Impact of Time on Detecting Spammers in Twitter

Mahdi Washha
IRIT Laboratory
University of Toulouse
Toulouse, France
mahdi.washha@irit.fr

Aziz Qaroush
Birzeit University
Birzeit, Palestine
aqaroush@birzeit.edu

Florence Sedes
IRIT Laboratory
University of Toulouse
Toulouse, France
florence.Sedes@irit.fr

ABSTRACT

Twitter is one of the most popular microblogging social systems, which provides a set of distinctive posting services operating in real time manner. The flexibility in using these services has attracted unethical individuals, so-called "spammers", aiming at spreading malicious, phishing, and misleading information over the network. The spamming behavior results non-ignorable problems related to real-time search and user's privacy. Although of Twitter's community attempts in breaking up the spam phenomenon, researchers have dived far in fighting spammers through automating the detection process. To do so, they leverage the features concept combined with machine learning methods. However, the existing features are not effective enough to adapt the spammers' tactics due to ease of manipulation, including the graph features which are not suitable for real-time filtering.

In this paper, we introduce the design of novel features suited for real-time filtering. The features are distributed between robust statistical features considering explicitly the time of posting tweets and creation date of user's account, and behavioral features which catch any potential posting behavior similarity between different instances (e.g. hashtags) in the user's tweets. The experimental results show that our new features are able to classify correctly the majority of spammers with an accuracy higher than 93% when using Random Forest learning algorithm, outperforming the accuracy of the state of features by about 6%.

Keywords

Twitter, Social Networks, Spam, Legitimate Users, Machine Learning, Honeypot, Time

1. INTRODUCTION

Twitter[1] has recently emerged as one of the most popular microblogging social networks, which allows users to publish, share, and discuss about everything such as social events, news, and jokes, through a messaging mechanism allowing 140 characters maximum. Statistics states that, in November 2015, the number of active users that use Twitter monthly is about 320 millions with an average of 500 million tweets published per day [1]. This popularity is because of the set of services that Twitter platform provides for the users, summarized in: (i) delivering in real time manner the users' posts (tweets), which allows their followers to spread the posts even more by re-tweeting them; (ii) inserting URLs pointing to an external resource such as web pages and images; (iii) permitting users to add hashtags into theirs posts to help other users to get the relevant posts; (iv) retrieving tweets through real time search service, letting Twitter, Google, and other memetracking services to find out what is currently happening in the world in minimum acceptable delay. To get much insight in the strengths of such services, Twitter has been well intended to be adopted as an alert system in the context of crisis management such as Tsunami disaster [2].

The power of spreading information mechanism and the absence of effective restrictions on posting action in Twitter have attracted some unethical individuals, so-called "spammers". Spammers misuse these services through publishing and spreading misleading information. Indeed, a wide range of goals drive spammers to perform spamming behavior, ranging from spreading advertisement to generate sales, and ending by disseminating pornography, viruses and phishing. As a consequence, spreading of spam material affects negatively in different areas, summarized in [3]: (i) polluting real-time search; (ii) interfering on statistics introduced by tweet mining tools; (iii) consuming extra resources from both systems and humans; (iv) degrading the performance of search engines that exploit explicitly social signals in their work; and (v) violating users' privacy that occurs because of viruses and phishing methods.

Obviously, the negative impacts of spam are not ignorable and are getting rapidly increasing everyday on social networks. To address the spam issue, a considerable set of methods has been proposed to reduce and eliminate the spam problem. Most existing researches [4, 3, 5, 6, 7, 8, 9, 10, 11, 12, 13] are dedicated for detecting Twitter spam accounts individually or spam campaigns. The works rely on extracting a set of features from the available information associated with the user account (e.g. number of followers, account age), features related to the content of the posts (e.g. number of URLs, number of characters), or features associated to the graph theory (e.g. local clustering). The other less used approach [14, 3] detects spam tweets instead

[1] https://twitter.com/

BDA 2016, 15 au 18 Novembre, Poitiers, France.

of spam accounts. However, this tweet level approach is not effective as not enough information available in the individual tweet itself. Moreover, the existing attempts that based on spam account detection have critical limitations and major drawbacks. One significant drawback is derived from focusing on using features that are easy to manipulate, leading to avoid detection when using these features. As a motivating example, the number of followers (i.e. the accounts that follow a user) is one of many features used mainly in detecting spammers with considering the small number of followers tend to be spammer more than a normal user (legitimate). However, the followers number is easy to be increased by spammer and that by creating a huge number of recent accounts with letting each account follow each other, resulting a set of recently created accounts having high number of followers. Similarly, the number of words in the tweets is considered as a feature to discriminate between spammer and legitimate user. Unfortunately, most of the features designed in the literature exploited in detecting spammers are completely similar to the given examples in regards of ease of manipulation. Also, the high time computation of some features (e.g. graph based features) makes them not appropriate for real time filtering, raising a strong motivation to conduct this work.

By examining the information that can be gathered from social networks in which they cannot be changed overtime and also not optional for the user, we found that the *time* of either creation date of the account or the posting time is the only property that the user cannot modify. Intuitively, it is not modifiable since the servers of the social network platform store it one time when the user takes the action. The *time* property is the biggest enemy against spammers since it is the only feature that can capture the spammers' behavior overtime (e.g. tweeting speed). Hence, in this paper, we introduce the design of 20 novel features suited for real-time filtering, distributed between robust statistical features leveraging explicitly the *time* property, and behavioral features that catch any potential posting behavior similarity between different instances (e.g. hashtags) available in tweets. In validating the features robustness, we first crawled a data-set from Twitter, containing 7,189 users labeled manually with almost 300k tweets. Then, various classification algorithms are investigated to identify spammers from legitimate users. Based on the experiments conducted on the crawled data-set, we found that our new features are able to classify correctly the majority of spammers with a detection rate higher than 93% when applying Random Forest as a classification algorithm which produced the best results against other classifiers. Besides, we implemented the state of art features, estimated at 70 features, that have been widely used in identifying individual spammers and then experimenting them on the crawled data-set. The results show that our features have outperformed the existing features by about 6% of detection rate. With such high detection rate, our features are adoptable in real time spammer detection and filtering because of their simplicity in regards of time complexity, as well as it validates our hypothesis about *time* property.

The rest of the paper is organized as follows. Section 2 presents the Twitter rules followed in fighting spammers as well as the related work in Twitter spam research area. Section 3 shows the formalization and the definition of the problem we study, in addition to the design of the proposed features. Section 4 describes the procedure adopted in crawling our data-set with a detailed description, including some statistics about the data-set. Section 5 evaluates our features proposed in this work using machine learning algorithms, including a deep comparison with the state-of-art features. At last, section 6 concludes the paper with giving some insights about future direction in spam detection.

