Information Quality in Social Networks: Predicting Spammy Naming Patterns for Retrieving Twitter Spam Accounts

Mahdi Washha[1], Aziz Qaroush[2], Manel Mezghani[1] and Florence Sedes[1]
[1]*IRIT Laboratory, University of Toulouse, Toulouse, France*
[2]*Department of Electrical and Computer Engineering, Birzeit University, Ramallah, Palestine*

Keywords: Twitter, Social Networks, Spam.

Abstract: The popularity of social networks is mainly conditioned by the integrity and the quality of contents generated by users as well as the maintenance of users' privacy. More precisely, Twitter data (e.g. tweets) are valuable for a tremendous range of applications such as search engines and recommendation systems in which working on a high quality information is a compulsory step. However, the existence of ill-intentioned users in Twitter imposes challenges to maintain an acceptable level of data quality. Spammers are a concrete example of ill-intentioned users. Indeed, they have misused all services provided by Twitter to post spam content which consequently leads to serious problems such as polluting search results. As a natural reaction, various detection methods have been designed which inspect individual tweets or accounts for the existence of spam. In the context of large collections of Twitter users, applying these conventional methods is time consuming requiring months to filter out spam accounts in such collections. Moreover, Twitter community cannot apply them either randomly or sequentially on each user registered because of the dynamicity of Twitter network. Consequently, these limitations raise the need to make the detection process more systematic and faster. Complementary to the conventional detection methods, our proposal takes the collective perspective of users (or accounts) to provide a searchable information to retrieve accounts having high potential for being spam ones. We provide a design of an unsupervised automatic method to predict spammy naming patterns, as searchable information, used in naming spam accounts. Our experimental evaluation demonstrates the efficiency of predicting spammy naming patterns to retrieve spam accounts in terms of precision, recall, and normalized discounted cumulative gain at different ranks.

1 INTRODUCTION

Online social networks (OSNs) have become the top communication media and almost the first option for users to share links, discuss, and connect with others. However, the existence of easy interactive interfaces and low barriers to publication have caused various information quality (IQ) problems (e.g. Social spam, Rumor) in OSNs, making the obtaining of accurate and relevant information a challenge. As an example, Twitter platform is one of many popular social networks, which has distinctive services not available in the same power in other social networks. Indeed, these services have properties revolving around: (i) delivering posts (tweets) in a real-time manner; (ii) inserting URL(s) pointing out to external source(s) of information; (iii) adding hashtags in tweets to group the similar tweets to facilitate the search process; (iv) providing a real-time search service to retrieve tweets; (v) opening new Twitter accounts without imposing restrictions from verification point of view (Benevenuto et al., 2010).

The openness of OSNs and the lack of effective restrictions have attracted a special kind of unethical and ill-intentioned individuals well known as "spammers". They misuse OSNs' services to publish and spread misleading, fake, and out of context information. A wide range of goals drives spammers to publish spam content, summarized in (Benevenuto et al., 2010): (i) spreading advertisements to generate sales and gain illegal profits; (ii) disseminating porn materials; (iii) publishing viruses and malwares; (iv) creating phishing websites to reveal sensitive information.

Performing spamming tasks can constitute major problems in different areas, not limited to: (i) polluting search results by spam posts or tweets; (ii) degrading the accuracy of statistics obtained by mining tools; (iii) consuming storage resources; (iv) and vi-

Washha, M., Qaroush, A., Mezghani, M. and Sèdes, F.
Information Quality in Social Networks: Predicting Spammy Naming Patterns for Retrieving Twitter Spam Accounts.
DOI: 10.5220/0006314006100622
In *Proceedings of the 19th International Conference on Enterprise Information Systems (ICEIS 2017) - Volume 2*, pages 610-622
ISBN: 978-989-758-248-6

olating user's privacy. Hence, because of the importance of OSNs data in various areas such as search engines and research fields, the optimal solution is filtering out the noisy data to have high quality information. Information quality process in social networks can be summarized in three main steps: (i) selecting the data collection (e.g. Facebook accounts, Tweets, Facebook posts) that needs improvement; (ii) determining the noise type (e.g. spam, rumor) to be filtered out; (iii) at last, applying pre-designed algorithms depending on the chosen noise type to produce a new noise free data collection.

Motivation and Problem. We focus on treating a special issue related to social spam problem to be a complementary solution with our team researches on social networks. Our team has researches(Abascal-Mena et al., 2015; Mezghani et al., 2014; Abascal-Mena et al., 2014; Mezghani et al., 2015; Abascal-Mena et al., 2015; On-at et al., 2016) addressing a wide range of social networks problems like social profiling, profile enrichment, social interests detection, and sociosemantic communities detection, in which Twitter platform has been adopted in performing experiments and validations. Subsequently, experimenting on an acceptable quality of data collections is highly required to have high accurate and precise results.

In the battle of fighting spam, a considerable set of methods (Wang, 2010; Benevenuto et al., 2010; Lee et al., 2010; McCord and Chuah, 2011; Stringhini et al., 2010; Yang et al., 2011; Amleshwaram et al., 2013; Cao and Caverlee, 2015; Chu et al., 2012b; Meda et al., 2014; Santos et al., 2014; Martinez-Romo and Araujo, 2013) has been designed for detecting either spam accounts or spam campaigns, with little attention dedicated toward spam tweets detection. These conventional methods are mainly grounded on exploiting the features extraction concept combined with supervised machine learning algorithms to build a predictive model using an annotated data-set. However, performing these methods on large collections consisting of millions of Twitter users (or accounts) is time consuming, requiring months to process "crawled" large collections. Indeed, the use of Twitter REST APIs[1] causes this bottleneck problem, where no alternative way exists to collect (or retrieve) complete information about users. More precisely, Twitter imposes unavoidable restrictions and limitations on the number of calls determined according to the functionally of the used REST API. For instance, retrieving information for half million of users, including the meta-data of users' followers and followees, may take approximately three months. Also, Twitter itself may avoid performing the conventional detection methods on each registered user (or account) either in a sequential or a random manner. We justify this avoidance because of the dynamicity of network in tracking daily updates (e.g. new accounts) on it.

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

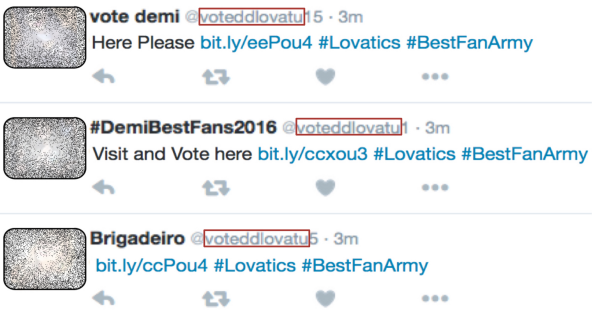

Figure 1: Illustrative example of spam tweets posted by different users having common screen name pattern "voteddlovatu".

