Behavioural account-based features for filtering out social spammers in large-scale twitter data collections [1]

Mahdi Washha, Manel Mezghani, Florence Sèdes

IRIT, Université Paul Sabatier, 118 Route de Narbonne, 31062 Toulouse Cedex 9, France

{mahdi.washha,florence.sedes}@irit.fr;mezghani.manel@gmail.com

ABSTRACT. *Online social networks (OSNs) have become an important source of information for a tremendous range of applications and researches. However, the high usability and accessibility of OSNs have exposed many information quality (IQ) problems which consequently decrease the performance of OSNs dependent applications. Social spammers are a particular kind of ill-intentioned users who degrade the quality of OSNs information through misusing all possible services provided by OSNs. Given the fact that Twitter is not immune towards the social spam problem, different researchers have designed various detection methods of a spam content. However, the tweet-based detection methods are not effective for detecting a spam content because of the dynamicity and the fast evolution of spam. Moreover, the robust account-based features are costly for extraction because of the need for huge volume of data from Twitter's servers, while most other account-based features don't model the behavior of social spammers. Hence, in this paper, we introduce a design of new 10 robust behavioral account-based features for filtering out spam accounts existing in large-scale Twitter "crawled" data collections. Our features focus on modeling the behavior of social spammers, such as the time correlation among tweets. The experimental results show that our new behavioral features are able to correctly classify the majority of social spammers (spam accounts), outperforming 75 account-based features designed in the literature.*

RÉSUMÉ. *Les réseaux sociaux en ligne (OSN) sont devenus une source importante d'information pour une vaste gamme d'applications et de recherches. Cependant, la grande facilité d'utilisation et l'accessibilité des OSN ont exposé de nombreux problèmes associés à la qualité de l'information qui, par conséquent, diminuent les performances des applications dépendantes des OSN. Étant donné que Twitter n'est pas à l'abri du problème de spam social, les chercheurs ont conçu diverses méthodes de détection de spam. Cependant, les méthodes de détection basées sur le tweet ne sont pas efficaces pour détecter le contenu spam en raison de la dynamique et*

1. This work is an extended version of a published work Washha *et al.* (2016a) in "32e Conférence sur la Gestion de Données - Principes, Technologies et Applications" (BDA 2016).

l'évolution rapide du contenu spam. En outre, les méthodes basées sur les comptes sont coûteuses pour l'extraction en raison de la nécessité d'un énorme volume de données provenant des serveurs de Twitter, tandis que la plupart des autres fonctionnalités basées sur le compte ne modélisent pas le comportement des spammeurs sociaux. Par conséquent, dans cet article, nous présentons une conception de nouvelles fonctionnalités robustes basées sur le compte pour filtrer les comptes spam existant dans de grandes collections "aspirées".Nos fonctionnalités se concentrent sur la modélisation du comportement des spammeurs sociaux, comme la corrélation du temps entre les tweets. Les résultats expérimentaux montrent que nos nouvelles fonctionnalités comportementales sont capables de classer correctement la majorité des spammeurs sociaux (comptes spam), surperformant 75 fonctionnalités de l'état de l'art.

KEYWORDS: *Twitter, social network, spam.*

MOTS-CLÉS : *Twitter, réseau social, spam.*

DOI:10.3166/ISI.22.3.65-88

1. Introduction

Online Social Networks (OSNs) have an enormous popularity over the Internet because of the wide range of services they provide for their users. The most popular OSNs such as Twitter, and Facebook have exceeded billions of registered users and millions of daily active users (Chen *et al.*, 2016). OSNs mainly rely on their users as primary contributors in generating and posting information. Users' contributions might be exploited in different positive ways such as understanding users' needs, and analyzing users' opinions for election purposes (Sedhai, Sun, 2016). However, the usability of OSNs and the absence of effective restrictions on posting have exposed different information quality problems such as social spam, and information overload. These characteristics have subjected OSNs to different attacks by ill-intentioned users, so-called social spammers, to post spam content (Agarwal, Yiliyasi, 2010). Social spammers intensively post non-sensical content in different contexts (e.g., topics) and in an automated way. For example, posting a tweet talking about "how to earn 100$ in 5 minutes" under the "#BBC" topic is a spam tweet because such a tweet has no relation to the given topic at all. Generally, social spammers have a wide range of goals to publish a spam content in OSNs, summarized in (Washha *et al.*, 2016b, Yardi *et al.*, 2009): (i) spreading advertisements to generate sales; (ii) disseminating porn materials; (iii) publishing viruses and malware; (iv) and creating phishing websites.

Motivation and Problem. As OSNs have many information quality problems, in this work, we handle a particular issue related to the social spam problem in Twitter platform. More precisely, we address the problem of filtering out spam accounts existing in large-scale "crawled" collections. The solution, that will be introduced in this paper, has been integrated with our team researches on social networks. Our team has researches addressing many issues related to OSNs such as tweet summarization (Abdelhamid *et al.*, 2016), event detection (Hoang, Mothe, 2016), social profiling (Sirinya *et al.*, 2014), profiles enrichment (Mezghani *et al.*, 2014), socio-semantic communi-

ties detection (Rocío *et al.*, 2015), and social interests (Manel *et al.*, 2014) detection, where Twitter platform has been adopted as information source. Thus, experimenting and working on high quality of Twitter information is an indispensable step to obtain and maintain high performance results in our team researches.

At the technical level, a considerable set of methods has been proposed to reduce and eliminate the social spam problem. Most of the existing works are dedicated for detecting individual Twitter spam accounts (Ahmed, Abulaish, 2013, Meda *et al.*, 2014, Yang *et al.*, 2012, Perdana *et al.*, 2015, Amleshwaram *et al.*, 2013, Guo, Chen, 2014, Singh *et al.*, 2014, Cao, Caverlee, 2015, Yardi *et al.*, 2009, Yang *et al.*, 2011, Fabricio *et al.*, 2010, Hai, 2010b, Almaatouq *et al.*, 2016, Washha *et al.*, 2016b, Hu *et al.*, 2013, Chen *et al.*, 2016) or spam campaigns (Zhang *et al.*, 2012, Zi *et al.*, 2012). These methods mainly exploit the supervised machine learning approach combined with the features extraction concept to produce binary classifiers using annotated data-sets. The features introduced in the literature for spam accounts detection are categorized into three types: (i) user-based features such as number of followers; (ii) content-based features like number of URLs; (iii) and graph-based features such as local-clustering. Most of the features introduced in the first two types are suitable for processing large-scale collections of Twitter accounts because those features don't require too many information from Twitter's servers like the graph-based features. The methods of spam campaign detection are not appropriate for handling large-scale collections since the features of this level require huge volume of information from Twitter's servers, making the treatment of large-scale collection almost impossible. One might suggest the use of cloud systems to process big volumes of Twitter data in parallelized and fast way. However, this solution is not applicable since the main source of limits is from Twitter's servers in which the number of API calls is constrained to a defined number of calls. Thus, to have a near-unlimited of API calls, we must obtain a big number of hundred API tokens to getting access in Twitter data. Unfortunately, obtaining big number of API tokens is not possible since it requires "phone verified" Twitter accounts.

The other less used approach (Chao *et al.*, 2015, Chen *et al.*, 2015, Martinez-Romo, Araujo, 2013, Fabricio *et al.*, 2010) detects spam tweets instead of spam accounts. However, this tweet level approach is not effective since the content of any tweet is up to 140 characters. Hence, the features that can be extracted from a tweet are not useful to detect spam tweets effectively. Moreover, the existing attempts based on spam account detection have critical limitations and major drawbacks. One of these limitations is the ease of manipulation in the existing features by social spammers. 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 social spammers; however, this feature can be easily manipulated by social spammers through creating a huge number of accounts and letting each account to follow each other. Another feature is counting the words in a user's tweets, where such a feature is used in discriminating among spam accounts and non-spam accounts. Unfortunately, most of the user and content features introduced in the literature are similar in performance to the given two examples. Thus, this raises the need to search for new robust features that can

detect spam accounts (or spam users) effectively and efficiently, with minimizing the need for Twitter's information as much as possible.

Contributions. In this paper, we introduce a design of a new set of features suitable for processing large-scale collections of Twitter users. Our features focus on modeling social spammers' behaviors by deeply analyzing their posting behavior such as writing style similarity among user's tweets. In designing our features, we assume that social spammers have different behavior from normal (legitimate) users in posting tweets. In other words, social spammers have systematic posting patterns, while normal users have a kind of randomness in posting content on Twitter. We validate the robustness of our features through a series of experiments conducted on a large-scale data-set consisting of more than 400,000 annotated Twitter accounts, using different supervised machine learning algorithms. The experimental results demonstrate that our new features are able to correctly classify the majority of social spammers (spam accounts) with more than 70% of accuracy, precision, recall, and f-measure, when using Random Forest learning algorithm, outperforming 75 account-based features designed in the literature.

The rest of the paper is organized as follows. Section 2 presents Twitter's rules followed in fighting social spammers, and the works that have addressed the social spam in Twitter. Section 3 shows the formalization and the definition of the problem we investigate, in addition to the design of our features. Section 4 describes the data-set used in performing a series of experiments. Section 5 evaluates our features by using machine learning algorithms, including a deep comparison with a wide set of state-of-art features. Section 6 concludes the paper with giving some insights about future directions in Twitter spam detection.