2. BACKGROUND AND RELATED WORK

Spammers exploit three services provided by Twitter in performing their spamming behavior: (i) *URL*; (ii) *Hashtag*; (iii) *Mention*. Since the tweet size is limited, shorten URL services (e.g. Bitly and TinyURL) are permitted in Twitter to convert the long URL to small one. Spammers abuse this service through posting spam websites with shorting first the desired URL to hide the domain. As a textual feature, *hashtag* is widely used in social networks as a service to group tweets by their topic to facilitate the search process. Spammers also misuse this service by attaching *hashtag* in their tweets, that may contain URLs, to step-up the chance of being searched by users. The *Mention* service provides a mechanism to send a direct message to a particular user, through using the @ symbol followed by the screen name of the target user. Differently from *URLs* and *hashtag*, spammers misuse this service to send their tweets for a defined target of users. Besides these services, Twitter provides APIs for developers to be used in their third party applications. Spammers exploit this strong service as an opportunity to automate their spamming behavior, in spite of the constraints imposed on the API calls that can be requested in 15 minutes time window.

In order to fight spammers, Twitter allows users to report spammer simply by clicking the "Report: they are posting spam" option available on the profile page of the spammer. Users, alternatively, can report spammers by posting a tweet mention @*spam* account with the spammer's screen name [15]. Once an account is reported, the administrators of Twitter will review manually this account to make the suspension decision. However, such a way in combating spammer requires a considerable effort from both users and administrators, rather than the fake reporters who send reports about legitimate users. In addition to the manual reporting mechanism, Twitter has released some rules to reduce the spamming problem, with suspending permanently the accounts that violate those rules [16]. The Twitter's rules define the spamming behavior at three levels: (i) behavioral; (ii) social relationship; (iii) tweet contents. In the behavioral level, intensive automation such as posting tweets without any human participation is completely forbidden. In the same level, using multiple accounts to post a duplicated tweets is also categorized as a spamming behavior. At the social relationship, following a large number of users in a short time may subject the user's account to permanent suspension, or having high number of friends (followings) compared to the number of followers. In the content level, it is disallowed users to post malicious URL or any content containing spam. Also, using large number of mentions and hashtags are prohibited.

Twitter's rules are easy to avoid by spammers. For example, spammers can coordinate multiple accounts with distributing the desired workload among these accounts to mislead the detection process since individual separated accounts tend to exhibit invisible spam behavior. Beyond

that, using multiple accounts can spread spam to more users, which is one of the major spammers' goals. Another critical limitation existing is that the spam accounts detection or the spam content filtering process is not achieved in real time way, making these solutions not effective for real time Twitter based services. These flaws have motivated researches to propose more powerful methods that can be suitable for real time filtering. Mainly, Twitter spam detection approaches can be categorized under two different types based on the automation detection level, including machine learning level as a fully automated approach, and social honeypot as a manual approach requiring human interaction.

Machine Learning Approach. Most of the existing studies followed the learning approach since it automates the detection process and it keeps away the human participation as well. Researchers in this approach built their methods through employing three levels of detection distributed between tweet-level detection, user-level detection, and campaign-level detection. At the tweet level, Martinez-Romo and Araujo [14] identified spam tweet through applying probabilistic language models to determine the topic of the considered tweet. Then, the decision about the tweet is made through calculating the divergence of the identified topic with respect to the relevant trending topic by using language models of the considered topics. At the same level, Benevenuto [3] has identified spam tweet via extracting a set of features such as number of words and number of characters from each tweet individually with employing then SVM learning algorithm on manually created data-set to get a binary classifier. Besides the simplicity of using the tweet level in regards of time complexity, it is an inefficient solution to identify spammers, since one tweet does not provide "informative" information to robustly distinguish the spamming behavior from the legitimate users' behavior. This limitation of tweet level has enforced the researchers to turn their attentions to user-level detection. For instance, Yardi et al [17] studied several features related to the account age, the tweeting behavior, and the network structure for the spammers and the legitimate users, with highlighting that these features are not enough to classify spammers correctly because of the considerable overlapping between these features. More deeply, the work accomplished in [4, 3, 6, 7] focused on extracting more of the account features including the account age, number of friends, number of followers, similarity between tweets posted or re-tweeted by the user, ratio of URLs in tweets. A dedicated study has been performed on detecting spam URLs based on analyzing the URLs' behavior, instead of relying on traditional black-listing filters and on analyzing the content of the landing page [10]. For each URL, fifteen click and posting-based features have been exploited, then machine learning algorithms have been employed to identify the type of URLs. However, as mentioned previously, such features are easy to avoid and manipulate by spammers. For instance, the behavior analysis of the URLs can be fooled by automating the clicking action on the desired posted URLs through using different URLs pointing to the same spam content.

The ease of manipulation in the user-level features by spammers gives a reasonable motivation to obtain much more complex features using graph metrics. For example, the authors of [8, 18] studied the relation between users through some graph metrics to measure three features, including the node betweenness, bi-directional relation ratio, and local clustering. Song, Lee, and Kim [19] examined some relational features such as the connectivity and the distance between a message sender and a message receiver. The authors concluded that the messages that comes from distance more than two have high probability to be spam. Although of the high detection rate when using features obtained by graph metrics, the extraction of such features is not suitable for real time filtering since it is computationally intensive and it requires the complete Twitter network graph to get accurate results. At the campaign level, Chu et al. [11] detected spam campaign through clustering users' accounts based on the URLs retrieved from their posted tweets, and then a set of features are extracted from the clustered accounts to be incorporated in identifying spam campaign via employing machine learning algorithms. Chu, Gianvecchio, Wang, and Jajodia [20] proposed a classification model trained over a set of accounts having a size of 500k to observe the difference among bot, human and cyborg with taking into account, tweet content, and tweeting behavior. Indeed, working at the campaign level are effective to detect big campaigns only. Even more, spammers nowadays are smart enough in designing spam campaigns such that it is difficult to detect the correlations between accounts that belong to the same campaign.

In the same context of machine learning approach, however, without doing any contributions at features level, there are some studies that addressed the spamming problem from the learning approach point of view. The motivation behind that view returns to the fast evolving of spammers' pattern, and thus the traditional batch-mode learning algorithms (e.g. SVM) cannot quickly respond to new spamming patterns since it requires long time to rebuild the classification models. In [21, 22], the authors presented an optimization framework that considers tweets content and the basic network information to detect social spammers using efficient online learning approach. However, the main challenge of this approach is in finding features suitable for real time filtering as well as they cannot be fooled by spammers.

