Iberian Journal of Applied Sciences and Innovations

2021 vol 1, Issue 2

Constructing New Features for Spam Detection on Twitter

Arash Erami
Department of Computer Engineering, Shiraz Branch, Islamic Azad University, Shiraz, Iran
arsherami@gmail.com

*Elham Parvinnia**
Department of Computer Engineering, Shiraz Branch, Islamic Azad University, Shiraz, Iran
parvinnia@iaushiraz.ac.ir

Abstract—Nowadays, by the growth of the internet, social networks are attracting unprecedented attention to themselves. Most people are at least active in one social network. Users in social networks follow their favorite people and topics to discover the latest news about them. This rising number of users has made social networks fertile grounds for advertising and finding the bait. Social networks also become celebrities' popularity criterion. The problem is that some accounts created to spread malicious links, steal user's information, and display advertising. These accounts are mainly controlled and supervised by an automatic program. Not only the increase in fake accounts has costs for social networks companies, but it also influences network quality. In this paper, we offer some new and low-cost features to distinguish spam accounts on Twitter. This paper offers some low cost and a new feature to distinguish spam accounts of Twitter. We apply machine learning algorithms to predestined datasets, and by looking at the characteristics of the accounts, then we anticipate class of users by the accuracy of 99.18%.

KeyWords— Spam Detection • Machine Learning • Twitter Spam Bots • Feature Extraction.

I. INTRODUCTION

Online Social Networks (OSNs) have spread at a remarkable speed over the past decade. They have become one of the main ways for people to keep track of events and communicate with one another. Websites such as Facebook, Twitter, and LinkedIn are consistently on the top 20 most-visited websites. Twitter is the fastest growing social networking web site among all the social networking websites [1]. The increase in the use of social networking websites is gaining a great deal of recognition because they play a double role of online social networking and micro-blogging, but these websites have constraints, i.e., the spammers.

Twitter is a popular online social networking and microblogging tool, which allows users to share content limited to 140 characters. These small messages (tweets) create substantial information dissemination in the network and make Twitter a successful social network for content share. There are about 500 million tweets published every day [expanded ramblings, 2015].

Spam is becoming a significant problem with Twitter as well as with other online social networking websites. Spammers can use Twitter as a tool to send unsolicited messages to legitimate users, post malicious links, and hijack trending topics. Spammers could be phishers, malware propagators, marketers, and adult content propagators. Fake followers are Twitter accounts specifically created to inflate the number of followers of a target account, in order to increase its popularity and influence.

With more than 500 million users on Twitter, it is almost impossible to manually verify the identity of every user who signs up on Twitter, and it is even more challenging to keep track of users who tend to spread information of questionable authenticity, unknowingly or deliberately. Therefore, we need some tools to identify these spammers automatically.

More than 19% of all tweets are about organizations or product brands, less than 20% of which are shown to have significant sentiment [13].

Since spam bots amend their behaviors to remain undetected, we need some new features to detect them. In this work, we combine features and make new rules to distinguish a spam account from a legitimate one.

a) Spam Bots and Sybil Accounts

There are several ways to take advantage of free online advertisement, and many agencies and companies rely on Spam Bots or Sybil accounts. These fake accounts pretend to be legitimate distinct users, and their behavior seems to be similar [12]. Some of these accounts might seem surprisingly akin to being legitimate. They cause the social network platform millions of dollars in revenue loss each year.

In a social network such as Twitter, users can access all public information, including usernames, tweets, etc. Spammers need to parse the public content to get all the information they need for both sending the malicious content (usernames) and for making it appealing to the victim (relationships, interests, and content of previous messages). By using this information, it is possible to semantic to analyze victim accounts.

b) Roadmap

The remainder of this paper is structured as follows. In Section 2, we consider related work in Twitter spams and bot detection. In Section 3, we describe the outlines of our baseline dataset. In Section 4, we extract and examine some new features of our baseline dataset. In Section 5, we present our results and compare them to previous works. In Section 6, we present possible methods for future work.

II. RELATED WORK

[8] used a machine learning approach to distinguish spam bots from normal ones. He suggested three graph-based features and three content-based features. They used graph-based features such as a number of friends, a number of followers, and a follower ratio. He also extracted the number of duplicate tweets, the number of HTTP links and the number of replies/mentions from the user's 20 most recent tweets. His best overall performance was .917 by Naïve Bayesian algorithm.
Some research focused on analyzing the behaviors of social spammers and detecting these spammers [5] [4]; [11]. LEE, K, and others conducted a long-term study of content polluters, analyzed their behaviors, and detected them. [6] used a machine learning approach to detect social spammers.
In other work [10], researchers suggested some new features and used the profile-based feature, content-based feature, graph-based features to distinguish spammers.
[7] analyze how spammers operate in social network sites operate. They created a large and diverse set of "honey-profiles" on three large social network sites and logged the messages and friends request they received. They analyzed the collected data and identified the anomalous behaviors of users who contacted predestine profiles. Based on the analysis of their behaviors, they showed that it is possible to automatically identify the accounts used by spammers and correctly detected 15,857 spam profiles.