2. Background and Related Work

Social Spam Definition. Social spam is defined as a nonsensical or a gibberish text content appearing on OSNs and any website dealing with user-generated content such as chats and comments (Agarwal, Yiliyasi, 2010). Social spam may take tremendous range of forms, including profanity, insults, hate speeches, fraudulent reviews, personally identifiable information, fake friends, bulk messages, phishing and malicious links, and porn materials (Chen *et al.*, 2016). One might view the social spam as an non-relevant information; however, this interpretation is quite not accurate. We justify this misinterpretation through the definition of information retrieval (IR) systems (Manning *et al.*, 2008). The relevancy of documents in IR systems is dependent on an input search query. Thus, the irrelevant documents with respect to an input query are "not" necessary to be a spam content. Hence, as an additional definition, social spam might be defined as irrelevant information that doesn't have an interpretation in any context as long as the input query is not a spam.

Social Spam and Information Quality. Social spam has a strong relation to the the information quality (IQ) field. As shown in Figure 1 (Agarwal, Yiliyasi, 2010), four categories and different dimensions are adopted for evaluating applications and

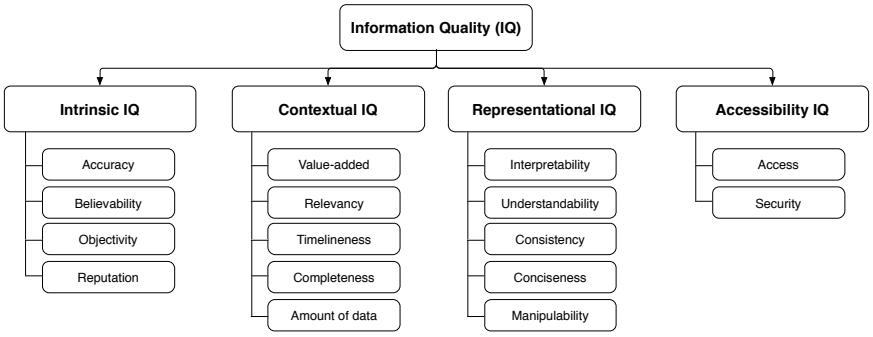

Figure 1. Information quality categories with their dimensions

challenges that have a relation with information area. Social spam problem can be projected on five IQ dimensions. Spam content does not represent a real world data and thus it has low degree in accuracy and believability dimensions. Similarly, the reputation of spam is also low because normal users tend to circulate accurate information. Also, spam content doesn't deliver any benefit for OSNs' users, leading to have low degree in terms of value-added and relevancy dimensions. Although projecting social spam problem on IQ world could provide more insights for handling the problem efficiently; social spammers spend great efforts to increase the degree of IQ dimensions. Therefore, understanding and knowing facts about social spammers can contribute in providing effective solutions for the social spam problem.

Social Spammers' Trends. Social Spammers misuse all legal methods or services supported by social networks to spread their spam contents. Regardless the targeted social network, social spammers adopt same trends in their goals and behaviors, summarized in (Washha *et al.*, 2016b):

– Social spammers are goal-oriented persons targeting to achieve unethical goals (e.g., promoting products), and thus they use their smartness to accomplish their spamming tasks in an effective and a quick way.

– Social spammers often create and launch a campaign of spam accounts in a short period (e.g., one day), to maximize their monetary profits and speedup their spamming behavior.

– As a set of APIs is provided by social networks, social spammers leverage them to automate their spamming tasks in a systematic way (e.g., tweeting every 10 minutes). More precisely, they avoid the random posting behavior because it may decrease their target profits.

Twitter's Anti-Spam Mechanism. Twitter provides an option for its users to report spam accounts through clicking on "Report: they are posting spam" option available in all accounts. Once an account is reported, Twitter's administrators manually review that account to make a suspension decision. However, combating social spammers using this reporting mechanism is inefficient because of the need for great efforts from both users and administrators. Moreover, not all reports are trustworthy, meaning

that some reported accounts might be for legitimate users, not for social spammers. In addition to this manual reporting mechanism, Twitter has defined some general rules (e.g., not allowed to post porn materials) for public in order to reduce the social spam problem as much as possible with permanently suspending the accounts that violate those rules (Twitter, 2016). However, social spammers can simply bypass Twitter's rules. For instance, social spammers may coordinate multiple accounts with distributing their desired workload among accounts to mislead the detection process. These accounts tend to exhibit an invisible spam behavior. Thus, these shortcomings have motivated researchers to introduce more robust methods for the applications that use Twitter as information source. We categorize the Twitter social spam detection approaches into two different types based on the automation detection level: (i) machine learning level as a fully automated approach; (ii) and social honeypot as a manual approach requiring human interactions.

Machine Learning Approach. In this approach, researchers have built their methods through employing three levels of detection, distributed between tweet-level detection, account-level detection, and campaign-level detection.

Tweet-Level. Martinez-Romo and Araujo (Martinez-Romo, Araujo, 2013) have designed a language model based method to detect spam tweets existing in trending topics. The method computes the kullback-leibler divergence between the language model of each tweet and the language model of the topic itself. However, this method is not suitable for real-time filtering because of the need for the tweets that have been posted in the same topic from Twitter's servers. The works introduced in (Fabricio *et al.*, 2010, Chao *et al.*, 2015, Chen *et al.*, 2015) have proposed a set of light statistical features such as number of words with a set of time-independent machine learning algorithms such as support vector machine (SVM), and Random Forest, to build a binary classifier. Although of suitability of these works for real-time filtering; they have a major drawback in efficiently detecting spam tweets (i.e., low spam recalling) due to the evolving of spam content over time.

Account-Level. The works introduced in (Fabricio *et al.*, 2010, Washha *et al.*, 2016b, Wu *et al.*, 2017, Hai, 2010b, McCord, Chuah, 2011, Stringhini *et al.*, 2010, Meda *et al.*, 2016, Bara *et al.*, 2015) have focused on extracting feature from users' accounts, including the number of friends, number of followers, similarity between tweets, and ratio of URLs in tweets. In more dedicated studies, the works proposed in (Cao, Caverlee, 2015, Wang, Pu, 2015) have identified the spam URLs through analyzing the shorten URLs behavior like the number of clicks and the length of redirection chain. However, the ease of manipulation in this type of features by social spammers has given a motivation to extract more complex features by using the graph theory. For instance, the studies presented in (Yang *et al.*, 2011, 2012, Almaatouq *et al.*, 2016) have examined the relation among users using some graph metrics to measure three features, including node betweenness, local clustering, and bi-directional relation ratio. Leveraging such complex features gives high spam accounts detection rate; however, they are not suitable for treating large-scale collections because of the huge volume of data that must be retrieved from Twitter's servers.

Campaign-Level. Chu *et al.* (Zi *et al.*, 2012) have treated the spam problem from the collective perspective view. They have clustered a set of desired accounts according to the URLs available in the posted tweets, and then a defined set of features is extracted from the accounts clustered to be incorporated in identifying spam campaign using machine learning algorithms. Chu *et al.* (Chu *et al.*, 2012) have proposed a classification model to capture the difference among bot, human, and cyborg with considering the content of tweets, and the tweeting behavior. Indeed, the methods belonging to this detection level have a major drawback. The features used in these methods require a great number of REST API calls to obtain information like users' tweets and followers. Consequently, exploiting the current version of campaign-level methods is not appropriate for filtering large-scale collections of Twitter accounts due to the high volume of data required from Twitter's servers.

Beyond the features design level, the works introduced in (Hu *et al.*, 2014, 2013) have proposed two optimization frameworks which use the content of tweets and basic network information to detect spam accounts using an efficient online learning approach. However, the major limitation in these works is the need for information about the network from Twitter, making these methods inapplicable to large-scale data collections

Honeypot Approach. Social honeypot is viewed as an information system resource that can monitor social spammers' behavior through logging their information such as the information of accounts and any available content (Lee *et al.*, 2010). In fact, there is no significant difference between Twitter's anti-spam mechanism and the social honeypot approach. Both of them need an administrative control to produce a decision about the accounts that have fallen into the honeypot trap. The necessity of administrative control is to reduce the false positive rate, as an alternative solution to blindly classifying all users dropped in the trap as spam users.

3. Account-Based Features Design

In this section, we introduce notations, definitions, and formalization of our target problem. Then, we present the design of our features by which we distinguish among spam accounts and non-spam accounts.