Honeypot Approach. Social honeypot is defined as an information system resource that can monitor spammers' behavior by logging their information such as profiles' information and any available content. Such monitoring is accomplished by first creating multiple accounts on the targeted social network and then these accounts are set to wait for spammers to establish a connection with them. Once a user falls in the accounts' trap, the user's profile is forwarded to manual annotation by honeypot's administrator or to be exploited directly in training classifier used in real time filtering, with considering blindly the user's profile as social spammer [5]. Although of effectiveness this approach in detecting accurately spammers because of the manual annotation; however, there is no major difference between the Twitter's mechanism in fighting spammers and social honeypot approach. Both of them require administration control to take the final decision about the users' accounts that downed in the honeypot trap; rather than classifying blindly all users as spammers leads to have high false positive rate.

Our motivations are derived from the fact that performing real time spam filtering in an automated way requires simple robust features in regards to time complexity. Thus, we turn our attention to improve the robustness of the features that are used at user-level detection only. As an intuition behind this proposition, user-level features can be involved to detect

spam campaigns as well as individual spammer detection.

3. FEATURES DESIGN

This section introduces notations, definitions, and formalization of the problem we study. Then, we present the design and the formalization of the features that will be exploited in distinguishing spammers from legitimate users, distributed between user, content, and graph features. To the best of our knowledge, the design of the features proposed in this section are novel. We highlight the difference between our features and the available ones in the state of art, to show their novelty.

3.1 Notations and Problem Definition

Let $G = (V, E)$ be a directed graph representing Twitter social network, where the node set $V = \{v_i | \ 0 < i \leq n\}$ represents the network's users (or accounts), n is the number of users, and the edges set E reflects the following relation direction property between users. For example, $e_{ij} = 1$ means that the user i follows user j, while the opposite is *not* true. Each user registered in Twitter, $v_i \in V$, can be modeled by 6-tuple $< Tweets, Followers, Followees, Verified, Age, Bio >$ where each element inside the tuple is described as follows:

Tweets: We consider the tweets that the user v_i has posted them as a finite list, defined as $Tweets = \{t_j | \ 0 < j \leq k\}$ where t_j represents a tweet object and k is the number of tweets that the user v_i has posted them on his account. As each tweet holds various information, we model each tweet by 6-tuple $< Time, Retweeters, Hashtags, URLs, Mentions, Words >$, where $Time$ is the posting date of the tweet represented in seconds time unit computed since 1970/1/1 as a date reference, $Retweeters \subseteq V$ is a finite set corresponding to the users that have re-tweeted the considered tweet. $Hashtags$ is a finite set containing all hashtags available in the tweet extracted through searching for the words that start by # symbol, $URLs$ represents also a finite set of all URLs posted in the tweet, $Mentions \subseteq V$ is a set of users that mentioned in the tweet extracted by looking for the words that start by @ symbol, and $Words$ is a finite set containing the words that posted in the tweet such that $Words \cap Hashtags \cap Mentions \cap URLs = \emptyset$.

Followers: Each user might be followed by subscribed users. We define those users who follow the user v_i as $Followers = \{v_j | v_j \in V, e_{ji} \in E\}$.

Followees: Users can follow any user they want as long as the desired user is not protected. We define the users followed by the user v_i as $Followees = \{v_j | v_j \in V, e_{ij} \in E\}$.

Verified: The accounts of the famous characters (e.g. athletes) or the well known organizations like BBC take special treatment from Twitter by verifying their accounts manually with adding special feasible symbol on their accounts. We model this property as a boolean attribute defined as $Verified \in \{True, False\}$.

Age: The creation date of each account is registered one time on Twitter's servers without allowing the user to change it in the future. We compute the age of an account in a unit of days through calculating the difference between the current time date ($Time_{now}$) and the creation date of the account ($Time_{creation}$), defined formally as $Age = Time_{now} - Time_{creation}$.

Bio: For each account, an empty field is given to write brief description about the account. We model the available information inside the description field by 3-tuple $< Hashtags, URLs, Words >$, where the definition of $Hashtags$, $URLs$, and $Words$ is similar to the corresponding one used in defining tweet element.

HashTags Similarity(HTS) (H_1, H_2): Given two sets of hashtags, we measure the similarity between them using Jaccard similarity coefficient [23] defined as $HTS(H_1, H_2) = \frac{|H_1 \cap H_2|}{|H_1 \cup H_2|}$.

URLs Similarity(URLsS) (U_1, U_2): Given two sets of URLs, we measure the similarity between them using Jaccard similarity coefficient defined as $URLsS(U_1, U_2) = \frac{|U_1 \cap U_2|}{|U_1 \cup U_2|}$.

Kullback–Leibler Divergence(KLD) (W_1, W_2). Given two sets of words with the probability distribution of the word occurrence of each set P_{W_1}, P_{W_2}, we measure the similarity between the sets using KullbackâĂŞLeibler Divergence [24] method, computed as:

$$KLD(W_1, W_2) = \sum_{w \in W_1} P_{W_1}(w) \log \frac{P_{W_1}(w)}{P_{W_2}(w)} \qquad (1)$$

where is $P_{W_\bullet}(w)$ is the occurrence probability of the word w.

Weighting Function(WF) (t). For a given value $t \in \mathbb{R}$, we use a non-linear function f selected from a set of functions to transform t to a new value, defined as $WF(t) = f(t)$ where $f \in \{\frac{\lambda}{t}, \exp^{\frac{t*t}{\lambda}}, \exp^{-\frac{t*t}{\lambda}}, 1 - \exp^{-\frac{t*t}{\lambda}}\}, \lambda \in \mathbb{R}$. The selection of f and λ is determined through an optimization process described in section 5 where this process is applied separately on each feature that leverages explicitly WF. This family of functions can boost-up the weights in a non-linear way.

With the presented notations and definitions, our main problem is to detect social spammers on Twitter network. Formally, given a training set $\{(v_i, y_i)\}_{i=1,\ldots,N}$ of N instances consisting of an user v_i associated with a class label $y_i \in \{spammer, legitimate\ user\}$, and given a set of M user features $X = \{x_j | 0 < i \leq M, x_j \in R\}$ extracted from the training set, the problem is turned to learn or build a binary classification model y using the given training set such that it takes user's features X as an input and predicts the class label of the user (Twitter account) as an output, defined as $y : X \rightarrow \{spammer, legitimate\ user\}$.

One contribution of this paper is the definition and the formalization of the features set X . Thus, in the followings, we show the design and the formalization of the features using the notations introduced.