Searchable Information. The conventional detection methods don't provide information more than the class label of the considered user (spammer or legitimate user) or tweet (spam or non-spam). Thus, to make the detection process more systematic and faster, having searchable information (i.e. string patterns used in naming accounts) can contribute in retrieving the accounts that have high probability for being spam. For instance, the spam tweets illustrated in figure 1 show a particular pattern "voteddlovatu" exploited in creating and launching a spam bot. Indeed, such a string pattern can be used as a query to retrieve all accounts that contain this string pattern to apply then the conventional detection methods. It is important to note that spammers avoid using random or rubbish names since they automate the creation of Twitter accounts where the IDs must be unique. Thus, to ensure the uniqueness, spammers define the IDs through using an unique pattern combined with a systematic simple counting (e.g. "15", "1", and "5") as illustrated in the given example. Also, as one of spammers' principles, the name of accounts must be attractive for normal users (legitimate users) and therefore using random or rubbish names is not the solution at all.

Contributions. In this paper, we aim at predicting spammy naming patterns exploited in establishing accounts, where those patterns are used as searchable information query for retrieving spam accounts. More precisely, we leverage the meta-data of users (or accounts) posted tweets associated with topics (e.g. hashtags). Then, we predict the spammy naming patterns through passing the considered set of users into three consecutive stages. The first stage

detects the communities that the users are belonging to using heuristic information extracted from the users' meta-data as an estimation for the social connections among users. In the second stage, each community is represented by two sets of string patterns elicited from the screen name and user name attributes. At last, each community is labeled as a spam bot through measuring the degree of similarity between the distribution of corresponding string patterns and the uniform distribution of patterns with classifying the string patterns as spam in case of high dissimilarity among distributions. We demonstrate the effectiveness of spammy string patterns through a series of experiments conducted on a crawled and an annotated data-set containing more than half million tweets collected from 50 trending hashtags. The experimental results show that the spammy patterns of the screen name and user name attributes have superior performance in terms of three different information retrieval metrics (Precision@L, Recall@L, Normalized Discounted Cumulative Gain@L), compared to five proposed baselines. With the results obtained, our method can be leveraged in different ways:

- For a crawled data collection, our method can be applied to retrieve the accounts that have high probability for being spam in the target collection. Then, the conventional detection methods can be performed to process the top accounts (e.g. top 1,000), speeding-up the filtration process without needing to examine each account in the collection.
- Twitter can integrate our spam retrieval method with its anti-spam mechanism to search for spam accounts in a systematic and fast way.

The remainder of the paper is organized as follows. Section 2 presents the Twitter's anti-spam mechanism as well as the spammer detection methods proposed in the literature. Section 3 presents the notations, problem formalization, and the design of the three stages to predict the spammy naming patterns. Section 4 details the data-set used in experimenting and validating our approach. The experimental setup and a series of experiments evaluating the proposed approach are described in section 5. At last, section 6 concludes the work with offering directions for future work.

2 BACKGROUND AND RELATED WORK

- Spammers are goal-oriented persons targeting to achieve unethical goals (e.g. promote products), and thus they use their smartness to accomplish their spamming tasks in an effective and a quick way.
- Spammers often create and launch a bot (group) of spam accounts in a short period (e.g. in one day), to maximize their monetary profit and speedup their spamming behavior.
- As a set of REST APIs is provided by social networks, spammers leverage them to automate their spamming tasks in a systematic way (e.g. tweet every 10 minutes). Indeed, they avoid the random posting behavior because it may decrease the target profit and decelerate their spamming behavior.

Twitter's Anti-Spam Mechanism. Twitter fights spammers through allowing users to report spam accounts simply by clicking on "Report: they are posting spam" option available on the account page. When an user reports a particular account, Twitter's administrators review manually the reported account to make the suspension decision. However, adopting such a method for combating spammers needs a great effort from both users and administrators. Moreover, not all reports are trustworthy, meaning that some reported accounts are for legitimate users, not spammers. Besides the manual reporting mechanism, Twitter has defined some general rules (e.g. not allowed to post porn materials) for public to reduce the spam problem as possible with suspending permanently the accounts that violate those rules (Twitter, 2016). However, Twitter's rules are easy to bypass by spammers. For instance, spammers may coordinate multiple accounts with distributing the desired workload among them to mislead the detection process. Doing so, the separated accounts tend to exhibit invisible spam behavior. Thus, these shortcomings have motivated researchers to propose more powerful methods for the applications that use Twitter as a source of information. Hence, we categorize the spam detection approaches dedicated for Twitter 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 interaction.

Machine Learning Approach. In this approach, researchers 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 and Araujo, 2013) identified spam tweet using probabilistic language models to determine the topic of the considered tweet. Using statistical features as a representation for the tweet object, Benevenuto (Benevenuto et al., 2010) identified spam tweet only by

leveraging some features extracted from the tweet text such as the number of words and the number of characters. Then, the well-known SVM learning algorithm has been applied on a manually created data-set to learn a binary classifier. Indeed, the main strength of tweet level is the fast detection from time computation point of view. However, adopting supervised learning approach to have a fixed classification model all the time is not an effective solution because of the high evolving and changing in the spam content overtime. In other words, hundred millions of ground truth spam tweets are required to have robust model, which is not possible to have such model.

Account Level. The works of (Wang, 2010; Benevenuto et al., 2010; McCord and Chuah, 2011; Stringhini et al., 2010) turned the attention toward account features, including the number of friends, number of followers, similarity between tweets, and ratio of URLs in tweets. In more dedicated studies, the work proposed in (Cao and Caverlee, 2015) identifies the spam URLs through analyzing the shorten URLs behavior like the number of clicks. However, the ease of manipulation in the account features by spammers gives a motivation to extract more complex features using graph theory. For instance, the authors of (Yang et al., 2011; Yang et al., 2012) examined the relation between users using some graph metrics to measure three features, including the 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 Twitter based application because of the huge volume of data that must be retrieved from Twitter's servers.

Campaign Level. Chu et al. (Chu et al., 2012b) treated the spam problem from collective perspective point of view. They clustered the desired accounts according to the URLs available in the posted tweets, and then a defined set of features is extracted from the clustered accounts to be incorporated in identifying spam campaign using machine learning algorithms. Chu, Gianvecchio, Wang, and Jajodia (Chu et al., 2012a) proposed a classification model to capture the difference among bot, human, and cyborg with considering the content of tweets, and tweeting behavior. Indeed, the major drawback in the accomplished methods at campaign level is the relying intensively on the features requiring too many API calls on Twitter's servers to obtain information like users' tweets and followers. Indeed, this makes such solutions not scalable for huge number of users (or accounts).

Beyond the features design level, (Hu et al., 2014; Hu et al., 2013) introduced an optimization framework which uses the content of tweets and basic network information to detect spammers using efficient online learning approach. However, the main limitations of such works are the need of information about the network, raising the problem of scalability again.

Honeypot Approach. Social honeypot is viewed as an information system resource that can monitor 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 major difference between the Twitter's anti-spam mechanism and social honeypot approach. Both of them need administration control to produce final decision about the accounts that fall in the honeypot trap. The necessity of administration control is to reduce the false positive rate, as an alternative solution for classifying blindly all users dropped in the trap as spammers.