3.1. Notations and Problem Formalization

Let $U_{Collection} = \{u_1, u_2, ...\}$ be a finite set of Twitter users representing a target data collection which requires processing to filter out the spam accounts (users) that belong to social spammers. In order to minimize the size of information needed from Twitter's servers, for each user $u_\bullet \in U_{Collection}$, we collect the top 100 tweets using a single REST API call. This number of tweets is the maximum number that Twitter can provide in a one single call. It is possible to retrieve more user's tweets, if any; however, this increases the number of API calls, doubling the required time to process the entire users in the given collection. This number of tweets (100 tweets) per user is

relatively enough to take a precises decision about user type. Hence, we model each user $u_\bullet \in U_{Collection}$ by a 2-tuple, $u_\bullet = (Tweets, Age)$. Each element in the tuple and some additional functions are defined as follows:

Tweets. We model the tweets of user $u_\bullet \in U_{Collection}$ as a finite set, $Tweets = \{t_1, t_2, ...\}$, where $t_\bullet$ represents a tweet object consisting of simple meta-data. We model these meta-data by a 5-tuple $t_\bullet = (Time, Hashtags, URLs, Mentions, Words)$, where $Time$ is the posting date of the tweet, $t_\bullet$, represented in seconds time unit computed since $1970/1/1$, $Hashtags$ is a finite set containing all hashtags posted in the tweet, $URLs$ also represents a finite set of all URLs posted in the tweet, $Mentions$ is a set of users who are mentioned in the tweet extracted through searching for words starting by @ symbol, and $Words$ is a finite set consisting of words posted in the tweet such that $Words \cap Hashtags \cap Mentions \cap URLs = \emptyset$.

Age. The creation date of each account is registered on Twitter's servers, when users setup their accounts. We compute the age of a user's account in days time unit through calculating the difference between the current time date ($Time_{now}$) and the creation date of the account ($Time_{creation}$), defined formally as $Age = Time_{now} - Time_{creation}$.

Kullback–Leibler Divergence (KLD) (M_1, M_2, V). Given two different language models, M_1, M_2 and a set of terms V (e.g., words of a tweet), we compute the similarity of the two language models using a customized version of KLD (Kullback, Leibler, 1951), defined as:

$$KLD(M_1, M_2, V) = \frac{\log |V| - \sum_{w \in V} P(w|M_1) * min(|\log \frac{P(w|M_1)}{P(w|M_2)}|, \log |V|)}{\log |V|} \quad (1)$$

where $P(w|M_\bullet)$ is the probability of the term w of being generated by the given language model $M_\bullet$. We perform this customization since the range of the classical KLD method is unbounded and thus the ∞ value appears when two language models are dissimilar. Hence, our customization reverses the semantic of KLD values (i.e., 0 $\Rightarrow$ dissimilar and 1$\Rightarrow$ similar), with bounding its value between 0 and 1.

Problem Formalization. With the presented notations and definitions, our main problem is to detect and filter out Twitter social spammers existing in a given collection. Formally, given a collection of Twitter accounts (or users), $U_{Collection}$, the problem is turned to build a binary classification model with minimizing the number of requests to Twitter's servers, $y : u_\bullet \rightarrow \{Social\ Spammer,\ Legitimate\ User\}$, $u_\bullet \in U_{Collection}$.

3.2. Features Design

In designing our features, we have deeply analyzed a large set of spam accounts suspended by Twitter. Unlike the state-of-the-art features, our features focus on modeling user's behaviors since, according to the social spammers' trends, legitimate users have completely different behaviors like the randomness in their tweets content and

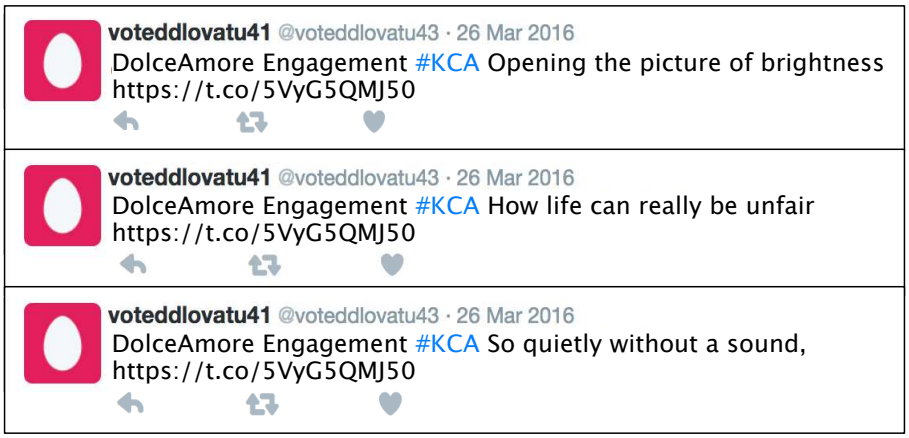

Figure 2. An example of a spam account posted three tweets having a correlation at the writing style level

structure. Also, the design of any feature must be not easy to be manipulated by social spammers. For instance, the number of tweet's words feature can be easily manipulated by social spammers through using low number of words in their tweets. Thus, we introduce a set of behavioral features that can efficiently and effectively distinguish among spam and non-spam accounts (or users), with relying only on the top 100 tweets.

Writing Style Similarity (WSS): Computing the textual similarity of user's tweets is a widely used feature; however, some social spammers are tricky enough to avoid tweets duplication. Given the fact that social spammers automate their posting in a systematic way, the probability of finding a correlation in the writing style among social spammers' accounts tweets is relatively high. For instance, the spam tweets in Figure 2 have a common style structure in writing tweets (word, word, hashtag, word, word,word, word, word, and then URL). In this instance, the social spammer of these tweets has been tricky in writing tweets so that no tweets duplicated in the content.

We model this feature through transforming tweet's content to a higher level of abstraction. Then, we measure the writing style similarity among tweets using Jaccard similarity index. In a formal way, a transformation function, $Type(ST, T) \in \{W, H, U, M\}$, is defined, which takes a string ST and a tweet T as parameters, and returns the type of the input string (**W**ord, **H**ashtag, **U**rl, and **M**ention), formalized as:

$$Type(ST, T) = \begin{cases} H & ST \in T.Hashtags \\ M & ST \in T.Mentions \\ U & ST \in T.URLs \\ W & ST \in T.Words \end{cases} \quad (2)$$

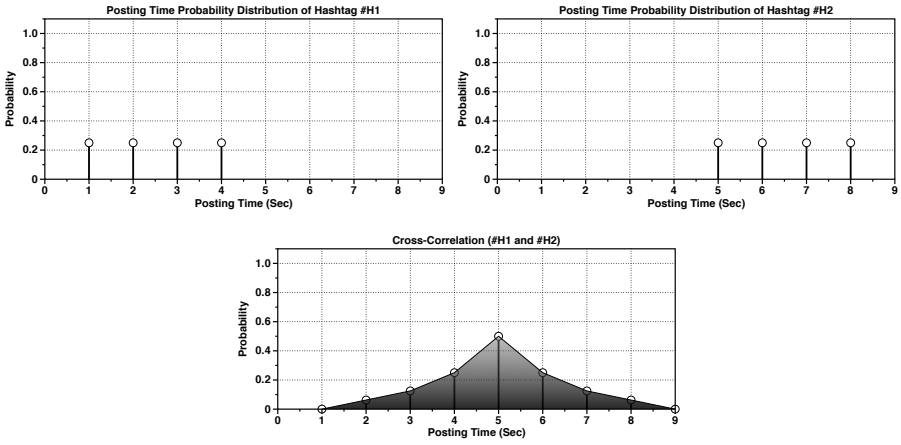

Figure 3. An illustrative example showing: (i) two correlated posting time probability distributions of two instances (#H1 and #H2); and (ii) the cross-correlation among the two probability distributions with the area (black shaded). The x-axis represents the shifted time-stamp of the tweets, computed by subtracting the time-stamp of each tweet from the most recent post tweet by the user. The 0 value on x-axis corresponds to the recency in time

Hence, for a tweet, $T_\bullet$, posted by a user $u \in U_{Collection}$, the writing style (WS) of the tweet is given as:

$$WS(T_\bullet) = \{(i, Type(S, T_\bullet)) | S \in T_\bullet.Hashtags \cup T_\bullet.Mentions \cup T_\bullet.URLs \cup T_\bullet.Words\} \tag{3}$$

where i is the position of the string S that requires transformation.
As a concert example using first tweet of Figure 2, the writing style set will be $WS = \{(1, W), (2, W), (3, H), (4, W), (5, W), (6, W), (7, W), (8, W), (9, U))\}$. With these definitions, for a user, $u \in U_{Collection}$, we compute the writing style similarity among user's tweets, $uTweets$, as follows:

$$WSS(u) = \frac{\sum_{T_1 \in u.Tweets} \sum_{T_2 \in u.Tweets} \frac{|WS(T_1) \cap WS(T_2)|}{|WS(T_1) \cup WS(T_2)|}}{(|u.Tweets|)(|u.Tweets| - 1)} \tag{4}$$

With this definition of WSS, the upper and lower bounds are between 1 and 0, respectively. The increasing in its value means that the user, u, has high probability of being a spam account. For instance, the writing style similarity of the user's tweets in Figure 2 is close to 1.

Posting Behavior. Social spammers are too dependent on the current events that are circulated in Twitter (Washha *et al.*, 2016b). For instance, trending topics of Twitter are changing over time and thus social spammers may intensively post tweets in a trending topic and then when that topic becomes not trending, they change to other trending one. Hence, given the fact that social spammers leverage the Twitter REST

APIs in automating the posting of spam content, a correlation might exist between the posting time probability distribution of hashatgs, mentions, URLs, or textual words. For example, Figure 3 shows two posting time probability distributions of tweets containing two instances (#H1 and #H2) of the hashtag service. It is obvious that the social spammer has focused on the #H2 when posting tweets and then after a while has changed the attention towards #H1. Although the two probability distributions are not identical in the posting time, they are correlated in the probability value and the time period between each two consecutive tweets. The probability of having such a correlation in a legitimate user's tweets is quite low since legitimate users are random in their posts and in using Twitter's services.