3.2 User Features

Followers and Followees: The number of followers, number of followees, ratio of followers to followees, number and percentage of bi-directional following users are used widely in the literature [4, 3, 6, 8] as features distinguishing between spammers and legitimate users. When a user has small number of followers, low ratio of followers to followees, and low percentage of bi-directional following users, they give an indication that the user has high probability for being spammer or spam account. However, spammers can easily step up these values by creating multiple accounts simultaneously with letting each account to follow each other, resulting a network of spam accounts having properties of legitimate users' accounts. Since those accounts are recently created and almost in the same period, the age of the created

accounts can be leveraged to handle such a phenomenon, since the account's age is unchangeable compared to the other features like number of followers and followees. Thus, we propose three measures as new features which utilize the account's age. Given a user (or account) v and given a set of $users$ (e.g. followers, followees), we compute the mean of the differences between the ages of the $users$ set and the v's user age, as first feature. As spammers tend to create recent accounts, the small value of age mean difference gives an indication that the user v has significant probability for being spammer. In the second feature, we measure the variance of the difference between the users' set ages and the v's user age. The low value of variance means that the users in the given set and the user v have been created in same period. At last, instead of computing the size of $users$ such as the number of followers which treats users uniformly, we modify the computation of the size through involving explicitly the user's age as a weighting factor with giving high weights for old accounts. The low value of this feature gives a strong indication that the users in the given set might be spammers. We model mathematically the mean, $Mean(v, users)$, variance, $Var(v, users)$, and time weighted users, $TWU(users)$, as follows:

$$Mean(v, users) = \frac{\sum_{u \in users} (u.Age - v.Age)}{|users|} \quad (2)$$

$$Var(v, users) = \frac{\sum_{u \in users} (u.Age - v.Age - Mean(v, users))^2}{|users|} \quad (3)$$

$$TWU(users) = \sum_{u \in users} WF(u.Age) \quad (4)$$

We apply these time features on three different sets of users: (i) $followers = v.Followers$; (ii) $followees = \{g | g \in v.Followees \wedge v.Verfied = False\}$; (iii) $bi-directional = \{g | g \in v.Followers \wedge g \in v.Followees\}$, resulting 9 features in total from the above three equations. The definition of the followers and the bi-directional set is straightforward, while for the followees set we consider only on the accounts that are not verified. We exclude the verified accounts since the Twitter's user has a full control to follow unlimited number of users regardless whether those accounts are verified or not, and thus spammers tend to follow huge number of verified accounts to look like as a legitimate user.

Profile Description (Bio): Two features are extracted from this property used in the literature [3, 11]: (i) checking whether the description containing spam words; and (ii) testing if the existing URLs are blacklisted or not. Spammers can simply avoid the spam words features by using alternative words. Also, the blacklisting method is avoidable with much more cost than avoiding spam words, that through putting new URLs in the description field, having a new not blacklisted domains. Beyond these two features, in spam campaign especially and based on our observations, spammers use almost the same description words and URLs. Thus, we model this spamming behavior by computing the user's profile description similarity with other users. As twitter provides a search API service for users, hashtags, words in tweets, and URLs, we employ such an API to search for the users that have potential similar description. Instead of searching for the exact profile description, which might return empty set of matched users, we search for each word in the description field. Formally, let the function $search(text) \subseteq V$ returns the users that contain a given $text$ inside the profile description field. For a considered user v, the potential set of similar users is given as $sim(v) = \{m | w \in v.Description.Words \wedge m \in search(w)\}$. Then, we use the set returned to compute the description similarity with taking into account the similarity of hashtags and URLs as well. Since spammers tend to behave like a legitimate user to avoid detection through filling their descriptions from legitimate users' profiles, we include explicitly the users' age in the computation as a weighting factor. Therefore, description similarity is weighted based on the age difference between user v and the other potential users, defined formally as:

$$PDS(v) = \sum_{u \in sim(v)} WF(|v.Age - u.Age|) * D\_Sim(v, u)$$
$$where$$
$$D\_Sim(v, u) = KLD(v.Bio.Words, u.Bio.Words) + URLsS(v.Bio.URLs, u.Bio.URLs) + HTS(v.Bio.Hashtags, u.Bio.Hashtags) \quad (5)$$

3.3 Content Features

Posting Behavior. The state of art features are statistical ones more than being behavioral, since the existing behavioral features such as the maximum time between two consecutive tweets are not too discriminative [3]. As an intuitive fact, spammers usually follow a particular defined and systematic pattern in posting tweets, while legitimate users have kind of randomness in posting since their mood are changing overtime. Moreover, spammers cannot adopt the random approach in posting their tweets to behave as legitimate user because such an approach cannot maximize the spammer's benefits. Thus, we model the posting behavioral feature through a generic temporal behavior detection framework applicable on different tweeting services, including hashtags, URLs, and mentions. For each tweeting service, the framework measures the similarity of posting behaviors between all possible unique instances of a service available in the considered user's tweets (i.e. posting behavior similarity of different hashtags). As an illustrative example, for the hashtag service, suppose we have three hashtags instances ($\#h1$, $\#h2$, and $\#h3$) found in a given user's tweets and for each hashtag the probability distributions of the posting time is given, the framework measures the similarity of posting behavior of all possible pairs of the three hashtags using their posting probability distributions drawn overtime.

In a formal way, let I_s represents a set of all unique instances available in the user's v tweets and posted by a tweeting service s. Also, let P_i and $i \in I_s$ represents the probability distribution of posting time for an instance. Since the probability distribution of the posting time can be viewed as time shifted signal, we adopt the correlation[2] method to measure the maximum available similarity between all in-

[2]https://en.wikipedia.org/wiki/Cross-correlation

stances, defined as:

$$IS_v(I_s) = \frac{\sum_{i1 \in I_s} Area(P_{i1} \star P_{i2})}{|I_s| * Area(P_{i_{max}} \star P_{i_{max}})}$$
$$i2 = \arg\max_{i3 \in I_s \cap i3 \neq i1} Area(P_{i1} \star P_{i3}) \quad (6)$$
$$i_{max} = \arg\max_{i \in I_s} Area(P_i \star P_i)$$

where $Area(\bullet)$ is a function that computes the area of the signal or the new distribution resulted after applying correlation (i.e. zero area means dissimilar distributions), $P_\bullet \star P_*$ is a cross-correlation between two different distributions, and $P_\bullet \star P_\bullet$ is the correlation for the same distribution known as auto-correlation. The intuition behind $i2$ is to get the instance that has the maximum correlation with the instance $i1$.

The summation of the maximum areas is normalized by the the area of the instance that have the maximum self-similarity multiplied by the number of instances. Thus, the IS value is ranged between zero and one where the zero value means that there are no instances having the same posting behavior, while the one value means that all instances have similar posting behavior.