3 SPAMMY NAMING PATTERNS PREDICTION

In this section, we introduce notations, definitions, and the formalization of the problem that we address. Then, we present the design of our method to predict the spammy naming patterns.

3.1 Terminology Definition and Problem Formalization

Twitter uses hashtags or keywords concept as a possible way for topic modeling to group similar tweets together. Indeed, spammers leverage the hashtags, especially the trending ones, to publish spam content. More particularly, spammers attack trending hashtags through launching spam campaigns or bots to deliver the spam content as fast as possible. Hence, intuitively, as extracting spammy naming patterns require a collection of users having high probability for being operated as a spam bot, we exploit the concept of hashtags as an entry point to predict the spammy naming patterns. Thus, since each hashtag consists of tweets posted by different users, we model the hashtag as a finite set of distinct users, without going in the representation of tweet object, defined as $Hashtag(U_H) = \{u_1, u_2, ...\}$, where the user element $u_\bullet$ is further defined by 3-tuple $u_\bullet =< UN, SN, UA >$. Each element inside the tuple is described as follows:

User Name (UN). Twitter, as other social networks, allows users to name their accounts with maximum length of 20 characters. Users can use whitespace, symbols, special characters, and numeric numbers in

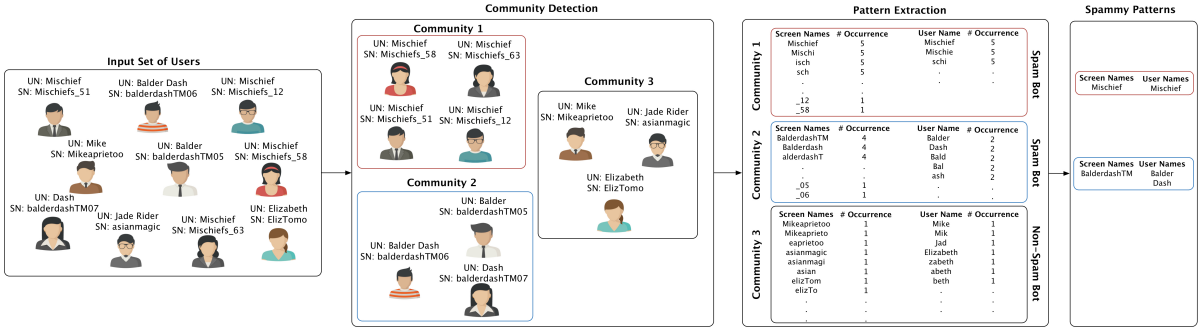

Figure 2: An example describing the main stages adopted for predicting spammy naming patterns from the user name and screen name attributes. The predicted patterns such as "Mischief" might be leveraged for searching for spam accounts on Twitter.

filling the user name attribute. Twitter provides also a facility for the users to use duplicated names, and thus this field is not necessary to be unique. This attribute can be modeled as a set of ordered characters, defined as $UN = \{< 1, a_1 >, ..., < i, a_i >\}$ where i is the position of the a_i character in the name.

Screen Name (SN). This attribute is a mandatory field and it must be filled at the creation time of the account. Users must choose an unique name not used previously by other users, and with maximum length of 16 characters. Besides, Twitter constrains the space of allowed characters to alphabetical letters, numbers, and "_" character. Similarly to the user name attribute, we model this field as an ordered set of characters, defined as $SN = \{< 1, a_1 >, ..., < i, a_i >\}$ where i is the position of the character a_i in the attribute.

User Age (UA). When a user creates an account on Twitter, the creation date is registered on Twitter's servers without providing permissions to modify it in the future. We exploit the creation date, as an accessible and available property in the user object, to compute the age of the account. Formally, we calculate the age in days time unit through subtracting the current time ($Time_{now}$) from the creation date of the account, define as $UA = Time_{now} - Time_{creation}$.

Problem Formalization. Given a set of Tweets T associated with a particular hashtag, posted by a set of distinct users U_H such that $|U_H| \leq |T|$, our main problem is to infer and discover a set of patterns P used in naming spam accounts as a searchable information, without requiring any prior knowledge in advance such as the relation between users (e.g. followers and followees of users). More formally, we aim at designing a hypothesis function y such that it processes and handles the given set of users U_H to predict spammy naming patterns, defined as $y : U_H \rightarrow \{p_1, p_2, ...\}$ where $p_\bullet$ is a set of ordered characters.

3.2 Model Design

In the literature, Freeeman (Freeman, 2013) designed a probabilistic method to classify the profiles of LinkedIn platform into binary classes (spam and non-spam) using the account name only. The key idea of the work is based on building a feature vector as a representation for each account using N-gram method. Then, for a given profile, a Bayes' rule is applied to decide whether the account is spam or not with estimating the prior probability component through an annotated data-set. However, this approach is not an adoptable solution for predicting the spammy patterns due to the following reasons: (i) using N-gram directly on large number of users to extract potential patterns increases the computational time, making the direct use of N-gram not a practical solution; (ii) Spammers' behaviors in social networks are dynamic either in the spam content or in using string patterns to create spam bots, and thus adopting a fixed feature vector (i.e. vector of bag of words) is not suitable for modeling all spammy naming patterns.

Hence, we address the issue of predicting spammy naming patterns in an unsupervised way through a 3-step approach, described through an example shown in Figure 2. First, for a given set of users, we detect the communities that the users are belonging to through leveraging the user name, screen name, and user age meta-data. Detecting communities might be viewed as a preprocessing step to cluster and group the users that have common features (e.g. matched names, similar accounts age), aiming to speed-up the pattern extraction process. In the second step, for each community being detected, we extract a set of potential patterns, as a representation for each community, using N-gram method. At last, the probability distribution of each patterns set is compared with the uniform probability distribution of the considered patterns set, to make a decision about the type (spammy

o non-spammy) of those patterns.

In the following, we introduce a mathematical model adopted for community detection. Then, we describe the proposed method in extracting string patterns from user name and screen name attributes. At last, we present our method by which the spammy naming patterns are predicted.

3.2.1 Community Detection

As the topics or hashtags might be attacked by various and different spam bots, more than one spammy naming pattern might be leveraged in creating spam bots. Also, the probability of using same patterns in many spam bots is quite low since no relation exists among spammers. Therefore, performing community detection can contribute in distinguishing between different spam bots where each bot can be viewed as a community.