We model this posting behavior through computing the cross-correlation between different instances of a tweeting service: hashtags, URLs, and mentions services. In a formal way, let I_s be a set of all unique instances available in user's, $u \in U_{Collection}$, tweets and posted with a tweeting service $s \in \{Mention, Hashtag, URL\}$. Also, let P_i be the posting time probability distribution of the instance, $i \in I_s$. Since the posting time distributions time can be viewed as time shifted signals, we adopt the correlation (Oppenheim, 1999) method to measure the posting behavior similarity among user's tweets of the user u, defined as:

$$\begin{aligned} IS_u(I_s) &= \frac{\sum\limits_{i1 \in I_s} Area(P_{i1} \star P_{i2})}{|I_s| * Area(P_{i_{max}} \star P_{i_{max}})} \\ i2 &= \underset{i3 \in I_s \cap i3 \neq i1}{\arg\max}\ Area(P_{i1} \star P_{i3}) \\ i_{max} &= \underset{i \in I_s}{\arg\max}\, Area(P_i \star P_i) \end{aligned} \tag{5}$$

where $Area(\bullet)$ is a function that computes the area of the new distribution resulted by applying correlation (i.e., 0 area means dissimilar distributions), $P_\bullet \star P_*$ is a cross-correlation between two different distributions, and $P_\bullet \star P_\bullet$ is a correlation between same distribution known as an 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 area of the instance that has the maximum self-similarity multiplied by the number of instances. Thus, the IS value is between 0 and 1, where 0 means that there are no instances having same posting behavior, while 1 means that all instances have similar posting behavior. As social spammers might use all possible posting services, we extract three features through applying equation 5 on hashtags, URLs, and mentions services. The definition of the three instances sets are defined as $I_{hashtags} = \bigcup_{t \in u.Tweets} t.Hashtags$, $I_{Mentions} = \bigcup_{t \in u.Tweets} t.Mentions$, and $I_{URLs} = \bigcup_{t \in u.Tweets} t.URLs$.

Posting Diversity. Legitimate users and social spammers may use 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 (Fabricio *et al.*, 2010) don not significantly contribute in distinguishing among users' types. Our feature goes beyond these statistical ones through computing the posting diversity for each service separately such as the diversity of hashtags used in user's tweets. As an intuition, spammers intensively post their tweets 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 a tweeting service. By using the same definitions used in posting behavior feature part, the diversity of an instance set, I_s, of a service s, is computed as:

$$PD(u, I_s) = \frac{|I_s|}{|u.Tweets|} \tag{6}$$

The 0 value of PD means that the instances set is empty, while the 1 value means that each instance in I_s has been used only once in the user's u tweets. We apply this feature on four different tweeting services, including hashtags, mentions, URLs, and textual words services. Hence, the definition of the four instance sets are defined as $I_{hashtags} = \bigcup_{t \in u.Tweets} t.Hashtags$, $I_{Mentions} = \bigcup_{t \in u.Tweets} t.Mentions$, $I_{URLs} = \bigcup_{t \in u.Tweets} t.URLs$, and $I_{Words} = \bigcup_{t \in u.Tweets} t.Words$.

Language Model-Based Tweets Similarity. Social spammers try to avoid detection through non-duplicating exactly their tweets by generating random sentences from a predefined dictionary of words. Thus, in this case, the exact similarity feature can easily fail in capturing this spamming behavior.

We model this spamming behavior through computing first the uni-gram word language model of user's, u, tweets. Then, we measure the similarity between the language model of each tweet and all user's tweets language model, using the customized version of Kullback-Leibler Divergence defined above. Formally, for a user $u \in U_{Collection}$, let M_{Tweets} be a uni-gram word language of the user's u tweets, and let M_T be a uni-gram word language model of the tweet $T \in u.Tweets$, the tweets similarity is computed as:

$$LMTS(u) = \frac{\sum\limits_{T \in u.Tweets} KLD(M_T, M_{Tweets}, T.Words)}{|u.Tweets|} \tag{7}$$

The upper and lower bound of $LMTS$ are between 1 and 0, respectively. The high value gives an indication that the user's u tweets are almost similar in the content and thus the probability of being a spam account (social spammer) is high, while the low value means that most user's tweets talking about different topics.

4. Data-set Description and Ground-Truth

In the literature, there is no publicly available data-sets for research uses. Moreover, for privacy reasons, social networks researchers provide only the IDs of objects in

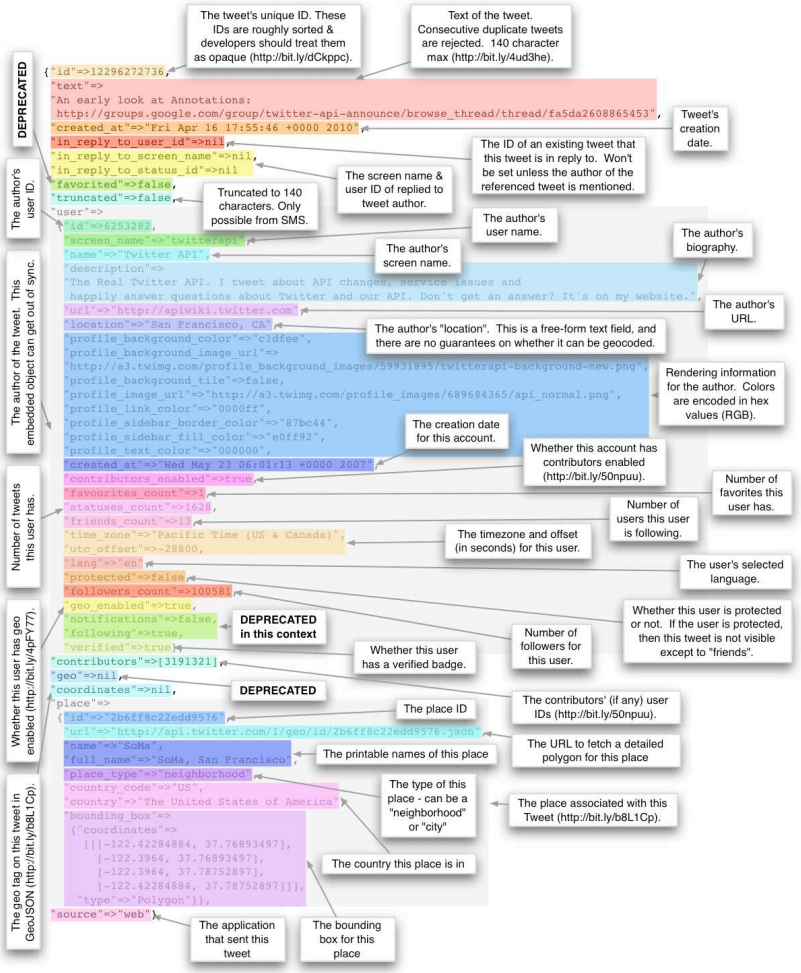

Figure 4. An example of tweet JSON object describing the accessible meta-data in any tweet object

their data-sets (e.g., tweets, accounts) to retrieve them from the OSNs' servers. However, inspired by the nature of the spam problem, providing IDs of spam tweets or accounts is not enough because Twitter might already have suspended the corresponding accounts and thus nothing to retrieve from Twitter's servers. Our data-set described in (Washha *et al.*, 2016b) consists of no more than 8,000 accounts manually annotated between spam and non-spam accounts. However, we have enlarged that data-set to draw more precise and accurate results.

Table 1. Detailed statistics of our data-set used in performing the experiments

	Social Spammers		Legitimate Users	
Statistic Name	**Value**	**Ratio (per 100 users)**	**Value**	**Ratio (per 100 users)**
Number of users	11,451 (2.8%)	—	409,170 (97.2%)	—
Number of geo-enabled users	2,542 (1.7%)	22 (38.6%)	147,200 (98.3%)	35 (61.4%)
Number of verified users	48 (0.3%)	0 (0.0%)	15,585 (99.7%)	4 (100%)
Number of users' followers	126,078,117 (0.11%)	1,101,022 (26.1%)	12,779,787,065 (99.8%)	3,123,344 (73.9%)
Number of users' followees	54,493,725 (3.6%)	475,886 (54.4%)	1,636,779,971 (96.4%)	400,024 (45.6%)
Number of tweets posted	146,626,275 (2.9%)	1,280,466 (51.1%)	5,024,219,375 (97.1%)	1,227,905 (48.9%)
Number of tweets retrieved	874,557 (2.6%)	7,637 (49.5%)	32,007,284 (97.4%)	7,822 (50.5%)
Number of retweeted tweets	331,995 (2.8%)	2,899 (50.7%)	11,464,552 (97.2%)	2,810 (49.2%)
Number of replied tweets	104,848 (2.4%)	915 (45.9%)	4,425,005 (97.6%)	1,081 (54.1%)
Number of URLs	185,925 (1.9%)	1,623 (39.9%)	10,011,831 (98.1%)	2,446 (60.1%)
Number of Hashtags	468,593 (3.3%)	4,092 (55.1%)	13,677,994 (96.7%)	3,342 (44.9)

Crawling Method. We exploit our research team crawler to collect accounts and tweets, launched since 1st June 2016. The streaming method is used to get an access to 1% of global tweets, as an unbiased crawling way. For each tweet being streamed, we extract the user ID of the tweet and then we retrieve the top 100 users' tweets using Twitter REST APIs. We store uesrs' tweets in JSON format where the meta-data that Twitter provides in any tweet is shown and annotated in Figure 4.

