed to identify the spam messages,

TABLE VII
COMPARISON OF PSEUDO-HONEYPOT SYSTEM WITH HONEYPOT-BASED SOLUTIONS.

Honeypot mehtod	**Time**	**Running duration**	**# of honeypots**	**# of spams**	**# of spammers**	**PGE**
Stringhini *et.al.* [27]	2010	11 months	300	–	15,857	0.0067
Lee *et.al.* [17]	2011	7 month	60	–	36,000	0.12
Yang *et.al.* [38]	2014	5 month	96	17,000	1,159	0.0034
Yang *et.al.* [38]'s advanced system	2014	10 days	10	–	–	0.087
Advanced pseudo-honeypot system	2018	100 hours	100	339,553	17,336	1.7336

fake accounts, or compromised accounts. The task of earlier spam message detection [1], [18], [12], [21] focuses on the analysis of the large set of blindly collected contents (e.g., tweets). Those contents are characterized by features or user behavioral patterns extracted for detecting the potential anomalous network activities. The machine learning techniques are commonly leveraged to classify the spam messages. Notably, the effort has been put forth in creating (1) large-scale annotated datasets [24] for hashtag-oriented spam research and (2) large ground truth datasets [7] via machine learning technique in Twitter spam detection.

In regard to fake account identification, the social graph-based approaches are popularly employed to analyze the social relationship among collected accounts. The malicious accounts can be identified by analyzing the graph partition similarity [35] or user ranking [4], [15], [3]. Specifically, *COMPA* [10] was proposed to group users and statistically model the similar sudden changes. *SynchroTrap* was proposed in [5] to classify accounts according to the similarity of user actions in a way that it clustered accounts based on their activities, with those acting similarly at around the same time for a sustained period, as malicious ones. In addition, Yang *et al.* [37] empirically analyzed the cybercriminal ecosystem in Twitter. They conducted an in-depth investigation of inner and outer social relationships and leveraged the criminal account inference algorithm to infer more criminal accounts.

While our work deals with spam message detection (by identifying the online social network spammers), it differs from earlier studies [1], [7], [24], [18], [12], [21], considering an effective mechanism for collecting network contents that are likely to include spam activities, instead of filtering a large amount of network contents to single out potential spam ones. As a result, the proposed work significantly reduces the data processing workload while substantially lifting the probability of spammer capturing.

Honeypot-based Spam Detection. The honeypot is a passive solution that has been applied widely for attracting and trapping spammers in online social networks. With manually setting up, a honeypot is equipped with some specific features that meet spammers' tastes and disguises itself as an ordinary user account. Pertinent approaches [27], [16], [22], [17] address how to deploy the honeypot with some collected features to attract spammers. After capturing target messages, they generate the ground truth data and then design machine learning classifiers to identify the spams/spammers. In [38], Yang *et al.* outlined the reverse engineering strategies to guide honeypot construction. First, honeypots with diverse and fine-grained social behavioral patterns are built. The set of features or behaviors that have high probabilities to garner spammers is then determined through in-depth analyses. Finally, more honeypots with the determined features or behaviors that highly meet the spammer's taste are constructed. Generally, the honeypot-based solutions have the drawbacks of a high deployment cost and limited scalability, since it is time-consuming to set up honeypot accounts manually. Moreover, honeypots are subject to high chances of being recognized by spammers. In sharp contrast to the traditional honeypot solutions [27], [16], [22], [17], our pseudo-honeypot utilizes real user accounts as the carriers, able to take advantage of feature and user diversity. Therefore, it substantially elevates its feature availability, deployment flexibility, network scalability, and system portability.

VII. CONCLUSION

In this paper, we have proposed pseudo-honeypot as a novel method for spammer detection in online social networks. By taking advantage of user diversity and harnessing the normal accounts, the proposed pseudo-honeypot system can substantially improve the deployment flexibility and enrich the attributes availability while largely avoiding being recognized by spammers. We have addressed the challenges associated with the pseudo-honeypot (i.e., transparent to the normal account, pseudo-honeypot selection, and portability), proposing its system design and implementing it in Twitter social network. We outline practical dataset labeling with an aid of the machine-learning technique to arrive at the ground truth of monitored data (as spams and spammers) in order to realize pseudo-honeypot performance evaluation. The results of experimental evaluation demonstrate the proposed pseudo-honeypot system can substantially outperform both the non pseudo-honeypot and the honeypot systems.