In this paper, we exploit the use of non-negative matrix factorization (NMF) method to infer the communities structure because of its outstanding performance in clustering (Yang and Leskovec, 2013). NMF works through partitioning an information matrix into hidden factor matrices, defined mathematically as an optimization minimization problem:

$$\min_{H\geq 0} ||\mathbf{X} - \mathbf{H}\mathbf{H}^T||_F^2 \quad (1)$$

where $||\bullet||_F$ is the Frobenius norm of the considered matrix, $\mathbf{X} \in R^{|U_H|\times|U_H|}$ is an information matrix representing the strength of social connections between users, $\mathbf{H} \in R^{|U_H|\times K}$ is the community structure hidden factor matrix of K communities. More precisely, the entry $X(i,j)$ reflects the strength of the social connection between the $u_i \in U_H$ user and $u_j \in U_H$ user. The entry $H(i,j)$ in the hidden factor matrix can be interpreted as the confidence degree of user $u_i \in U$ belonging to the j^{th} community. It is important to mention that each user belongs to one community only, not more than one.

Obviously, inferring the hidden matrix $\mathbf{H}$ requires a formal definition of the information matrix $\mathbf{X}$. For example, $\mathbf{X}$ might be an adjacency matrix representing the social connections or links between the users of the given set U_H. However, obtaining the adjacency matrix in our case is not possible because the available information about users are limited to simple metadata describing the account of each user without providing information about the followers and followees. Hence, in this paper, we leverage the available and accessible information to estimate social connections among users through proposing three definitions of the information matrix $\mathbf{X}$ denoted as $\mathbf{X}^{SN}$, $\mathbf{X}^{UN}$, and $\mathbf{X}^{UA}$, where each of which is formally defined as follows:

Screen Name Similarity ($\mathbf{X}^{SN}$). As the screen name field must be unique, spammers tend to adopt a particular fixed pattern when creating multiple accounts to act as a spam bot. For instance, in Figure 1, the spammer has adopted the name "voteddlovatu" as a fixed pattern for the screen name field. Intuitively, the high overlapping or matching in the screen name among users increases the probability of the users to belong to the same community. Therefore, we define the information matrix $\mathbf{X}^{SN}$ to measure the degree of matching in the screen name attribute. More precisely, given two users $u_i, u_j \in U$, the degree of matching for a particular entry in the matrix $\mathbf{X}^{SN}$ is defined as:

$$\mathbf{X}^{SN}(i,j) = \frac{max\{|m| : m \in Patterns)\}}{min(|u_i.SN|, |u_j.SN|)}$$

$$Patterns = \bigcup_{N\in Max} N-gram(u_i.SN) \cap N-gram(u_j.SN) \quad (2)$$

where $|\bullet|$ is the cardinality of the considered set, $Max = \{1,...,min(|u_i.SN|, |u_j.SN|)\}$ is a set consisting of positive integers representing the potential number of characters that have overlapping between the names, $N-gram(\bullet)$ is a function returning a set of contiguous sequence of characters for the given name (set of ordered characters) based on the value of N. For better understanding, the 3-gram (or tri-gram) of this screen name "vote" is $\{\{<1,v>,<2,o>,<3,t>\},\{<1,o>,<2,t>,<3,e>\}\}$. The above definition is able to detect the matched pattern wherever it appears in the screen name attribute. For instance, let "vote12" and "tovote" be screen names for two different users, the degree of matching according to equation 2 is around $(\frac{4}{6})66.6\%$ resulting from the use of pattern "vote", regardless the position of the pattern.

User Name Similarity ($\mathbf{X}^{UN}$). Differently from the screen name attribute, spammers can duplicate the entry of user name attribute as many they desire. They exploit representative (not random) names to attract the normal users. Therefore, the full or partial matching among users in such attribute increases the performance of community detection. We define the information matrix $\mathbf{X}^{UN}$ to catch the degree of similarity among users in the user name attribute. Formally, given two users $u_i, u_j \in U$, the degree of similarity is defined as:

$$\mathbf{X}^{UN}(i,j) = \frac{max\{|m| : m \in Patterns)\}}{min(|u_i.UN|, |u_j.UN|)}$$

$$Patterns = \bigcup_{N\in Max} N-gram(u_i.UN) \cap N-gram(u_j.UN) \quad (3)$$

where here $Max = \{1,...,min(|u_i.UN|, |u_j.UN|)\}$.

Age Closeness ($\mathbf{X}^{UA}$). As the spammers automate the process of creating Twitter accounts to act as a spam bot, few hours are enough to create hundreds of accounts. Obviously, the common property among these accounts is the age (creation time) of accounts as one possible way to distinguish between different spam bots. We leverage the closeness in the age of accounts as an additional factor to detect communities. More formally, for a given two users $u_i, u_j \in U$, we compute the difference in the age between each pair of users where the corresponding entry in the information matrix $\mathbf{X}^A$ is defined as

$$\mathbf{X}^{UA}(i,j) = \frac{|u_i.age - u_j.age|}{max\{u_1.age - u_2.age | u_1, u_2 \in U_H\}} \quad (4)$$

The key point of performing normalization is to range the entire values between 0 and 1 where the increasing in the value means that the pair of accounts was not created in the same time.

Combining Information Matrices. With these three information matrices, NMF method allows to integrate them together in the same objective function. Thus, the new version of the objective function is defined as:

$$\min_{H \geq 0} ||\mathbf{X}^{SN} - \mathbf{H}\mathbf{H}^T||_F^2 + ||\mathbf{X}^{UN} - \mathbf{H}\mathbf{H}^T||_F^2 + ||\mathbf{X}^{UA} - \mathbf{H}\mathbf{H}^T||_F^2 \quad (5)$$

Obviously, equation 5 infers the hidden factor matrix H to represent the consistent community structure of the users.

Optimization Method. The objective function is not jointly convex and no closed form solution exists. Hence, we propose the use of gradient descent as an alternative optimization approach. As we have one free variable ($\mathbf{H}$), the gradient descent method updates it iteratively until the variable converge.

Formally, let $\mathcal{L}(\mathbf{H})$ denotes to the objective function given in equation 5 . At the iteration τ, the updating equation is given as:

$$\begin{aligned}\mathbf{H}^{\tau} &= \mathbf{H}^{\tau-1} - \eta . \frac{\partial \mathcal{L}(\mathbf{H}^{\tau-1})}{\partial(\mathbf{H})} \\ &= \mathbf{H}^{\tau-1} - 2\eta\big(6\mathbf{H}^{\tau-1}(\mathbf{H}^{\tau-1})^T\mathbf{H}^{\tau-1} \\ &\quad - (\mathbf{X}^{SN} + \mathbf{X}^{UN} + \mathbf{X}^{UA})\mathbf{H}^{\tau-1} \\ &\quad - ((\mathbf{X}^{SN})^T + (\mathbf{X}^{UN})^T + (\mathbf{X}^{UA})^T)\mathbf{H}^{\tau-1}\big) \quad (6)\end{aligned}$$

where the parameter η denotes to the gradient descent step in updating the matrix $\mathbf{H}$. We assign the value of η to a small constant value (i.e. 0.05). As the gradient descent method is an iterative process, a stop condition is required in such a case. Thus, we exploit two stop conditions: (i) the number of iterations, denoted as M; (ii) and the absolute change in the H matrix in two consecutive iterations to be less than a threshold, i.e. $|(||H^{\tau}||_F - ||H^{\tau-1}||_F)| \leq \varepsilon$.