Data-set Description and Ground-Truth Building. We perform our experiments on a data-set consisting of around 420,000 Twitter accounts, after merging our previous data-set used in (Washha *et al.*, 2016b). These accounts are a result of 60 days of crawling from 1/June/2016 to 31/July/2016. To evaluate the effectiveness of our features and the state-of-the-art ones, we created an annotated data-set through labeling each account (user) as a spam or non-spam. However, with the huge amount of accounts, using manual annotation approach to have labeled data-sets is an impractical solution. Hence, we leverage a widely followed annotation process in the social spam detection researches, named as "Twitter Suspended Spammers (TSS)" (Hu *et al.*, 2013, 2014), summarized in Figure 5. The process checks whether each user was suspended by Twitter. In case of suspension, both the user and his tweets are labeled as a spam; otherwise we assign non-spam to both of them. In total, as reported in Table 1, we have about 11,500 spam accounts suspended by Twitter, forming around 3.0% of all accounts in our data-set. It is important to mention that not all non-spam accounts are truly non-spam, since Twitter might not have suspended some of them yet. However, on the other side, all accounts suspended by Twitter in our data-set are truly belonging to social spammers. Thus, the ratio of social spammers in our data-set is more than the percentage reported in Table 1. Although our data-set is not balanced; the normalized (per 100 users or accounts) statistics in Table 1 show the effectiveness of social spammers in polluting Twitter content. For example, Twitter social spammer averagely posts about 12K of tweets, which is almost equal to legitimate users' tweets. Also, the given statistic about the average number of hashtags posted by 100 users shows how much the hashtag service is used by social spammers in spreading their spam content. One important thing is the number of followers and followers. Indeed, according to

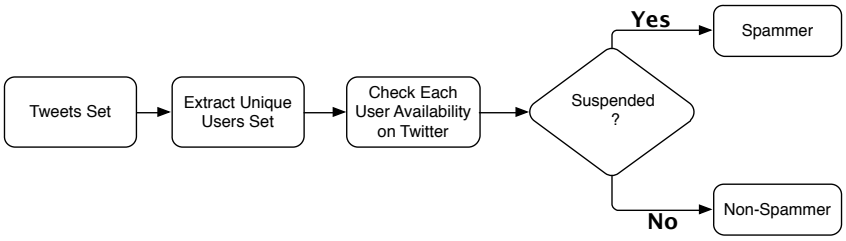

Figure 5. A digram showing the process used in building ground-truth data-set for an input set of tweets or accounts

the recent statistics [2] released in the 4^{th} quarter of 2016, the number of Twitter active users is about 320 million, while the numbers reported about followers and followees are exponentially higher than that number. Indeed, we compute these two numbers by summing the followers and followees counters available in user's object as meta-data, without deleting the duplicated accounts. These unimaginable numbers of follower and followees show the impracticality of graph-based features to process our data-set because of the need to retrieve almost all users (320 million) from Twitter's severs.

5. Results and Evaluations

5.1. Experimental Setting

Performance Metrics. As the ground truth class label about each user is available, we exploit the accuracy, precision, recall, F-measure, average precision, average recall, and average F-measure, computed according to the confusion matrix of Weka tool (Hall *et al.*, 2009), as commonly used metrics in classification problems. As our problem is two-class (binary) classification, we compute the precision, recall, and F-measure for the "spammer" class, while the average metrics combine the both classes based on the ratio of each class (e.g., 4.9% * "spammer precision" + 95.1% * legitimate user precision").

Baselines. We define two baselines to compare our method with them: (i) baseline "A" which represents the results when classifying all users as legitimate users directly without doing any kind of classification; (ii) and baselines "B" and "C" reflect the results when using the account user and content features that have been introduced in the literature, respectively. We adopt 77 of the state-of-the-art features listed in Table 2 and 3, distributed between user features and content features (Meda *et al.*, 2014, Yang *et al.*, 2012, Amleshwaram *et al.*, 2013, Guo, Chen, 2014, Hai, 2010a, Stringhini *et al.*, 2010, Yardi *et al.*, 2009, Yang *et al.*, 2011,?, Hai, 2010b, Almaatouq *et al.*, 2016, McCord, Chuah, 2011, Lee *et al.*, 2010). We avoid the comparison with the graph

2. https://www.statista.com/statistics/282087/number-of-monthly-active-twitter-users/

Table 2. A set of state-of-the-art of user-based features adopted in the comparison with our new account-level features

Index	Feature Name	Description
	User Features	
1	Number of tweets	Number of tweets that have been posted by the user.
2	Verified user	Boolean indicator showing whether the user is verified by Twitter.
3	Account age	Number of milliseconds spent since the creation date of the user's account.
4	Existence of Spam words in the screen-name attribute	Boolean indicator checking whether the screen name attribute contains a spam word.
5	Number of lists	Number of groups that have listed the user.
6	Default account image	Boolean indicator checking whether the image of the user's profile is the default one.
7	Geo-enabled	Boolean indicator showing whether the geographical location of the user's account is activated or not.
8	Screen-Name length	Length of the screen-name attribute of the user.
9	Profile description length	Number of words of the profile description.
10	URL in profile description	Boolean indicator stating whether the description of the user's profile has a URL.
11	Number of followers	Number of accounts that follow the user.
12	Number of followees	Number of accounts that the user follows.
13	Fraction of followers to followees	Ratio of the user's followers to the user's followees.
14	Following to Followers Ratio	Ratio of the user's followees to the user's followers.
15	Number of Friends	Number of accounts that the user follows them and they follow the user in the same time (i.e., intersection of followers and followees).
16	Reputation	Ratio of number of followers to the sum of both followers and followees sets.
17	Number of spam words in the account description	Number of the spam words that exist in the description of the user's account.

features because of the huge number of information needed from Twitter's servers to extract them, requiring months to collect these information.

Balanced Data-sets. It is obvious that our data-set is imbalanced in the class label distribution where the ratio of social spammers class is less than 3%. In fact, conventional supervised machine learning algorithms are often biased towards the majority class (Wallace *et al.*, 2011). The biasing problem happens because the loss functions of these algorithms attempt to optimize quantities such as error rate without taking the distribution of classes into consideration. In our problem, the worst case happens when the minority class (social spammers) examples are treated as outliers with respect to the majority class (legitimate users). Thus, these learning algorithms simply generate trivial classifiers that classify every example as the majority class. The recommended solution for such a problem is performing either oversampling or undersampling as a preprocessing step (Wallace *et al.*, 2011). However, applying oversampling method is not suitable in our case since to make the data-set balanced, we have to increase the number of social spammer examples to around 400,000 and thus the over-fitting problem might easily occur. Hence, we adopt the undersampling approach to have a balance data-set. In order to utilize all labeled examples, we have created $35 \approx \frac{409{,}170}{11{,}451}$ sub-datasets where each one has $22,902$ examples distributed equally between the so-

Table 3. A set of state-of-the-art of content-based features adopted in the comparison with our new account-level features