We leverage the behavioral detection framework to get the similarity posting value as a feature for three tweeting services, including hashtags, URLs, and mentions. The instances set of each service is defined as $I_{hashtags} = \{h | t \in v.Tweets \wedge h \in t.Hashtags\}$, $I_{mentions} = \{m | t \in v.Tweets \wedge m \in t.Mentions\}$, and $I_{urls} = \{r | t \in v.Tweets \wedge r \in t.URLs\}$.

Posting Diversity. Legitimate users and spammers are using sometimes hashtags, URLs, and mentions tweeting services in an intensive way. In such a common scenario, the classical statistical features existing in the literature such as number of URLs, number of hashtags, number of mentions, and percentage of URLs [3, 25] in tweets don't contribute significantly in distinguishing between users' types. Our feature goes beyond the statistical ones through computing the posting diversity for each service separately such as the diversity of the hashtags used in the user's tweets. As an intuition, spammers post their tweets intensively with focusing on a single instance of a tweeting service (e.g. hashtag), while legitimate users have a kind of diversity in posting their tweets without focusing on a particular instance or even tweeting service. By using the same definition used in posting behavior feature extraction part, the diversity of an instances' set I_s for a given user v is computed as

$$PD(v, I_s) = \frac{|I_s|}{|v.Tweets|} \quad (7)$$

The zero value of PD means that the instances set is empty, while the one value means that each instance in I_s is used only one time in the user's v tweets. We apply this feature on four different tweeting services, including hashtags, mentions, URLs, and textual words services, where the instances set definition of each service is similar to the corresponding one defined in the posting behavior part, and the set of textual words is defined as $I_{words} = \{w | t \in v.Tweets \wedge w \in t.Words\}$.

Tweets Similarity. Legitimate users might post a tweet and then after not short period post again the same tweet, resulting duplicated tweets. The related features existing in the literatures checks the similarity between tweets without considering the posting time between them. Intuitively, the user who posts too similar or duplicated tweets in short time has high probability for being spammer, while the opposite is not true. We model such behavior by involving, with the textual similarity, the difference between the posting date of two tweets as a weighting factor. For a given user v, the time weighted tweets similarity, $TWTS(v)$, computed as:

$$TWTS(v) = \frac{\sum_{t1 \in v.Tweets} \sum_{\substack{t2 \in v.Tweets \cap \\ t1.Time > t2.Time}} T\_sim(t1, t2)}{\frac{|v.Tweets| * (|v.Tweets| - 1)}{2}}$$
$$where$$
$$T\_sim(t1, t2) = KLD(t1.Words, t2.Words) * WF(|t1.Time - t2.Time|) \quad (8)$$

The tweet similarity function $T\_sim(t1, t2)$ compares between two tweets through textual words without taking into account the similarity value that come from hashtags and URLs sets. We use only Kullback-Leibler Divergence (KLD) which gives indication on how much two tweets are close together even though the tweets are not exactly duplicated. The value that return by $TWTS$ is ranged to be between zero and one, where zero means that there are no similarity between user's v tweets or the user does not have any tweet posted yet, while the value one means that all tweets posted are duplicated as well as they have been posted in the same short period.

Re-tweeters Diversity. Spammers often collaborate together in performing a spamming behavior. For example, one spammer posts a tweet and other spammers react by re-tweet the posted tweet directly, aiming to propagate tweets over the network as fast as possible. Given that the probability of having legitimate user to re-tweet a spam tweet is low, we model such a behavior through examining the diversity of re-tweeters in the re-tweeted posts. We exploit the biodiversity index[3] measure to get the diversity in the re-tweeters, computed as:

$$RT\_Diversity(v) = \frac{|\{t.Retweeters | t \in v.Tweets\}|}{\sum_{t \in v.Tweets} |t.Retweeters|} \quad (9)$$

where the numerator represents the size of unique re-tweeters set, and the denumerator is the summation of all re-tweeters. Obviously, the $RT\_Diversity$ produces a real number ranged between zero and one where the value one indicates that the re-tweeters of the tweets are completely different, while zero value means that all tweets of the user v are not tweeted yet.

3.4 Graph Features

Local Clustering Coefficient. This graph metric quantifies how much the neighbors (followers and followees) of a user are close to form a clique [11]. It is defined as the proportion of edges between the users within their neighborhood divided by the number of edges that could exist between them. According to the literature intuition [11], spammers tend to have small local clustering value. On the contrary, legitimate users have high value, since the legitimate users' have high probability to make a direct relation-

[3]http://www.coastalwiki.org/wiki/measurements_of_biodiversity

ship with the their followers' and followees' friends, which is not the case of spammers. However, spammers can simply boost up this metric value by letting a considerable number of spam accounts to follow each other. Thus, to overcome this issue, we improve the metric through involving the users' ages as a weighing factor, since the spam accounts often are recently created and close to each other. For a given user v, the time weighted local clustering coefficient, $TWLC(v)$ is computed as follows:

$$\begin{aligned} &TWLC(v) = 2 * \frac{TWU(users(v))}{K_v * (K_v - 1)} \\ &where \\ &users(v) = \{u_i, u_j : u_i, u_j \in N_v, e_{ij} \in E\} \quad , \\ &N_u = u.followers \cup u.followees, \end{aligned} \tag{10}$$

$K_v = |v.followees| + |v.followers|$ is the number of users that the user v has direct relation with them, and TWU is the time weighted users computed by equation 4.