3.2.2 String Patterns Extraction

Once the K defined communities are inferred through finding the hidden factor matrix $\mathbf{H}$, the next stage is to extract naming patterns corresponding to each inferred community. Before going on, we introduce a definition for the i^{th} community, defined by 6-tuple $C_i = < U, PT^{SN}, PT^{UN}, P_D^{SN}, P_D^{UN}, SN_{Spam}, UN_{Spam}, L >$, where

- U is a set of users belonging to the i^{th} such that $C_i.U \cap C_j.U = \emptyset$, $\forall i, j \in \{1, 2, ..., K\}$, $i \neq j$, and $\bigcup_i C_i.U = U_H$.
- PT^{SN} and PT^{UN} are two finite sets of string patterns extracted from the screen name and user name attributes, respectively, of the users of i^{th} community.
- P_D^{SN} and P_D^{UN} represent the probability distributions of screen name and user name string patterns, respectively. We model each probability distribution as a finite set of 2-tuple object, defined as $P_D^{\bullet} = \{< p_1, v_1 >, < p_2, v_2 >, ...\}$, where $p_{\bullet}$ is a string pattern and $v_{\bullet} \in [0, 1]$ represents the occurrence probability of the string pattern $p_{\bullet}$.
- SN_{Spam} and UN_{Spam} are two string patterns where each represents the spammy pattern of screen name and user name attributes, respectively. The value of these two strings are set at the spammy patterns identification step (next sub-section). Also, these two strings are set to empty in case the i^{th} community is labeled as non-spam.
- L is the class label of the i^{th} community, $L \in \{Spam, non - Spam\}$

For string patterns extraction step, we apply the $N-gram$ character method on each name (screen name or user name) in the inferred community. As spammers may define patterns varying in both length and position, to catch all or most potential patterns, we use different values of N ranging from *three* to the length of the name. We avoid the values *one* and *two* since it is meaningless to have one or two characters pattern. Formally, for a given name in form of $Name = \{< 1, a_1 >, < 2, a_2 >, ...\}$, the potential patterns of which are extracted by

$$Patterns(Name) = \bigcup_{N \in Max} N - gram(Name) \quad (7)$$

where $Max = \{3, ..., |Name|\}$ is a finite set of possible values of N.

With the introduced definition, the string patterns

sets of i^{th} community are given as:

$$C_i.PT^{SN} = \bigcup_{u \in C_i.U} Patterns(u.SN)$$
$$C_i.PT^{UN} = \bigcup_{u \in C_i.U} Patterns(u.UN) \quad (8)$$

3.2.3 Spammy Pattern Identification

After extracting the corresponding patterns for each community, we have to identify the spammy naming patterns either in screen name or user name attributes. Thus, we have to define a classification function predicting the spammy pattern strings. To do so, we examine first each community through leveraging its patterns to decide whether the community is a spam or not. However, as no alternative way to identify spammy patterns, it is *not* necessary that all users in the spam community are spammers. We justify this issue because of the lack information (e.g. followers) about users to make accurate community detection. Moreover, since no exact solution exists to find the optimal hidden matrix **H**, the iterative method used may not find the global minimal hidden matrix **H**.

Hence, to make such a decision about each community detected, we propose to compare the probability distribution of patterns with the uniform probability distribution of patterns. For instance, for a particular community, let $PT^{SN} = \{$"*mischief*","*isch*","_12","_14"$\}$, $PD^{SN} = \{<$ "*mischief*", $0.7>, <$ "_15", $0.1>, <$ "_14", $0.1>, <$ "_12", $0.1>\}$ be a set of screen name patterns with its probability distribution, the uniform probability distribution of these patterns will be $\{<$ "*mischief*", $0.25>, <$ "_15", $0.25>, <$ "_14", $0.25>, <$ "_12", $0.25>\}$. The intuition behind this proposition is that each spam community is biased toward a particular pattern in naming accounts and thus its probability is high compared to the other few patterns belonging to the same set. More precisely, the probability distribution of patterns in spam communities is far from being uniform. On the other side, the distribution of patterns in non-spam ones is close for being uniform. We quantify the similarity between distributions through performing cross-correlation between the probability distribution of patterns set associated with a community and the corresponding uniform distribution of the considered patterns set. Formally, we compute the probability distribution similarity (PDS) through the following generic formula:

$$PDS(P_D) = \frac{Area(P_D \star P_D^{uniform})}{Area(P_D^{uniform} \star P_D^{uniform})} \quad (9)$$

where $P_D \in \{C_i.P_D^{SN}, C_i.P_D^{UN}\}$ is the probability distribution of user name or screen name patterns of i^{th} community, $P_D^{uniform}$ is the uniform probability distribution of the considered patterns set, and $Area(\bullet)$ is a function computing the area under the new resulting distribution by the correlation operation. The purpose of normalizing the area of cross-correlation by the auto-correlation of uniform distribution is to range the value between 0 and 1. The high value of degree similarity means that the considered community has high probability for being non-spam bot. One worthy note is that the all users of the communities that are classified as "spam" might not be spammers (spam accounts). This problem might occur because of the difficulty of finding the optimal hidden matrix **H**. Furthermore, predicting the *true* spam communities accurately is not possible since the available information about each user (or account) is no more than simple meta-data.

We exploit the degree of similarity as one possible way to label each community as spam or non-spam. However, intuitively, the degree of similarity given here is not a crisp value and thus it is required to define a cut-off point to discriminate among the spam and non-spam communities (bots). Thus, for a given community C_i, we propose the following cut-off point by which the considered community is classified into spam and non-spam.

$$C_i.L = \begin{cases} spam & min(PDS(C_i.P_D^{UN}), PDS(C_i.P_D^{SN})) \leq Th \\ non-spam & otherwise \end{cases}$$

where Th is a global threshold value fixed (i.e. 0.1), and determined experimentally for all communities. The key idea of taking the minimum is that spammers may use patterns in one of the considered attributes (screen name or user name), not necessary in both of them.

In the last step, for each community labeled as spam, we choose one pattern from each string patterns set (screen name and user name sets). We select the string pattern that has highest probability value among other patterns in the same set. The longest pattern is chosen, in case of two or more patterns have the same probability value.

4 DATA-SET DESCRIPTION AND GROUND TRUTH

The data-sets used at tweet level detection (Benevenuto et al., 2010; Martinez-Romo and Araujo, 2013) are not publicly available for research use. Also, for privacy reasons, the researchers of social

Table 1: Statistics of the data-set.

Number of Trending Topics	50
Number of Users	210,315
Number of Tweets	567,675

Table 2: Statistics of the annotated users (accounts) and tweets.

	Spam	non-Spam
Number of Tweets	172,191 (30.3%)	395,484 (69.7%)
Number of Users	37,126(17.7%)	173,189(82.3%)

networks provide the IDs of the interested in object (e.g. tweets, accounts) to retrieve them from the servers of the target social network. However, inspired by the nature of spam problem, providing the IDs of spam tweets or accounts is not enough because Twitter might already have suspended their accounts.