Although this work has shown the potential of pseudo-honeypot and its design details in Twitter social network, much work remains open. One direction is to exploit the Twitter spammer drift problem for the long-time spam detection (as we have discussed in Section IV-C). It is important yet challenging to keep tracking of spammers' taste while updating pseudo-honeypot's attributes in the long-time spam detection. Another direction is to design pseudo-honeypot in other types of social networks, such as Facebook, Instagram, and others. This requires specific attribute identification and feature engineering for each social network, according to the characteristics of spammers in the target network. Nonetheless, each customized design for other social network can follow the similar workflow as described in this paper. In addition, some insights are provided as the guideline in this paper for improving pseudo-honeypot performance with more effective attributes in the future.

Acknowledgment

The authors would like to thank the shepherd Marco Vieira and the anonymous reviewers for their fruitful comments. This research was supported in part by Louisiana Board of Regents under Contract Number LEQSF(2018-21)-RD-A24. Any opinion and findings expressed in the paper are those of the authors and do not necessarily reflect the position of their employers or the view of funding agency.

References

[1] F. Benevenuto, G. Magno, T. Rodrigues, and V. Almeida. Detecting spammers on twitter. In *Proceedings of the Conference on Annual Collaboration, Electronic Messaging, Anti-Abuse and Spam (CEAS)*, volume 6, page 12, 2010.

[2] S. Y. Bhat, M. Abulaish, and A. A. Mirza. Spammer classification using ensemble methods over structural social network features. In *Proceedings of the IEEE/WIC/ACM International Joint Conferences on Web Intelligence (WI) and Intelligent Agent Technologies (IAT)*, pages 454–458. IEEE Computer Society, 2014.

[3] Y. Boshmaf, D. Logothetis, G. Siganos, J. Lería, J. Lorenzo, M. Ripeanu, and K. Beznosov. Integro: Leveraging victim prediction for robust fake account detection in osns. In *Proceedings of the NDSS*, volume 15, pages 8–11, 2015.

[4] Q. Cao, M. Sirivianos, X. Yang, and T. Pregueiro. Aiding the detection of fake accounts in large scale social online services. In *Proceedings of the USENIX conference on Networked Systems Design and Implementation*, 2012.

[5] Q. Cao, X. Yang, J. Yu, and C. Palow. Uncovering large groups of active malicious accounts in online social networks. In *Proceedings of the ACM SIGSAC Conference on Computer and Communications Security*, pages 477–488, 2014.

[6] C. Chen, Y. Wang, J. Zhang, Y. Xiang, W. Zhou, and G. Min. Statistical features-based real-time detection of drifted twitter spam. *IEEE Transactions on Information Forensics and Security*, 12(4):914–925, 2017.

[7] C. Chen, J. Zhang, X. Chen, Y. Xiang, and W. Zhou. 6 million spam tweets: A large ground truth for timely twitter spam detection. In *Proceedings of the Conference on IEEE International Conference on Communications (ICC)*, pages 7065–7070, 2015.

[8] W. Chen, C. K. Yeo, C. T. Lau, and B. S. Lee. A study on real-time low-quality content detection on twitter from the users' perspective. *PLOS ONE*, 12(8):1–22, 08 2017.

[9] L. Corporation. Hashtag analytics for your brand, business, product, service, event or blog, 2018.

[10] M. Egele, G. Stringhini, C. Krügel, and G. Vigna. Compa: Detecting compromised accounts on social networks. In *Proceedings of the NDSS*, 2013.

[11] S. Fakhraei, J. Foulds, M. Shashanka, and L. Getoor. Collective spammer detection in evolving multi-relational social networks. In *Proceedings of the ACM SIGKDD international conference on Knowledge Discovery and Data Mining*, pages 1769–1778, 2015.

[12] H. Gao, Y. Chen, K. Lee, D. Palsetia, and A. N. Choudhary. Towards online spam filtering in social networks. In *Proceedings of the NDSS*, volume 12, 2012.

[13] S. Gurajala, J. S. White, B. Hudson, and J. N. Matthews. Fake twitter accounts: Profile characteristics obtained using an activity-based pattern detection approach. In *Proceedings of the International Conference on Social Media & Society*, 2015.

[14] M. Isaac and S. Ember. For election day influence, twitter ruled social media. *The New York Times. Retrieved from https://www.nytimes.com/2016/11/09/technology/for-election-day-chatter-twitterruled-social-media. html*, 2016.

[15] J. Jia, B. Wang, and N. Z. Gong. Random walk based fake account detection in online social networks. In *Proceedings of Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN)*, pages 273–284, 2017.

[16] K. Lee, J. Caverlee, and S. Webb. Uncovering social spammers: Social honeypots + machine learning. In *Proceedings of the ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 435–442, 2010.