4. DATA-SET DESCRIPTION AND CRAWLING

As the features designed are quite novel and advanced compared to the existing ones used in the literatures, the available crawled data-sets [4, 26, 3, 17, 5, 6, 7, 27, 8, 10, 28, 22, 14, 19, 12] are not suitable to verify the robustness of our features. Crawling individual users' profiles for collecting data-set without considering their followers' and followings' profiles are the main reasons for not using them in our experiments. Besides, the crawling mechanism adopted in the existing researches uses the tweet streaming approach which provides low latency access to 1% of Twitter's global stream of Tweet data, making the probability to collect spam tweet posted by real spammer too low. Thus, we developed our crawler in Java which uses the REST API[4] as a mechanism to run specific real-time searching methods on indexed tweets and users, making the crawling process more deterministic and controlled. In spite of those advantages in REST API, Twitter platform constraints the number of API's calls that can be requested in 15 minutes time window, imposing challenges on having a big annotated data-set. We have reduced the API rate limits problem through creating multiple authenticated accounts scheduled intelligently to switch automatically between accounts when exceeding the limit rate in one account. As the typical majority of users on social networks are legitimate users, using random approach in collecting data is not an efficient way to have a data-set consisting of a considerable number of spammers to conduct deep analysis. Thus, in crawling profiles process, we prepared first a list of spammers using their screen names as an ID. Then, for each spammer in the list we retrieve his profile, top 200 tweets available, followers' names, and followings names. Also, we get all re-tweeters whom retweeted the retrieved users' tweets. As some tweets might be a reply to a user, we extract from the retrieved tweets the users' names that the considered user has replied to them. The intuition behind considering re-tweeters returns to the fact that sometimes spammers work together such that one spammer posts a tweet and the others re-tweet it. As one of the designed features is graph based, so instead of considering only the users who have direct connection as follower or friend with spammer, we extended the crawling algorithm to include users' that fall at distant *three* from the input spammer node. Retrieving users' that fall at distant three allows us to extract accurate features when using graph metrics as well as it reduces the crawling time that grows exponentially when going far in the distance. To make the process consistence, the followers and the followees of the considered user are added to a queue identified by their distance value. To handle the growing size of the graph, we add all followers and followees if the distance value is *zero* (spammer user). For the distances *one* and *two*, we constrained the adding process to be based on the number of unified names list of both followers and followees. So, we condition the length of the list to be less than 500 at distance *one*, and 100 at distance *two*. These values are enough to have near complete graph that can give accurate results. We add also to the queue the re-tweeters and the replied-to users without any condition, however, with setting the distance to $infinity$ since those users might not be in a direct relation with the considered user, and thus we don't add their followees and followers to the queue. At last, the retrieved information for each user which include user's profile, tweets, re-tweeters, names of followers, name of followings, and replied-to users are stored in an already created database to be used later in feature extraction process. It is important to mention that the user's profile object consists of all information required to extract the proposed features such as the creation date of the account.

We launched our crawler in February for two weeks using a prepared list containing names of 200 spammers. In total, we collected 7,189 users with almost 300k tweets. To validate our features using machine learning algorithms, we need to create an annotated data-set from the crawled information. To do so, we entered all accounts collected in a manual annotation process, to assign the correct class, $Spammer$ or $Legitimate\ User$, to each account, because the crawled users may have spammers not considered in the input list. Thus, beyond the given list of spammers, we assigned the class of each account using three-levels of observations. First, if the account was suspended in the time of annotation, we labeled that account as spammer. Second, we labeled the account as legitimate user in case the account is verified by Twitter community. At last, when the account is neither suspended nor verified, we examined precisely each account before deciding to which class belongs, starting from the posted tweets, profile description, and ending by the followers and followings. Obviously, the third level of filtering is a subjective process, requiring an explicit definition of the conditions that must be examined before judging on the user. Thus, we consider the following rules to label a given user as spammer: (i) containing phishing or malicious URLs in the user's posts; (ii) spreading pornography materials; (iii) misusing of hashtags or replied-to services in the posts; (iv) duplicating posts in an intensive way. With these conditions, the annotation process has resulted in 1083 users labeled as spammer including the spammers of the input list, and 6106 users labeled as legitimate users. Table 1 shows statistics about our annotated data-set based on user's class. We plan to make our labeled data-set available online to the research community in due time.

[4]https://dev.twitter.com/rest/public

Table 1: Statistics of the crawled Twitter accounts.

	Spammer	Legitimate User
Number of Users	1082	6106
Number of Tweets	47094	264746
Number of Hashtags	72697	152814
Number of URLs	27140	186468
Number of Geo-tagged Tweets	86	1096

5. RESULTS AND EVALUATIONS

5.1 Experimental Setting

Metrics. To assess the effectiveness and robustness of our proposed features using a machine learning algorithm, we adopt the accuracy and the standard existing information retrieval metrics of precision, recall, and f-measure as defined in [3]. We compute the retrieval metrics only for the *spammer* class only since the main problem is spammers detection, not legitimate users detection.

Data-sets. The crawled data-set is not balanced from classes distribution point of view. According to the machine learning principles, performing learning must be accomplished on totally or almost balanced data-set. Therefore, in order to utilize all examples we have, we created $five$ data-sets where each has 1082 examples of spammer's class and 1221 randomly selected examples of legitimate user's non-duplicated in other sets. The distribution of spammer and legitimate classes in each resulted data-set is about 53% and 47%, respectively.

Learning Algorithms. We used Random Forest, Support Vector Machine (SVM) with kernel trick method, J48, and Adaboost learning algorithms implemented well in WEKA [29] tool, to build or learn the binary predictor function y. We report here the results of Random Forest only which has shown the best results in terms of evaluation metrics when setting number of trees parameter to 1000, using 10-cross validation.

Weigthing Functions. For each feature that uses weighting function, we selected the optimal function f and the parameter λ through following steps: (i) a particular function and parameter value are picked from a defined list viewed as a search space; (ii) the feature vector is extracted from the $five$ data-sets using the selected f and λ; (iii) the Random Forest learning algorithm is applied on the feature vector extracted with averaging the accuracy values that results from $five$ data-sets, using 10-folds cross validation; (iv) the first step is repeated until the list becomes empty; (v) the best f and λ that have the highest accuracy are considered as the optimal function and parameter, respectively. In defining the list, we use different values of $\lambda \in [1, 1000]$ with 100 steps increment. The optimal weighting functions of the time weighted features are shown in Table 2.

5.2 One Feature Results

We compare our 20 features designed with 70 features existing in the literature distributed between 11 user features, 58 content features, and 1 graph feature. We implemented these features with experimenting them on our crawled data-set to have fare comparison. However, we excluded some of graph features such as distance, connectivity, and node betweenness since they are not suited for real time filtering. Also, the features extracted from geo-tag property [30] have been ignored because the ratio of tweets that have enabled utilized this property is less than 1% according to our data-set statistics.

Table 2: Accuracy of each feature used individually in the detection task, compared to the corresponding ones in the state of art, categorized based on the property or service.