Crawling Method. Hence, we exploit our research team crawler to collect accounts and tweets, launched since 1/Jan/2015. The streaming method is used to get an access for 1% of global tweets, as an unbiased crawling way. Such a method is commonly exploited in the literature to collect and create data-set in social networks researches.

Data-set Description. Using our team Twitter data-set, we clustered the collected tweets based on the hashtag available in the tweet with ignoring the tweets that do not contain hashtag. Then, we selected the tweets of 50 trending hashtags (e.g. Trump) randomly sampled to conduct our experiments. Table 1 shows the main statistics of the selected hashtags tweets. We exploited this way in crawling and sampling to remove any possible biasing in the data, and to draw unbiased conclusions.

Ground Truth Data-set. To evaluate the effectiveness of the spammy string patterns in retrieving spam accounts, we created an annotated data-set through labeling each tweet as spam or non-spam. However, with the huge amount of tweets, using manual annotation approach to have labeled data is an impractical solution. Hence, we leveraged a widely followed annotation process in the social spam detection researches. The process checks whether the user of each tweet was suspended by Twitter. In case of suspension, the user is considered as a spammer as well as the corresponding tweet is labeled as a spam; otherwise we assign non-spam and legitimate user for tweet and user, respectively. In total, as reported in Table 2, we found that more than 172,000 tweets were classified as spam, posted by almost 37,000 spammers (spam accounts).

5 RESULTS AND EVALUATIONS

5.1 Experimental Setup

Metrics. Various metrics are used in information retrieval area to evaluate the degree of relevance between documents and an input search query. In our work, we adopt three widely used metrics (Manning et al., 2008; Wang et al., 2013): (i) Precision at L ($P@L$); (ii) Recall at L ($R@L$); (iii) and Normalized Discounted Cumulative Gain at L ($NDCG@L$). In our context, $P@L$ corresponds to the ratio of *spam* accounts retrieved in the top L retrieved accounts. $R@L$ represents the ratio of *spam* accounts retrieved in the top L with respect to the total number of spam accounts in the collection. As precision metric fails to take into account the positions of the relevant spam accounts among the top L, $NDCG@L$ metric penalizes the late appearance of spam accounts in the search list logarithmically proportional to the position in the list retrieved. For the values, we measure the three performance metrics at two ranks $L \in \{100, 500\}$.

Baselines. To assess the performance of our method in retrieving spam accounts, we propose five simple retrieval baseline methods, which rank accounts of a hashtag according to a particular criteria, summarized in:

- **Random (RN)**. This method ranks accounts in a random way without biasing toward any account in the set.
- **Recent Age Account (RAA)**. This method sorts accounts in an ascending order according to the accounts' creation date (age).
- **Old Age Account (OAA)**. Conversely to **RAA** method, **OAA** sorts accounts in a descending order according to the accounts' creation date (age).
- **Recent Posted Tweet (RPT)**. This method ranks accounts according to the publication date of the considered hashtag tweets. Thus, it sorts first the tweets in an ascending order according to the tweets' date and then selects the corresponding accounts in the same order.
- **Old Posted Tweet (OPT)**. This method is similar in concept to the **RPT** method; however, it sorts the tweets first in a descending order and then picks the corresponding accounts in the same order.

Parameter Setting. For community detection stage, we set $\eta = 0.001$, $M = 10,000$, and $\varepsilon = 0.0001$ as values for the learning rate, number of iterations, and the threshold of absolute change in the hidden matrix

Table 3: Performance results of five baseline methods evaluated using **P***recision*@L, **R***ecall*@L, and **NDCG**@L at $L \in \{100, 500\}$.

Baseline Retrieval Method	P@100	P@500	R@100	R@500	NDCG@100	NDCG@500
Random (RN)	7.9%	7.7%	0.5%	1.9%	7.7%	7.7%
Recent Age Account (RAA)	19.9%	20.9%	1.3%	5.8%	19.5%	20.7%
Old Age Account (OAA)	0.6%	1.1%	0.1%	0.3%	0.6%	1.0%
Recent Posted Tweet (RPT)	6.6%	6.9%	0.4%	1.7%	6.3%	6.8%
Old Posted Tweet (OPT)	7.8%	7.0%	0.5%	1.9%	8.0%	7.1%

Table 4: Performance results of the screen name and user name spammy patterns in retrieving spam accounts, experimented at various classification threshold (**Th**) and at different values of **K** $\in \{2, 5, 10\}$ as a number of communities, in terms of **P***recision*@L, **R***ecall*@L, and **NDCG**@L at $L \in \{100, 500\}$.