[17] K. Lee, B. D. Eoff, and J. Caverlee. Seven months with the devils: A long-term study of content polluters on twitter. In *Proceedings of the ICWSM*, pages 185–192, 2011.

[18] S. Lee and J. Kim. Warningbird: Detecting suspicious urls in twitter stream. In *Proceedings of the NDSS*, volume 12, 2012.

[19] Y. Li, R. Krishnamurthy, S. Raghavan, S. Vaithyanathan, and H. Jagadish. Regular expression learning for information extraction. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, pages 21–30, 2008.

[20] A. Makkar and S. Goel. Spammer classification using ensemble methods over content-based features. In *Proceedings of Conference on Soft Computing for Problem Solving*, pages 1–9, 2017.

[21] Z. Miller, B. Dickinson, W. Deitrick, W. Hu, and A. H. Wang. Twitter spammer detection using data stream clustering. *Information Sciences*, 260:64–73, 2014.

[22] M. Nisrine et al. A security approach for social networks based on honeypots. In *IEEE International Colloquium on Information Science and Technology (CiSt)*, pages 638–643, 2016.

[23] P. Prasse, C. Sawade, N. Landwehr, and T. Scheffer. Learning to identify concise regular expressions that describe email campaigns. *The Journal of Machine Learning Research*, 16(1):3687–3720, 2015.

[24] S. Sedhai and A. Sun. Hspam14: A collection of 14 million tweets for hashtag-oriented spam research. In *Proceedings of the ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 223–232, 2015.

[25] S. Sedhai and A. Sun. Semi-supervised spam detection in twitter stream. *IEEE Transactions on Computational Social Systems*, 5(1):169–175, 2018.

[26] A. Shrivastava and P. Li. In defense of minhash over simhash. In *Artificial Intelligence and Statistics*, pages 886–894, 2014.

[27] G. Stringhini, C. Kruegel, and G. Vigna. Detecting spammers on social networks. In *Proceedings of the Annual Computer Security Applications Conference (ACSAC)*, 2010.

[28] D. Suryawanshi. *Image recognition: Detection of nearly duplicate images*. PhD thesis, California State University Channel Islands, 2018.

[29] K. Thomas, C. Grier, D. Song, and V. Paxson. Suspended accounts in retrospect: an analysis of twitter spam. In *Proceedings of the ACM SIGCOMM conference on Internet Measurement*, pages 243–258, 2011.

[30] K. Thomas, D. McCoy, C. Grier, A. Kolcz, and V. Paxson. Trafficking fraudulent accounts: The role of the underground market in twitter spam and abuse. In *Proceedings of the USENIX Security Symposium*, pages 195–210, 2013.

[31] Twitter. Update on twitter's review of the 2016 u.s. election. https://blog.twitter.com/official/en_us/topics/company/2018/2016-election-update.html, 2018.

[32] Twitter. Developer agreement and policy. https://developer.twitter.com/en/developer-terms/agreement-and-policy.html, 2019.

[33] Twitter. The twitter rules. https://help.twitter.com/en/rules-and-policies/twitter-rules, 2019.

[34] C. Wagner, S. Mitter, C. Körner, and M. Strohmaier. When social bots attack: Modeling susceptibility of users in online social networks. *Making Sense of Microposts*, 2(4):1951–1959, 2012.

[35] G. Wang, T. Konolige, C. Wilson, X. Wang, H. Zheng, and B. Y. Zhao. You are how you click: Clickstream analysis for sybil detection. In *USENIX Security Symposium*, volume 9, 2013.

[36] Y. Xie, F. Yu, K. Achan, R. Panigrahy, G. Hulten, and I. Osipkov. Spamming botnets: Signatures and characteristics. *Proceedings of the ACM SIGCOMM conference on Computer Communication Review*, 38(4):171–182, 2008.

[37] C. Yang, R. Harkreader, J. Zhang, S. Shin, and G. Gu. Analyzing spammers' social networks for fun and profit: A case study of cyber criminal ecosystem on twitter. In *Proceedings of the 21st international conference on World Wide Web*, pages 71–80, 2012.

[38] C. Yang, J. Zhang, and G. Gu. A taste of tweets: Reverse engineering twitter spammers. In *Proceedings of the Annual Computer Security Applications Conference (ACSAC)*, pages 86–95, 2014.

[39] X. Zheng, Z. Zeng, Z. Chen, Y. Yu, and C. Rong. Detecting spammers on social networks. *Neurocomputing*, 159:27–34, 2015.