III. DATASET

In this section, we describe the datasets of Twitter accounts that we used to conduct our study throughout the paper. We use "The Fake Project Dataset" provided by MIB Datasets, which is publicly available to the scientific community. This dataset contains five different sources of Twitter user's data in two classes: Human and Fake. Gathering data from multiple sources make the data more reliable because each source contains different kinds of behaviors and information. More information on the dataset can be found in [2] as we use the same dataset in order to compare the result.
Each source contains four separate files named: followers, friends, tweets, users. Here are the details of each file:
Followers: The list of user IDs and the IDs of their followers
Friends: The list of user IDs and the IDs of their friends
Tweets: Contains 19 attributes of a tweet like a tweet text, time of creation, number of hashtags, user ID, tweet ID, the source of the tweet, etc.
Users: Contains user 's information from such as name, number of friends, location, description, language, profile text color, class, etc.
In each file, we have a user`s ID so that we can join files together by a unique User ID.
Since we only had 1950 genuine users and many machine learning algorithms are affected by the imbalance [9] of natural distributions of the minority and majority classes, we selected all human users and randomly selected 1950 accounts out of 3351 bots accounts, to balance out both categories. Table 1 provides the details of this dataset.

Table 1: Brief Description of Our Dataset Sources

Dataset	**Accounts**
TFP(@TheFakeProject)	**469**
E13 (#elezzioni2013)	**1481**
INT(intertwitter)	**1337**
TWT(twittertechnology)	**845**
FSF(fasstfolliwerz)	**1169**
HUM(total human dataset)	**1950**
FAKE(total fake dataset)	**3351**
BAS(baseline dataset Hum Union with random Fake dataset)	**3900**

a) Preprocessing

In this section, we reduced features in order to make our dataset lighter. First, we removed useless features and features with lots of missing values. So, how can we tell if a feature is useful? Attributes that have lots of different values for each user such as name, description, URL, and ID are not practical for machine learning algorithms. Since in machine learning algorithms such as decision tree, unique attributes, and attributes that have massive variety in value have a GINI index close to zero, which means it cannot be a good separator for detecting different classes.

IV. PROPOSED FEATURES

We have extracted features from two sources of information: the feature of tweets and the user's information. Obviously, good features should be informative and have discriminative power. Some features such as followers count and friends count have some correlation with each other so we decided to make a ratio.

a) Extracting New Features

We previously mentioned that programmers of these bots try to find ways to evade from spam detection algorithms, therefore it is necessary to continuously need new features and algorithms to correctly distinguish bots.
Based on [2] we categorized attributes into three categories by their crawling costs.

A) Profile: Features that use information in a profile account.

B) Timeline: Features that use information in tweets.

C) Relationship: Features that uses information about the accounts that are in a relationship with followers of the target account

As [2] mentioned features of profile have the least crawling time and relationship features have the most time needed. Our suggested features are only in profile and timeline category.

b) Profile Attributes

Attributes extracted from timeline need more time to collect than attributes which can be found in profile information. In user information, we had "created_time" so we can calculate new attribute named "Howlong_day" that shows how long does this account exist and no need for curling timeline. Another attribute that we amended to be more accurate is the number of tweets. Since the number of tweets in users' info had slightly different from tweet files, we calculate the exact number called that feature "num_of_tweets."
Dou to following reasons, the number of tweets foregoing in users' files and the number of tweets existed in tweets file are not the same. One of the possible reasons for this is that user data is cached by Twitter and thus, it is not always updated on the current number of followers, friends, statuses (tweets), etc. Another reason might be that we were not able to collect all the tweets produced by a user because the user deleted some of his/her original tweets. Alternatively, because he/she "protected" his/her timeline. The third possible reason is that the crawling of user data and tweets have been carried out at two slightly different times, and this may have caused inconsistencies.
We made new ratios by combining profile attributes. These ratios demonstrate the popularity of the account. We did this because using only one attributes could deceive the machine learning algorithm. For example, number of followers without considering the number of friends could lead us to wrong predictions.
Twitter saves each user's signup date and the time of each tweet. We figured out how long each user has been active on Twitter by an attribute called "Howlong_day." Spambots probably have a shorter lifespan than real users, because spam bots may be deactivated by Twitter spam detection system or deactivated by programmer after some time. We assume that the more significant this number is, the higher possibility of being a legitimate user.
Follower counts have long provided a decent indicator of Twitter accounts' popularity. Twitter provides a feature to make a list of accounts that a user follows. By this feature, users can make any list they want (brands, news, entertainment, ...), so users can categorize each account to a list. We think this feature can be another indicator of popularity. The more list you are on, the more popular you probably are. So we create an attribute called "listed_count" that shows the number of public lists a specific user is a member of. Obviously, if the number of followers is relatively small compared to the number of people you are following, the follower ratio is relatively small and close to zero. At the same time, the probability that the associated account is spam is high.
We know a number of followers and friends are two major attributes, and using ratio can be helpful. To consider other attributes, we add "listed_count" attribute to this ratio and create "Ratio1" feature. Because "listed_count" most of the time is much smaller than the number of followers and friends, we rose it to a power of two to infect ratio. In the ratio2 feature, we emphasize on a number of friends by rose friends count to the power of two. To see how tweets can attract followers, we created "populaty_by_tweet" ratio. Growth rate considers how fast an account absorbs followers.