		Screen-name pattern						**User-name pattern**					
K	Th	P@100	P@500	R@100	R@500	NDCG@100	NDCG@500	P@100	P@500	R@100	R@500	NDCG@100	NDCG@500
2	0.1	**55.4%**	**33.4%**	**7.8%**	**19.8%**	**58.9%**	**46.4%**	48.9%	**43.1%**	7.1%	**23.8%**	47.1%	**43.8%**
	0.2	54.2%	26.3%	7.7%	15.0%	55.9%	37.0%	**49.4%**	35.3%	7.2%	18.7%	48.4%	37.3%
	0.3	48.7%	19.2%	6.7%	10.9%	55.1%	30.6%	49.4%	33.1%	**7.4%**	17.5%	**48.6%**	35.4%
	0.4	45.1%	18.3%	6.0%	9.8%	49.8%	23.5%	46.1%	31.6%	6.9%	16.4%	46.2%	33.9%
	0.5	44.8%	17.2%	5.9%	9.0%	46.9%	22.5%	44.9%	30.1%	6.8%	15.3%	45.0%	32.4%
	0.6	44.4%	15.4%	5.9%	8.9%	46.4%	21.5%	46.7%	29.5%	7.0%	15.1%	46.6%	31.9%
	0.7	42.0%	13.7%	5.5%	8.0%	46.5%	20.0%	43.0%	28.4%	6.5%	14.3%	43.3%	30.6%
	0.8	38.8%	12.5%	5.3%	7.5%	44.6%	18.3%	41.6%	27.5%	6.3%	13.7%	42.1%	29.6%
	0.9	36.0%	10.3%	5.3%	7.0%	41.6%	16.9%	36.9%	25.9%	5.1%	11.8%	37.7%	27.7%
	1.0	36.5%	9.8%	5.4%	6.7%	39.3%	14.6%	37.8%	25.9%	5.2%	11.8%	38.6%	27.9%
5	0.1	**59.4%**	**32.9%**	**8.2%**	**18.1%**	**59.8%**	**36.9%**	48.8%	**41.6%**	7.1%	**22.3%**	47.4%	**42.6%**
	0.2	55.9%	26.1%	7.6%	14.9%	56.9%	30.7%	**50.7%**	36.9%	**7.5%**	19.8%	**49.5%**	38.9%
	0.3	50.9%	19.7%	7.2%	11.6%	52.1%	24.3%	49.8%	34.0%	7.3%	17.8%	48.6%	36.1%
	0.4	39.2%	14.9%	5.3%	8.1%	41.9%	19.0%	44.2%	29.4%	6.3%	14.4%	44.6%	31.7%
	0.5	39.1%	13.6%	5.2%	7.1%	42.2%	18.0%	41.7%	28.1%	5.9%	13.4%	41.8%	30.1%
	0.6	40.4%	12.3%	5.4%	6.9%	44.0%	17.1%	41.2%	26.7%	6.2%	13.0%	42.5%	29.2%
	0.7	42.6%	13.2%	5.4%	7.0%	45.8%	18.1%	42.3%	28.5%	6.2%	14.1%	42.3%	30.5%
	0.8	41.2%	12.2%	5.4%	7.5%	44.1%	16.9%	38.4%	26.6%	5.6%	12.6%	38.8%	28.5%
	0.9	37.9%	10.8%	5.2%	7.0%	40.7%	15.2%	37.7%	26.2%	5.1%	11.8%	38.5%	28.1%
	1.0	37.0%	10.5%	5.0%	7.2%	40.4%	15.0%	38.9%	25.6%	5.5%	11.6%	39.7%	27.7%
10	0.1	**60.2%**	**30.7%**	**8.3%**	**15.7%**	**60.8%**	**35.0%**	**50.8%**	**43.3%**	**7.5%**	**23.0%**	**44.3%**	**28.9%**
	0.2	54.0%	23.3%	7.5%	12.3%	56.0%	28.1%	49.3%	35.5%	7.3%	18.6%	37.7%	24.5%
	0.3	47.0%	18.5%	6.3%	9.5%	48.9%	22.9%	48.4%	32.7%	7.1%	16.8%	35.0%	22.9%
	0.4	43.4%	16.4%	5.7%	7.5%	45.7%	20.7%	44.1%	28.9%	6.3%	13.7%	31.1%	20.5%
	0.5	38.9%	13.6%	5.0%	6.7%	41.9%	17.9%	39.4%	27.3%	5.7%	12.1%	29.4%	19.7%
	0.6	34.5%	11.8%	4.4%	5.5%	37.9%	15.9%	38.0%	25.3%	5.7%	11.6%	27.5%	18.7%
	0.7	35.4%	10.0%	4.8%	5.6%	39.5%	14.6%	38.3%	24.6%	5.6%	11.2%	26.7%	18.3%
	0.8	37.2%	10.8%	5.0%	6.1%	41.3%	15.5%	37.4%	24.8%	5.4%	11.4%	27.1%	18.4%
	0.9	35.9%	10.7%	4.4%	6.9%	40.1%	15.2%	36.9%	25.7%	4.9%	11.5%	27.9%	18.8%
	1.0	35.6%	10.0%	4.5%	7.1%	40.0%	14.6%	38.1%	26.1%	5.1%	11.9%	28.4%	19.0%

H, respectively. For the number of communities K, we experiment our method at three different values, $K \in \{2, 5, 10\}$, to study its effect. For the size of information matrices $\mathbf{X}^{\bullet}$, we consider all distinct users (accounts) of each hashtag without excluding any user available in the testing collection. As an iterative algorithm is used for solving the optimization problem, we initialize each entry of the hidden matrix **H** by a small positive real value drawn from an uniform distribution on the interval [0,1]. For the threshold (Th), we study the effect of this threshold in predicting effective spammy string patterns using different values, $Th \in [0.1, 1]$.

Experiment Procedure. For each hashtag, we perform the following steps: (i) we extract the users who posted the tweets related to the considered hashtag; (ii) we apply then the community detection method on the extracted users set; (iii) afterward, each community is labeled as spam and non-spam based on the designed objective function for a certain threshold (Th); (iv) for each community labeled as spam, the highest likelihood string pattern is extracted from both screen name and user name potential sets, respectively; (v) for each spammy string pattern chosen in the previous step, we retrieve the accounts that contain the selected pattern based on the considered attribute (user name or screen name), using all accounts in the data collection; (vi) the accounts retrieved are ranked according to the degree of matching or overlapping between the exploited pattern and the string of the considered attribute, using the definition of equation 2; (vii) the performance of the spammy string pattern in retrieving spam accounts is evaluated using the three described metrics; (viii) the previous step is repeated for the pattern chosen from a spam community; (ix) in last step, the performance of all spammy string patterns of the selected attribute (screen name or user name) is averaged for the final computation.

5.2 Experimental Results

To the best of our knowledge, the work introduced in this paper is the first one in spam retrieval direction, where all prior works in the literature have focused on spam detection only. Hence, we compare our approach with the five baselines proposed as a reference point for the comparison step.

In the first experiment, we evaluate the performance of the five baseline retrieval methods (**RN**, **RAA**, **OAA**, **RPT**, and **OPT**) by applying each method on each hashtag in our data-set. Averaged on all hashtags, we observe in Table 3 that the recent age account retrieval (**RAA**) method has superior performance in terms of precision, recall, and NDCG, compared to the rest four baseline methods. For **RAA** method, the 19.9% value of P@100 can be interpreted in two ways: (i) the number of spam accounts retrieved (ranked based on accounts' age) in the top 100 accounts is 20; (ii) or the probability to retrieve a spam account in the top 100 is about 19.9%. The recall@100 values of all baseline methods are not at the same level as precision@100 values. Indeed, the low values of recall are because of the huge number of spam accounts existing in our collection. This interpretation can be ensured by the significant increasing in the recall@500 values of all baselines where more spam accounts can be found at rank 500. However, the recall values of all baseline methods are not satisfactory and adoptable for retrieving all spam accounts in the collection. More precisely, the best recall@500 is obtained by **RAA** method which has retrieved no more than 2,200 spam accounts available in our data-set. The behavior of baseline methods in regards of **NDCG** metric is almost similar to the precision metric. The strength of the **NDCG** metric is in giving weight for each account retrieved based on its position in the list (i.e. top position $\Longrightarrow$ highest weight). The 19.5% of **NDCG**@100 of **RAA** baseline method gives an indication that the spam accounts retrieved are *not* in the top 80 accounts. Overall, the performance results obtained by the five baseline methods draw the following conclusions: (i) the results of **RAA** method ensure the validity of a strong hypothesis that spam accounts are recent in age; (ii) the recall values of the five methods are *not* satisfactory to adopt them in retrieving spam accounts at all.