Property/Service	Feature Name	Accuracy(%)	Function
Follower	Number of Followers [4, 3, 6, 8]	66.6%	-
	Weighted Followers	79.4%	$\exp^{\frac{t*t}{301}}$
	Followers' Age Mean	67.5%	-
	Followers' Age Variance	66.7%	-
Followee	Number of Followees [4, 3, 6, 8]	56.7%	-
	Weighted Followees	81.2%	$\exp^{\frac{t*t}{101}}$
	Followees' Age Mean	64.9%	-
	Followees' Age Variance	63.8%	-
Bi-directional	Bi-directional Percentage	62.8%	-
	Followers to Followees Ratio [5, 7]	64.5%	-
	Weighted Bi-directional	76.8%	$\exp^{\frac{t*t}{1}}$
	Bi-directional Age Mean	63.2%	-
	Bi-directional Age Variance	62.7%	-
Bio	Spam Word in Bio [3, 11]	59.1%	-
	Weighted Bio Similarity	72.2%	$\exp^{\frac{t*t}{101}}$
Hashtag	All Hashtags' Features [4, 3, 6, 25, 8]	66.6%	-
	Posting Hashtags Behavior	81.5%	-
	Hashtags' Diversity	70.1%	-
URL	All URLs' Features [4, 3, 6, 25, 8]	58.2%	-
	Posting URLs Behavior	77.6%	-
	URLs' Diversity	60.8%	-
Mentions	All Mentions' Features [4, 3, 6, 25, 8]	61%	-
	Posting Mentions Behavior	69.3%	-
	Mentions' Diversity	66.4%	-
Tweets	All Tweets' Features [4, 3, 6, 25, 8]	58.8%	-
	Weighted Tweets Similarity	65.5%	$\exp^{\frac{t*t}{1}}$
	Words Diversity	76.6%	-
Re-Tweets	All Re-Tweets' Features [9, 11]	58.7%	-
	Re-Tweeters Diversity	62.1%	-
Graph	Local Clustering [11]	57.1%	-
	Weighted Local Clustering	66.7%	$\exp^{\frac{t*t}{101}}$

Table 2 shows the resulting accuracy obtained when using one single feature in learning process, grouped at property or service level. As there are some features (e.g. URLs, Hashtags) extracted from one property, we averaged the resulting accuracy by those features. The effect of weighting function is obvious in follower, followee, bi-directional properties; it boosted up the accuracy by more than 12% compared to the classical state of art features. However, the use of mean and variance has not increased the accuracy too much compared to the state of art ones. As an interpretation, the distribution of users' ages may follow more than one Gaussian distribution, rather than there are legitimate users having recently created followers or followees which leads to be similar with spammer behavior. The posting behavior features have shown their strength in detecting spammers, especially the hashtag and URL services, with little contribution added by mention service. Indeed, this does not mean that the mention service is ineffective; however, based on our observation, the mention service had not been used in the same volume of URLs and hashtags by spammers. The diversity feature has not the same discriminant power of posting behavior, in particular URL and hashtag services. We associate this degradation in results with the fact of having legitimate users posting sometimes hashtags and URLs in some short period as interaction with particular events on Twitter. Conversely, words' diversity had captured the spamming behavior of using same words in the

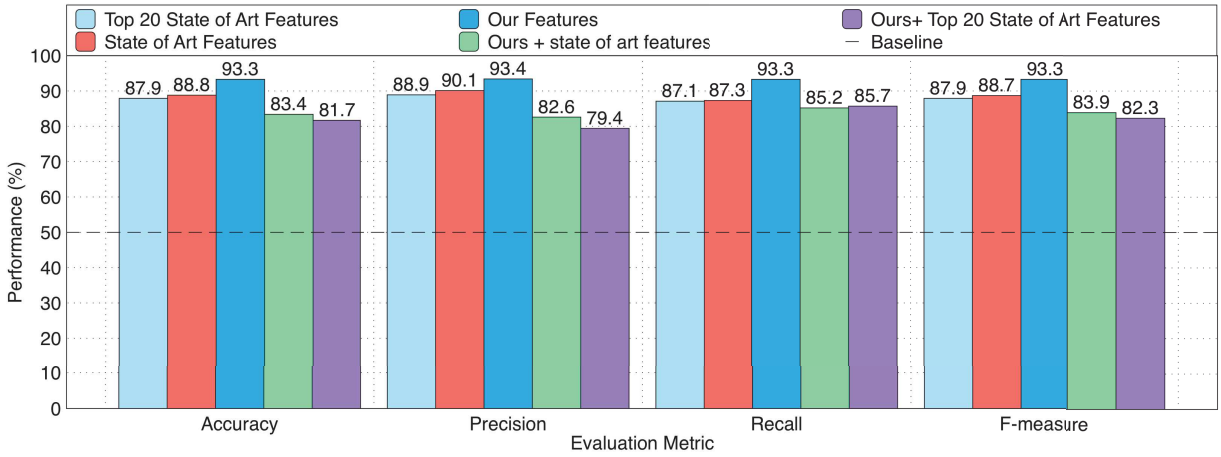

Figure 1: Performance of our 20 features compared with the state of art features and a baseline using different evaluation metrics.

most of tweets.

By these results, we can conclude that the leveraging of time either explicitly in designing features or implicitly through posting behavior and diversity has defeated and outperformed the corresponding features in the state of art, which verified our hypothesis about the time property.

5.3 All Features Results

Differently from one feature experiment, we experimented all 20 features together with comparing them to the performance of the state of art features. Also, we include in the comparison the concept of baseline, which is computed when predicting blindly any user as legitimate user. Therefore, since our data-sets are almost balanced, we set approximately the baseline to 50% for all metrics used.

According to Figure 1, our features outperform the 70 features of the state of art with about 4.5% in accuracy, 3.3% in precision, and 6% in recall. The 90.1% precision of the state of art features indicates that their features can detect successfully spammers more than legitimate users, however, without having high classification rate of legitimate users as spammers. As the size of our features is 20, we compare them with the top 20 features of the state of art selected using information gain algorithm implemented in Weka tool [29]. We found that the rest 50 features don't contribute significantly in the spammer detection task because of the decreasing in the performance by almost 1% with respect to the performance when using 70 features together. Beyond this simple degradation in the performance, the experiments of our features combined with the 70 and top 20 state of art features, respectively, have shown a dramatic decreasing in the classification task by more than 6% in all metrics used. The only interpretation to this phenomena is associated to the problem of over-fitting, which means that the learnt function y contains model(s) for noisy example.

All in all, our 20 features have defeated both the baseline and the 70 features existing in the state of art. Our features are suitable for real time filtering as well as they are robust against spammers', even though we use local clustering method to get a feature related to the graph level. Indeed, local clustering is the only metric that can be used for real time filtering because of its simplicity in regards to time complexity, compared to other graph metrics.

6. CONCLUSION AND FUTURE DIRECTIONS

In this paper, we approached the problem of detecting spammers on Twitter social network in a real time manner. Thus, we introduced a design of 20 robust features, suitable for real time filtering, through leveraging explicitly and implicitly the time property as an unmodifiable value by users. The experimental results, conducted on crawled and manually annotated data-set, shown that our features proposed are able to classify correctly both legitimate users and spammers with accuracy of more than 93%, defeated 70 features used in state of art by about 6%. Our work has answered on the question of how to identify spammers only. As a future work, we intend to search for spammers through predicting the spammy naming patterns as a search-able information in Twitter.

7. REFERENCES

[1] Formerly Digital Marketing Ramblings. By the numbers: 170+ amazing twitter statistics. `http://expandedramblings.com/index.php/march-2013\\-by-the-numbers-a-few-amazing-twitter-stats/`, 2013. [Online; accessed 1-March-2016].