Index	Feature Name	Description
	Content Features	
1	Number of mentions	Number of accounts that have been mentioned in the user's tweets
2	Number of unique mentions	Size of the unique set of accounts that have been mentioned in the user's tweets.
3	Number of mentions per tweet	Total number of mentions in the user's tweets to the number of the tweets.
4	Number of numeric characters per tweet	Ratio of the total number of numeric characters existing in the user's tweets to the number of the user's tweets.
5	Number of replied to	Number of tweets that Twitter users have replied to the user.
6	Number of user's replies	Number of tweets that contain a reply to other users.
7	Ratio of replied tweets	Ratio of the number of replied tweets to the number of user's tweets.
8	URL ratio	Ratio of the tweets that contain URLs to the number of user's tweets.
9	Tweets Similarity	Textual similarity degree among the user's tweets.
10	Time weighted Tweets similarity	Similarity between the user's tweets with weighting it using the difference time between tweets.
11	Number of words per tweet	Ratio of the number of words existing in the user's tweets to the number of user's tweets.
12	Duplicated Tweets	Ratio of tweets that have been duplicated by the user.
13	Number of Hashtags	Number of hashtags that are available in the user's tweets.
14	Max Hashtag frequency	Maximum probability value of the probability distribution of the hashtags mentioned in the user's tweets.
15	Mentions ratio	Ratio of the number of mentions in the user's tweets.
16	Unique mentions ratio	Ratio of the number of unique mentions in the user's tweets.
17	Unique URLs ratio	Ratio of the number of unique URLs in the user's tweets.
18	Interaction rate	Ratio of the number of tweets that have been replied to the user's followees and followers.
19	Cosine similarity	Average content similarity over all user's tweets using the standard cosine similarity over the bag-of-words vector representation.
20	Hashtaged tweets ratio	Ratio of the number of the user's tweets that contain at least one hashtag to the total number of the user's tweets.
21	Avg Hashtags per tweet	Average number hashtags that exist in each user's tweets.
22	Retweeted tweets ratio	Ratio of the number of retweeted tweets to the total number of the user's tweets.
23	Average tweet length	Average number of characters of the user's tweets.
24	Number of URLs	Number of URLs existing in the user's tweets.
25	Number of unique URLs	Length of the unique set URLs posted in the user's tweets.
26	URL repetition frequency	Average repetition frequency of a URL posted in the user's tweets.
27	Average number of spam words	Average number of the spam words that exist in the user's tweets.
28	Average number of hashtags per word	Average number of hashtags per word in the user's tweets.
29	Average number of URLs per word	Average number of URLs per word in the user's tweets.
30	Hijacking Topics	Cosine similarity between the user's tweets and the tweets of the topics that have been mentioned in the user's tweets.
31	Intersection with trending topics	Ratio of the number of trending topics mentioned in the user's tweets to the total number of all topics in the user's tweets.
32	Duplicated domain names ratio	Ratio of the number of unique domains that exist in the user's tweets to the number of tweets that contain at least one URL.
33	IP-to-Domain ratio	Ratio of the unique IP addresses resolved from the existing domains in the user's tweets to the number of unique domains.
34	URL and Tweet Similarity	Cosine similarity between the content of a tweet containing URL and the landing page content of that URL.
35	One-gram characters	Number of alphabetical characters in the user's tweets.
36	Number of favorites	Total number of tweets that the user has liked them.
37	Number of retweets	Total number of retweets that the user's tweets have gained.
38	Ratio of retweeted tweets	Ratio of the number of retweeted tweets to the number of user's tweets.
39	Mean tweets similarity	The average pairwise tweets similarity based on the term frequency inverse document frequency.
40	Number of Hashtaged tweets	The number of tweets that contain at least on hashtag in the user's tweets.
41	Hashtags density	The number of hashtags in the user's tweets normalized by the number of the user's tweets.
42	Tweets with links	The number of tweets that contain at least one URL link.
43	Links density	The number of URLs in the user's tweets normalized by the number of the user's tweets.
44	Average number of characters	Total number of characters in the user's tweets divided by the number of user's tweets.
45	Min, Max, Avg, Median of number of tweets posted per day	Minimum, maximum, average, and median of the number of tweets that are posted daily by the user.
46	Min, Max, Avg, Median of number of tweets posted per week	Minimum, maximum, average, and median of the number of tweets that are posted per week by the user.
47	Min, Max, Avg, Median of the time tweets	Minimum, maximum, average, and median of the time difference between two consecutive tweets.
48	Tweets time distribution	Distribution of the posting time of the user's tweets over 24-hour period.
49	Variance in tweet intervals	Variance in the posting time interval between each two consecutive tweets.
50	Active days	Number of days between the oldest tweet and newest tweet in the user's tweets.
51	API Ratio	Ratio of the number of the user's tweets posted by an API tool to the total number of the user's tweets.
52	API URL Ratio	Ratio of the number of the user's tweets containing a URL posted by an API tool to the total number of the user's tweets posted by API.
53	API Tweet Similarity	Content similarity of the tweets posted only by an API tool.
54	Country changes per month	Number of times per month that a user moves across country boundaries between consecutive tweet.
55	Speed limit per month	Monthly average number of times that the user has a traveling speed exceeding a defined speed threshold.
56	Mean speed	Average tweeting speed.
57	Max speed	Maximum tweeting speed that the user followed in his tweets.
58	Unique countries per month	Average monthly number of countries that the user has been in.

cial spammer and legitimate user classes. In all of these sub-datasets, we have used the same social spammer examples without duplicating legitimate user examples across the sub-datasets.

Learning Algorithms. In experimenting the performance of our features and the state-of-the-art ones, we use $five$ supervised machine learning algorithms widely adopted in building binary classification models for Twitter social spam detection. These algorithms are: Naive Bayes, Random Forest with $\#Trees \in \{100, 500, 1000\}$, J48 with Confidence Factor $CF \in \{0.1, 0.5, 1.0\}$, K-nearest neighbor with $K \in \{2, 5, 10\}$, and support vector machine (SVM) with radial basis function kernel (RBF) and $\gamma \in \{0.5, 1.0\}$. We select these parameters since they have direct impact on producing high performance classification models, with taking the over-fitting problem into the consideration. We use Weka tool (Hall *et al.*, 2009) as an implementation for these algorithms.

Experiments Procedure. For each type of features (User, Content, and our Behavioral) and for a particular learning algorithm (Naive Bayes, Random Forest, J48, K-NN, or SVM), we perform the following steps: (i) we extract the selected type of features from each sub-dataset, producing 35 (number of sub-datasets) feature spaces; (ii) we apply 10-fold cross validation on each feature space using the learning algorithm chosen, resulting 35 confusion matrices; (iii) finally, we compute the final performance metrics for the chosen features type using the summation of the 35 confusion matrices, avoiding the computation of the variance across the 35 sub-datasets.

5.2. Experimental Results

Baseline Results. The results of the three baselines are reported in Table 4. The baseline "A" has accuracy of 50% because all social spammer examples have been classified as legitimate users in sub-dataset of the 35 sub-datasets. Thus, in such a case, the precision, recall, and F-measure of social spammer class are "0.0%" since no account is classified as a social spammer at all. This baseline is easy to be bypassed in all metrics when using supervised learning methods. For instance, the baseline "B" of the 17 user features has performance of 69,4% of accuracy as a best result when applying Random Forest learning method with $\#Trees = 1000$, compared to the other learning methods. We expect this behavior from Random Forest since it creates various classification models through constructing a multitude of decision trees (e.g, 1,000). However, in terms of precision metric, SVM has better performance than other learning methods. According to the low recalls and high precisions of social spammer class, the SVM method has effectively modeled very small sets of social spammer class examples, the rest of examples has been modeled as legitimate users. Moreover, the average precision of SVM is lower than the precision of social spammer class by about 24%. We interpret this behavior because our method followed in annotating our data-set is not too precise from legitimate users class view (i.e., not all legitimate users are truly legitimate users). The Naive Bayes learning method does not have an acceptable performance, especially in recalling social spammers (35.6%). On the other

Table 4. Performance results of the three baselines, our behavioral features, and the combination of user+content+behavioral features when applying the five mentioned machine learning algorithms in terms of different performance metrics