In producing the experimental results of our method, the procedure described in 5.1 subsection is performed on each hashtag in the data-set used, with averaging the results over the number of hashtags. We perform two main experiments to provide more insights into the impact of screen name and user name attributes in retrieving spam accounts. In Table 4, we report the evaluation results of both attributes, experimented at different classification thresholds Th and at finite set of number of communities K. As an external comparison with the baseline methods, our approach has superior performance in retrieving spam accounts either when considering screen name or user name attributes. As an internal comparison, we find that 0.1 is almost the optimal classification threshold in terms of the performance metrics. We don't report the results of the threshold when be less than 0.1 since no spam community have been identified. The optimal threshold value, 0.1, ensures our hypothesis that the distribution of patterns in a spam community follows non-uniform distribution. Also, we find that increasing the classification threshold implies to classify communities as spam where they are truly not spam ones. Consequently, this misclassification at community level produces spammy patterns which are not spammy at all. Indeed, this explains the degradation in the performance results when increasing the classification threshold. The effect of number of communities **K** is obvious in improving precision@100, recall@100, and NDCG@100 metrics. We associate this increasing with the number of spam bots that had attacked the considered 50 hashtags. More precisely, when a hashtag is attacked by 10 different spam bots, the low values of K cannot provide all possible spammy patterns, and thus not all spam accounts might be retrieved. On contrary, we observe that the increasing in the number of communities decreases the precision@500, recall@500, and NDCG@500 metrics. We explain this behavior because of the existence non-spam accounts having partial matching with the spammy patterns of either screen name or user name attributes.

Obviously, according to the results, none of the both attributes has superior performance compared to each other. For instance, the spammy patterns of screen name attribute perform better than the spammy patterns of user name attribute in terms of precision@100, recall@100, and NDCG@100. The selection among screen name and user name attributes in performing the retrieval process is completely dependent on the size of the list that contains the potential spam accounts (e.g 100, 500). For instance, user name attribute is preferred when performing the retrieval task to return 500 potential spam accounts.

6 CONCLUSION AND FUTURE DIRECTIONS

Conventional spam detection methods check individual tweets or accounts for the existence of spam. In this paper, we study the spam problem from informa-

tion retrieval point of view to provide searchable information acting as a query to retrieve spam accounts. Our work uses the simple meta-data of a particular set of users posted tweets associated with a topic (hashtag) to predict spammy naming patterns as a searchable information. Our work can be leveraged by Twitter community to search for spam accounts and also for Twitter based applications that work on large collection of tweets. As our work is the first in this direction, we intend to extend the method to predict searchable information also from tweets as well, with working on improving the retrieval metrics of the current method.

REFERENCES

Abascal-Mena, R., Lema, R., and Sèdes, F. (2014). From tweet to graph: Social network analysis for semantic information extraction. In *IEEE 8th International Conference on Research Challenges in Information Science, RCIS 2014, Marrakech, Morocco, May 28-30, 2014*, pages 1–10.

Abascal-Mena, R., Lema, R., and Sèdes, F. (2015). Detecting sociosemantic communities by applying social network analysis in tweets. *Social Network Analysis and Mining*, 5(1):1–17.

Abascal-Mena, R., Lema, R., and Sèdes, F. (2015). Detecting sociosemantic communities by applying social network analysis in tweets. *Social Netw. Analys. Mining*, 5(1):38:1–38:17.

Amleshwaram, A. A., Reddy, N., Yadav, S., Gu, G., and Yang, C. (2013). Cats: Characterizing automation of twitter spammers. In *Communication Systems and Networks (COMSNETS), 2013 Fifth International Conference on*, pages 1–10. IEEE.

Benevenuto, F., Magno, G., Rodrigues, T., and Almeida, V. (2010). Detecting spammers on twitter. In *In Collaboration, Electronic messaging, Anti-Abuse and Spam Conference (CEAS*, page 12.

Cao, C. and Caverlee, J. (2015). Detecting spam urls in social media via behavioral analysis. In *Advances in Information Retrieval*, pages 703–714. Springer.

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

Chu, Z., Widjaja, I., and Wang, H. (2012b). Detecting social spam campaigns on twitter. In *Applied Cryptography and Network Security*, pages 455–472. Springer.

Freeman, D. M. (2013). Using naive bayes to detect spammy names in social networks. In *Proceedings of the 2013 ACM Workshop on Artificial Intelligence and Security*, AISec '13, pages 3–12, New York, NY, USA. ACM.

Hu, X., Tang, J., and Liu, H. (2014). Online social spammer detection. In *AAAI*, pages 59–65.

Hu, X., Tang, J., Zhang, Y., and Liu, H. (2013). Social spammer detection in microblogging. In *IJCAI*, volume 13, pages 2633–2639. Citeseer.

Lee, K., Caverlee, J., and 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*, SIGIR '10, pages 435–442, New York, NY, USA. ACM.

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

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

McCord, M. and Chuah, M. (2011). 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.

Meda, C., Bisio, F., Gastaldo, P., and Zunino, R. (2014). A machine learning approach for twitter spammers detection. In *2014 International Carnahan Conference on Security Technology (ICCST)*, pages 1–6. IEEE.

Mezghani, M., On-at, S., Péninou, A., Canut, M., Zayani, C. A., Amous, I., and Sèdes, F. (2015). A case study on the influence of the user profile enrichment on buzz propagation in social media: Experiments on delicious. In *New Trends in Databases and Information Systems - ADBIS 2015 Short Papers and Workshops, BigDap, DCSA, GID, MEBIS, OAIS, SW4CH, WISARD, Poitiers, France, September 8-11, 2015. Proceedings*, pages 567–577.

Mezghani, M., Zayani, C. A., Amous, I., Péninou, A., and 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*, pages 1–11.

On-at, S., Quirin, A., Péninou, A., Baptiste-Jessel, N., Canut, M., and Sèdes, F. (2016). Taking into account the evolution of users social profile: Experiments on twitter and some learned lessons. In *Tenth IEEE International Conference on Research Challenges in Information Science, RCIS 2016, Grenoble, France, June 1-3, 2016*, pages 1–12.

Santos, I., Miambres-Marcos, I., Laorden, C., Galn-Garca, P., Santamara-Ibirika, A., and Bringas, P. G. (2014). Twitter content-based spam filtering. In *International Joint Conference SOCO'13-CISIS'13-ICEUTE'13*, pages 449–458. Springer.

Stringhini, G., Kruegel, C., and Vigna, G. (2010). Detecting spammers on social networks. In *Proceedings of the 26th Annual Computer Security Applications Conference*, ACSAC '10, pages 1–9, New York, NY, USA. ACM.

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

Wang, A. H. (2010). Don't follow me: Spam detection in twitter. In *Security and Cryptography (SECRYPT),*

Proceedings of the 2010 International Conference on, pages 1–10.

Wang, Y., Wang, L., Li, Y., He, D., Chen, W., and Liu, T.-Y. (2013). A theoretical analysis of ndcg ranking measures. In *Proceedings of the 26th Annual Conference on Learning Theory (COLT 2013)*.

Yang, C., Harkreader, R., Zhang, J., Shin, S., and 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*, WWW '12, pages 71–80, New York, NY, USA. ACM.

Yang, C., Harkreader, R. C., and Gu, G. (2011). 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. Springer-Verlag.

Yang, J. and Leskovec, J. (2013). Overlapping community detection at scale: A nonnegative matrix factorization approach. In *Proceedings of the Sixth ACM International Conference on Web Search and Data Mining*, WSDM '13, pages 587–596, New York, NY, USA. ACM.