[2] Akemi Takeoka Chatfield and Uuf Brajawidagda. Twitter tsunami early warning network: a social network analysis of twitter information flows. In *ACIS 2012: Location, location, location: Proceedings of the 23rd Australasian Conference on Information Systems 2012*, pages 1–10. ACIS, 2012.

[3] Fabricio Benevenuto, Gabriel Magno, Tiago Rodrigues, and Virgilio Almeida. Detecting spammers on twitter. In *In Collaboration, Electronic messaging, Anti-Abuse and Spam Conference (CEAS*, page 12, 2010.

[4] Alex Hai Wang. Don't follow me: Spam detection in twitter. In *Security and Cryptography (SECRYPT), Proceedings of the 2010 International Conference on*, pages 1–10, July 2010.

[5] Kyumin Lee, James Caverlee, and Steve Webb. Uncovering social spammers: Social honeypots + machine learning. In *Proceedings of the 33rd International ACM SIGIR Conference on Research*

and Development in Information Retrieval, SIGIR '10, pages 435–442, New York, NY, USA, 2010. ACM.

[6] M. McCord and M. Chuah. Spam detection on twitter using traditional classifiers. In *Proceedings of the 8th International Conference on Autonomic and Trusted Computing*, ATC'11, pages 175–186. Springer-Verlag, 2011.

[7] Gianluca Stringhini, Christopher Kruegel, and Giovanni Vigna. Detecting spammers on social networks. In *Proceedings of the 26th Annual Computer Security Applications Conference*, ACSAC '10, pages 1–9, New York, NY, USA, 2010. ACM.

[8] Chao Yang, Robert Chandler Harkreader, and Guofei Gu. Die free or live hard? empirical evaluation and new design for fighting evolving twitter spammers. In *Proceedings of the 14th International Conference on Recent Advances in Intrusion Detection*, RAID'11, pages 318–337, Berlin, Heidelberg, 2011. Springer-Verlag.

[9] Amit A Amleshwaram, Nutan Reddy, Suneel Yadav, Guofei Gu, and Chao Yang. Cats: Characterizing automation of twitter spammers. In *Communication Systems and Networks (COMSNETS), 2013 Fifth International Conference on*, pages 1–10. IEEE, 2013.

[10] Cheng Cao and James Caverlee. Detecting spam urls in social media via behavioral analysis. In *Advances in Information Retrieval*, pages 703–714. Springer, 2015.

[11] Zi Chu, Indra Widjaja, and Haining Wang. Detecting social spam campaigns on twitter. In *Applied Cryptography and Network Security*, pages 455–472. Springer, 2012.

[12] Claudia Meda, Federica Bisio, Paolo Gastaldo, and Rodoleo Zunlno. Machine learning techniques applied to twitter spammers detection. pages 177–182.

[13] Igor Santos, Igor Miambres-Marcos, Carlos Laorden, Patxi Galn-Garca, Aitor Santamara-Ibirika, and Pablo Garca Bringas. Twitter content-based spam filtering. In *International Joint Conference SOCOâĂŹ13-CISISâĂŹ13-ICEUTEâĂŹ13*, pages 449–458. Springer, 2014.

[14] Juan Martinez-Romo and Lourdes Araujo. Detecting malicious tweets in trending topics using a statistical analysis of language. *Expert Systems with Applications*, 40(8):2992–3000, 2013.

[15] Twitter. Twitterhelpcenter: How to report spam on twitter. `https://support.twitter.com/articles/64986#`, 2016. [Online; accessed 1-March-2016].

[16] Twitter. The twitter rules. `https://support.twitter.com/articles/18311#`, 2016. [Online; accessed 1-March-2016].

[17] Sarita Yardi, Daniel Romero, Grant Schoenebeck, and danah boyd. Detecting spam in a twitter network. *First Monday*, 15(1), 2009.

[18] Chao Yang, Robert Harkreader, Jialong Zhang, Seungwon Shin, and Guofei 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*, WWW '12, pages 71–80, New York, NY, USA, 2012. ACM.

[19] Jonghyuk Song, Sangho Lee, and Jong Kim. Spam filtering in twitter using sender-receiver relationship. In *Recent Advances in Intrusion Detection*, pages 301–317. Springer, 2011.

[20] Zi Chu, Steven Gianvecchio, Haining Wang, and Sushil Jajodia. Detecting automation of twitter accounts: Are you a human, bot, or cyborg? *Dependable and Secure Computing, IEEE Transactions on*, 9(6):811–824, 2012.

[21] Xia Hu, Jiliang Tang, and Huan Liu. Online social spammer detection. In *AAAI*, pages 59–65, 2014.

[22] Xia Hu, Jiliang Tang, Yanchao Zhang, and Huan Liu. Social spammer detection in microblogging. In *IJCAI*, volume 13, pages 2633–2639. Citeseer, 2013.

[23] Pang-Ning Tan, Michael Steinbach, and Vipin Kumar. *Introduction to Data Mining, (First Edition)*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2005.

[24] Solomon Kullback and Richard A Leibler. On information and sufficiency. *The annals of mathematical statistics*, 22(1):79–86, 1951.

[25] Alex Hai Wang. Detecting spam bots in online social networking sites: A machine learning approach. In *Proceedings of the 24th Annual IFIP WG 11.3 Working Conference on Data and Applications Security and Privacy*, DBSec'10, pages 335–342, Berlin, Heidelberg, 2010. Springer-Verlag.

[26] Po-Ching Lin and Po-Min Huang. A study of effective features for detecting long-surviving twitter spam accounts. In *Advanced Communication Technology (ICACT), 2013 15th International Conference on*, pages 841–846, Jan 2013.

[27] Kurt Thomas, Chris Grier, Dawn Song, and Vern Paxson. Suspended accounts in retrospect: An analysis of twitter spam. In *Proceedings of the 2011 ACM SIGCOMM Conference on Internet Measurement Conference*, IMC '11, pages 243–258, New York, NY, USA, 2011. ACM.

[28] Hongyu Gao, Yan Chen, Kathy Lee, Diana Palsetia, and Alok N Choudhary. Towards online spam filtering in social networks. In *NDSS*, page 16, 2012.

[29] Mark Hall, Eibe Frank, Geoffrey Holmes, Bernhard Pfahringer, Peter Reutemann, and Ian H. Witten. The weka data mining software: An update. *SIGKDD Explor. Newsl.*, 11(1):10–18, November 2009.

[30] Diansheng Guo and Chao Chen. Detecting non-personal and spam users on geo-tagged twitter network. *Transactions in GIS*, 18(3):370–384, 2014.