Model	Accuracy	Precision	Recall	F-Measure	Avg. Precision	Avg. Recall	Avg. F-measure
Baseline (A): All Users Labeled as Legitimate Users							
—	**50,0%**	0,0 %	0,0 %	0,0 %	**50,0%**	**50,0%**	**50,0%**
Baseline (B): User Features							
Naive Bayes	57,7 %	63,8 %	35,6 %	45,7 %	59,6 %	57,7 %	58,6 %
Random Forest (#Trees=100)	65,4 %	65,2 %	60,4 %	62,7 %	65,8 %	65,4 %	65,6 %
Random Forest (#Trees=500)	67,1 %	66,5 %	63,5 %	65,0 %	67,3 %	67,1 %	67,2 %
Random Forest (#Trees=1,000)	**69,4 %**	68,4 %	**65,4 %**	66,9 %	68,1 %	**69,4 %**	**68,7 %**
K-NN(K=2)	60,4 %	72,7 %	45,5 %	56,0 %	66,5 %	60,4 %	63,3 %
K-NN(K=5)	62,8 %	67,1 %	64,4 %	65,7 %	66,4 %	62,8 %	64,5 %
K-NN(K=10)	66,5 %	71,6 %	58,8 %	64,6 %	68,3 %	66,5 %	67,4 %
J48 (Confidence Factor=0,1)	62,1 %	72,1 %	63,5 %	**67,5 %**	69,8 %	62,1 %	65,7 %
J48 (Confidence Factor=0,5)	61,2 %	70,5 %	62,4 %	66,2 %	68,4 %	61,2 %	64,6 %
J48 (Confidence Factor=1,0)	60,5 %	70,3 %	62,2 %	66,0 %	68,2 %	60,5 %	64,1 %
SVM (Gamma=0,5)	51,0 %	**98,8 %**	2,0 %	3,9 %	**74,6 %**	51,0 %	60,6 %
SVM (Gamma=1,0)	52,1 %	97,8 %	4,0 %	7,7 %	74,6 %	52,1 %	61,4 %
Baseline (C): Content Features							
Naive Bayes	55,6 %	65,2 %	24,0 %	35,1 %	59,3 %	55,6 %	57,4 %
Random Forest (#Trees=100)	66,2 %	67,3 %	58,1 %	62,4 %	66,6 %	66,2 %	66,4 %
Random Forest (#Trees=500)	67,4 %	68,5 %	59,3 %	63,6 %	67,4 %	67,4 %	67,4 %
Random Forest (#Trees=1,000)	**68,3 %**	69,3 %	**60,5 %**	**64,6 %**	69,2 %	**68,3 %**	**68,7 %**
K-NN(K=2)	61,2 %	69,0 %	40,5 %	51,0 %	63,5 %	61,2 %	62,3 %
K-NN(K=5)	61,4 %	62,3 %	57,8 %	60,0 %	61,5 %	61,4 %	61,4 %
K-NN(K=10)	62,3 %	66,9 %	48,9 %	56,5 %	63,3 %	62,3 %	62,8 %
J48 (Confidence Factor=0,1)	60,7 %	61,4 %	57,4 %	59,3 %	60,7 %	60,7 %	60,7 %
J48 (Confidence Factor=0,5)	59,7 %	60,1 %	57,6 %	58,8 %	59,7 %	59,7 %	59,7 %
J48 (Confidence Factor=1,0)	59,6 %	60,0 %	57,5 %	58,7 %	59,6 %	59,6 %	59,6 %
SVM (Gamma=0,5)	51,8 %	89,9 %	4,1 %	7,8 %	70,4 %	51,8 %	59,7 %
SVM (Gamma=1,0)	53,2 %	**90,4 %**	5,3 %	10,0 %	**72,6 %**	53,2 %	61,4 %
Behavioral (Our) Features							
Naive Bayes	62 %	66,2 %	9,3 %	16,3 %	57,6 %	52,0 %	54,7 %
Random Forest (#Trees=100)	68,3 %	72,5 %	56,6 %	63,6 %	61,4 %	61,3 %	61,3 %
Random Forest (#Trees=500)	72,5 %	76,4 %	64,5 %	69,9 %	64,6 %	72,5 %	68,3 %
Random Forest (#Trees=1,000)	**78,5 %**	**78,6 %**	67,8 %	**72,8 %**	**67,4 %**	**78,5 %**	**72,5 %**
J48 (Confidence Factor=0,1)	68,8 %	70,9 %	**69,5 %**	70,2 %	59,1 %	58,8 %	58,9 %
J48 (Confidence Factor=0,5)	69,7 %	70,8 %	68,8 %	69,8 %	59,0 %	58,7 %	58,8 %
J48 (Confidence Factor=1,0)	74,7 %	70,8 %	68,8 %	69,8 %	59,0 %	58,7 %	58,8 %
K-NN(K=2)	67,2 %	72,9 %	55,4 %	63,0 %	58,9 %	57,2 %	58,0 %
K-NN(K=5)	69,7 %	68 %	65,8 %	66,9 %	57,7 %	57,7 %	57,7 %
K-NN(K=10)	71,7 %	71,5 %	66,4 %	68,9 %	59,2 %	58,7 %	58,9 %
SVM (Gamma=0,5)	65,4 %	72,3 %	67,4 %	69,8 %	60,0 %	59,4 %	59,7 %
SVM (Gamma=1,0)	67,4 %	73,1 %	65,5 %	69,1 %	60,2 %	59,4 %	59,8 %
Baseline (B)+Baseline (C)+Behavioral (Our) Features							
Naive Bayes	56,5 %	68,4 %	24,3 %	35,9 %	61,2 %	56,5 %	58,8 %
Random Forest (#Trees=100)	70,3 %	73,7 %	63,2 %	68,0 %	70,8 %	70,3 %	70,5 %
Random Forest (#Trees=500)	72,5 %	75,6 %	65,8 %	70,4 %	73,5 %	72,5 %	73,0 %
Random Forest (#Trees=1,000)	**76,5 %**	76,3 %	**67,5 %**	**71,6 %**	**74,5 %**	**76,5 %**	**75,5 %**
K-NN(K=2)	61,8 %	70,7 %	40,3 %	51,3 %	64,5 %	61,8 %	63,1 %
K-NN(K=5)	63,5 %	64,9 %	58,9 %	61,8 %	63,6 %	63,5 %	63,5 %
K-NN(K=10)	64,3 %	69,6 %	50,8 %	58,7 %	65,4 %	64,3 %	64,8 %
J48 (Confidence Factor=0,1)	63,9 %	64,2 %	62,9 %	63,5 %	63,9 %	63,9 %	63,9 %
J48 (Confidence Factor=0,5)	62,4 %	62,4 %	62,4 %	62,4 %	62,4 %	62,4 %	62,4 %
J48 (Confidence Factor=1,0)	62,3 %	62,3 %	62,4 %	62,3 %	62,3 %	62,3 %	62,3 %
SVM (Gamma=0,5)	50,3 %	**99,3 %**	0,6 %	1,2 %	74,7 %	50,3 %	60,1 %
SVM (Gamma=1,0)	52,4 %	97,5 %	1,4 %	2,8 %	76,4 %	52,4 %	62,2 %

side, the average recall value of Naive Bayes, 57.7% , shows that the method can recall about 80% of legitimate users.

The results of the third baseline ,"C", when using 55 content features exploited in the literature are worse than the user features in terms of all metrics. Although

of the great number of content features, their performance shows that the content features designed in the literature cannot effectively model the behavior of social spammers. Also, in our case, the combination of weak features have not produced a strong classification model. This behavior occurs probably because of possible correlation among features. Similarly to the results of baseline "B", the Random Forest with $\#Trees = 1000$ is almost the best one in all metrics.

Behavioral Features Results. Our behavioral features, which are 10 features (one by Writing Style Similarity, one by Language Model-Based Tweets Similarity, four by Posting Behavior, and four by Posting Diversity), perform better than the three baselines ("A","B","C") in most performance metrics. As expected, Random Forest has a classification accuracy more than 78%. As SVM has the highest social spammer class precision and almost the lowest recall, using the F-measure metric is the right way to compare with other baselines since this metric combines the precision and recall metrics. Thus, compared to the other baselines, the F-measure of our behavioral features is higher than other two kinds of features by $5\% \sim 7\%$. Differently from baselines' results, the performance of SVM when using our features is completely different at the recall level. The use of RBF (Gaussian) kernel with γ of 0.5 gives an indication that the distribution of our features for social spammer class might be correlated with the Gaussian distribution and thus such a knowledge might help in building unsupervised advanced models like Gaussian mixture model. Finally, the results of our features ensure our hypothesis about the need to focus on designing features that model the behavior of social spammers.

All Features Results. To provide more insights into the performance of our features, we report the results when using our features with the user and content features. Unfortunately, we observe a slight degradation in the accuracy, recall and F-measure of social spammer class, while we expect the opposite behavior exactly. In the machine learning world, this phenomenon appears when the classification model over-fits the input data. In other words, the increasing of the features dimension has separated well the examples of both classes. However, separating well examples of different classes is not always something perfect since the learning algorithms build a model for some examples while those examples are truly noise.

False Positive v.s. High Quality. As our main problem has direct relation with the information quality field, it is necessary to discuss the results from quality point of view. As known in the spam email filtering, the efforts are directed towards the false positive problem that occurs when a truly "non-spam" email is classified as "spam". However, in the context of social spam, the false positive problem is less important because of the availability of large-scale data collections, meaning that classifying "non-spam" account as a "spam" one is not a serious and relevant problem to worry about. Thus, the attention is turned in social networks context to increase the quality of data where a wide range of Twitter based applications (e.g., tweets summarization) has a high priority to work on noise-free collections. For choosing the appropriate model that can effectively filter out spam accounts, the recall of social spammer class is the right metric that must be considered at models selection step. Thus, the J48 with

confidence factor of 0.1 produces the best model which gives the highest recall when leveraging our 10 behavioral features. F-measure of social spammer class comes after the recall metric since it considers both recall and precision in the computation. In the case of F-measure metric, the classification model that is produced by the Random Forest learning method with $\#Trees = 1000$ when adopting our 10 behavioral features.

As the computational time aspect is significant when targeting large-scale collections, the extraction of our features is completely suitable to process large-scale collections with providing high quality collections. For instance, the time required to process our Twitter data-set is no more than few hours, distributed between crawling data from Twitter (Top 100 tweets) and features extraction, and predicting the class label of each account using an already learned classification model. Although of our features are suitable for handling large-scale collections, they are not suitable for real-time detection because of the need for information from Twitter's servers to process each Twitter account (user).

6. Conclusion

In this paper, we have approached the problem of filtering out social spammers existing in large-scale Twitter data collections. We have introduced a design of new features which focus on modeling the social spammers' behaviors, after analyzing deeply a large set of spam accounts. The experimental results show that reducing the social spam problem starts from understanding first the behavior of social spammers in posting spam content. Thus, the simple design of our new 10 behavioral-based features has performed better than 75 features introduced in the state-of-the-art. We cannot conclude that our features have completely solved the social spam problem; however, our features are able to detect the Twitter social spammers that have surly and massive behavior in polluting Twitter content. As a future work, we intend to apply unsupervised learning approach methods with designing more robust features.

Bibliographie

Abdelhamid C., Mohand B., Bernard D. (2016). Multi-criterion real time tweet summarization based upon adaptive threshold. In *2016 IEEE/WIC/ACM international conference on web intelligence, WI 2016, omaha, ne, usa, october 13-16, 2016*, p. 264–271.

Agarwal N., Yiliyasi Y. (2010). Information quality challenges in social media. In *International conference on information quality (iciq)*, p. 234-248.

Ahmed F., Abulaish M. (2013). A generic statistical approach for spam detection in online social networks. *Computer Communications*, vol. 36, n° 10, p. 1120–1129.

Almaatouq A., Shmueli E., Nouh M., Alabdulkareem A., Singh V. K., Alsaleh M. *et al. (2016). If it looks like a spammer and behaves like a spammer, it must be a spammer: analysis and detection of microblogging spam accounts.* International Journal of Information Security, *vol. 15, n° 5, p. 475–491.*

Amleshwaram A. A., Reddy N., Yadav S., Gu G., Yang C. (2013). Cats: Characterizing automation of twitter spammers. In *Communication systems and networks (comsnets), 2013 fifth international conference* on, p. 1-10.

Bara I.-A., Fung C. J., Dinh T. (2015). Enhancing twitter spam accounts discovery using cross-account pattern mining. In *Integrated network management (im), 2015 ifip/ieee international symposium on*, p. 491-496.

Cao C., Caverlee J. (2015). Detecting spam urls in social media via behavioral analysis. In *Advances in information retrieval*, p. 703-714. Springer.

Chao C., Jun Z., Yi X., Yang X., Wanlei Z., Mehedi H. M. *et al.* (2015). A performance evaluation of machine learning-based streaming spam tweets detection. *IEEE Transactions on Computational Social Systems*, vol. 2, n° 3, p. 65-76.

Chen C., Zhang J., Xiang Y., Zhou W. (2015). Asymmetric self-learning for tackling twitter spam drift. In *Computer communications workshops (infocom wkshps), 2015 ieee conference on*, p. 208-213.

Chen C., Zhang J., Xiang Y., Zhou W., Oliver J. (2016). Spammers are becoming "smarter" on twitter. IT professional, vol. 18, n° 2, p. 66-70.

Chu Z., Gianvecchio S., Wang H., Jajodia S. (2012). Detecting automation of twitter accounts: Are you a human, bot, or cyborg? *Dependable and Secure Computing, IEEE Transactions on*, vol. 9, n° 6, p. 811-824.

Fabricio B., Gabriel M., Tiago R., Virgilio A. (2010). Detecting spammers on twitter. *In collaboration, electronic messaging, anti-abuse and spam conference (ceas)*, p. 12.

Guo D., Chen C. (2014). Detecting non-personal and spam users on geo-tagged twitter network. *Transactions in GIS*, vol. 18, n° 3, p. 370-384.

Hai W. A. (2010a). 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*, p. 335-342. Berlin, Heidelberg, Springer-Verlag.

Hai W. A. (2010b, July). Don't follow me: Spam detection in twitter. *In Security and cryptography (secrypt), proceedings of the 2010 international conference on*, p. 1-10.

Hall M., Frank E., Holmes G., Pfahringer B., Reutemann P., Witten I. H. (2009, novembre). The weka data mining software: An update. *SIGKDD Explor. Newsl.*, vol. 11, n° 1, p. 10-18.

Hoang T. B. N., Mothe J. (2016). Building a Knowledge Base using Microblogs: the Case of Cultural MicroBlog Contextualization Collection (regular paper). In K. Balog, L. Cappellato, N. Ferro, C. Macdonald (Eds.), *Conference on Multilingual and Multimodal Information Access Evaluation (CLEF)*, Evora, Portugal, 05/09/2016-08/09/2016, vol. 1609, p. 1226-1237. http://CEUR-WS.org, CEUR Workshop Proceedings.

Hu X., Tang J., Liu H. (2014). Online social spammer detection. In *Aaai*, p. 59-65.

Hu X., Tang J., Zhang Y., Liu H. (2013). Social spammer detection in microblogging. In *Ijcai*, vol. 13, p. 2633-2639.

Kullback S., Leibler R. A. (1951). On information and sufficiency. *The annals of mathematical statistics*, vol. 22, n° 1, p. 79-86.

Lee K., Caverlee J., Webb S. (2010). Uncovering social spammers: Social honeypots + machine learning. In *Proceedings of the 33rd international acm sigir conference on research and development in information retrieval*, p. 435-442. New York, NY, USA, ACM.

Manel M., André P., Corinne Amel Z., Ikram A., Florence S. (2014). Analyzing tagged resources for social interests detection. *In ICEIS 2014 - proceedings of the 16th international conference on enterprise information systems*, vol.1, Lisbon, Portugal, 27-30 april, p. 340-345.

Manning C. D., Raghavan P., Schütze H. (2008). *Introduction to information retrieval*. New York, NY, USA, Cambridge University Press.

Martinez-Romo J., Araujo L. (2013). Detecting malicious tweets in trending topics using a statistical analysis of language. *Expert Systems with Applications*, vol. 40, n° 8, p. 2992-3000.

McCord M., Chuah M. (2011). Spam detection on twitter using traditional classifiers. *In Proceedings of the 8th international conference on autonomic and trusted computing*, p. 175-186. Springer-Verlag.

Meda C., Bisio F., Gastaldo P., Zunino R. (2014). A machine learning approach for twitter spammers detection. *In 2014 international carnahan conference on security technology (iccst)*, p. 1-6.

Meda C., Ragusa E., Gianoglio C., Zunino R., Ottaviano A., Scillia E. *et al.* (2016). Spam detection of twitter traffic: A framework based on random forests and non-uniform feature sampling. In *Advances in social networks analysis and mining (asonam), 2016 ieee/acm international conference on*, p. 811-817.

Mezghani M., Zayani C., Amous I., Péninou A., Sèdes F. (2014). Dynamic enrichment of social users' interests. *In IEEE 8th international conference on research challenges in information science, RCIS 2014*, Marrakech, Morocco, May 28-30, 2014, p. 1-11.

Oppenheim A. V. (1999). *Discrete-time signal processing*. Pearson Education India.

Perdana R. S., Muliawati T. H., Alexandro R. (2015). Bot spammer detection in twitter using tweet similarity and time interval entropy. *Jurnal Ilmu Komputer dan Informasi*, vol. 8, n° 1, p. 19-25.

Rocío A., Rose L., Florence S. (2015). Detecting sociosemantic communities by applying social network analysis in tweets. *Social Netw. Analys. Mining*, vol. 5, n° 1, p. 38:1-38:17.

Sedhai S., Sun A. (2016). *An analysis of 14 million tweets on hashtag-oriented spamming*.

Singh M., Bansal D., Sofat S. (2014). Detecting malicious users in twitter using classifiers. *In Proceedings of the 7th international conference on security of information and networks*, p. 247.

Sirinya O., C. Marie-Françoise C., André P., Florence S. (2014). Deriving user's profile from sparse egocentric networks: Using snowball sampling and link prediction. *In 9th international conference on digital information management, ICDIM 2014*, Phitsanulok, Thailand, Sept. 29-oct. 1, p. 80-85.

Stringhini G., Kruegel C., Vigna G. (2010). Detecting spammers on social networks. In *Proceedings of the 26th annual computer security applications conference*, p. 1-9. New York, NY, USA, ACM.

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

Wallace B. C., Small K., Brodley C. E., Trikalinos T. A. (2011). Class imbalance, redux. In *Data mining (icdm), 2011 ieee 11th international conference on*, p. 754-763.

Wang D., Pu C. (2015). Bean: a behavior analysis approach of url spam filtering in twitter. In *Information reuse and integration (iri), 2015 ieee international conference on*, p. 403-410.

Washha M., Qaroush A., Sèdes F. (2016a, novembre). Impact of Time on Detecting Spammers in Twitter (regular paper). In *Gestion de Données Principes, Technologies et Applications (BDA)*, Poitiers, 15-18 novembre 2016, p. 1-10. http://hal.inria.fr, HAL-INRIA.

Washha M., Qaroush A., Sèdes F. (2016b). Leveraging time for spammers detection on twitter. In *Proceedings of the 8th international conference on management of digital ecosystems*, p. 109-116.

Wu T., Liu S., Zhang J., Xiang Y. (2017). Twitter spam detection based on deep learning. In *Proceedings of the australasian computer science week multiconference*, p. 3.

Yang C., Harkreader R., Zhang J., Shin S., Gu G. (2012). 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*, p. 71-80. New York, NY, USA, ACM.

Yang C., Harkreader R. C., Gu G. (2011). Die free or live hard? empirical evaluation and new design for fighting evolving twitter spammers. I*n Proceedings of the 14th international conference on recent advances in intrusion detection*, p. 318-337. Berlin, Heidelberg, Springer-Verlag.

Yardi S., Romero D., Schoenebeck G., boyd danah. (2009). Detecting spam in a twitter network. First Monday, vol. 15, n° 1. Consulté sur http:// firstmonday.org/ ojs/ index.php/ fm/ article/ view/ 2793

Zhang X., Zhu S., Liang W. (2012). Detecting spam and promoting campaigns in the twitter social network. *In Data mining (icdm), 2012 ieee 12th international conference on*, p. 1194-1199.

Zi C., Indra W., Haining W. (2012). Detecting social spam campaigns on twitter. In *Applied cryptography and network security*, p. 455-472.