$$\text{Ratio1} = \frac{\text{followers\_count} + \text{num\_of\_tweets}}{\text{friends\_count} + \text{listed\_count}^2} \tag{1}$$

$$\text{Ratio2} = \frac{\text{friends\_count}^2 + \text{num\_of\_tweets}}{\text{followers\_count}} \tag{2}$$

$$\text{Popularity\_by\_tweets} = \frac{\text{friends\_count} + \text{number\_of\_tweets}}{\text{followers\_count}} \tag{3}$$

$$\text{Growth\_rate} = \frac{\text{followers\_count}}{\text{howlong\_day}} \tag{4}$$

c) Timeline Attributes

Besides profile attributes, we select six timeline attributes that exist in every single tweet. Since these attributes are for every tweet, not every user, we calculate the average, variance, and maximum of these attributes for all accounts in our dataset.

Favorite count: Shows how many users have set the tweet as a favorite Retweet count: Demonstrates the importance of the tweet. Retweets build on the authority of another user and are used to increase the volume of followers to see a tweet.

Replay count: Represents the reaction of users to a certain tweet Num hashtags: Number of hashtags in every single tweet.

Num URLs: Number of URL in every single tweet Num mentions: Number of mentions in every single tweet These attributes are important for us for the following reasons. Since spam tweets are seldom retweeted, set to favorite, or replied to, we selected these features. Spam bots are very likely to share URLs to reach their goal like phishing, advertising, or spreading malware. Spammers also use trending hashtags and mention other users to be indexed in search results and more people be able to see their tweets. For these reasons, we choose the attributes “num_hashtags,” “num_URLs” and “num_mentions.” We also calculated tweet lengths for each tweet and then computed the average tweet length and created a new attribute called “average character.”

Mention: To address a particular user in order to reference the user directly. Mentions may be used by spammers to personalize the message in an attempt to increase the likelihood a victim follow spam links. Mentions can be used to communicate with users that do not follow a spammer.

When a tweet is sent, Twitter keeps the source of the tweet. This is embodied in the device type and application or API, which is used to publish the tweet. We put all sources into the three following categories:

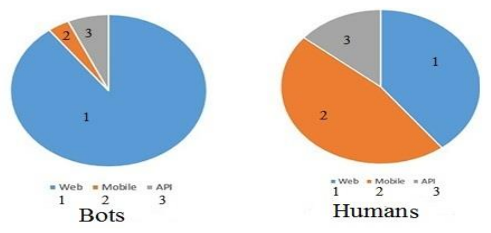
Web: Uses Twitter web site for sending tweets.

Mobile or Tablet: Tweets sent from portable devices like phones and tablets.

3rd Party and API: Tweets sent from applications and API requests.

We categorized tweets for each user, then we took the most repeated sources and made a new attribute called “Mode Source.” Figure 1 shows the dispersion of legitimate users against spam bots.

Figure 1 Source dispersion of legitimate users versus spam bots



So far we have created 21 features from our base dataset, which is listed in table 2. We should test the attributes to see if they are suitable for detecting spammer. Our new features are tested in the following section.

Table 2: Extracted Features

#	Name	Description
1.	Average_character	Average number of tweet character
2.	Average_favorite_count	Average number of tweets favorite
3.	Average_num_hashtags	Average number of hashtags in tweets
4.	Average_num_mentions	Average number of mention in tweets
5.	Average_num_urls	Average number of URL in tweets
6.	Average_reply_count	Average number of replay for tweets
7.	Average_retweet_count	Average number of retweet of their tweets
8.	Variance_favorite_count	Variance of favorite count
9.	Variance_num_hashtags	Variance of number of hashtags in tweets
10.	Variance_num_mentions	Variance of number of mentions in tweets
11.	Variance_num_urls	Variance of number of URL in tweets
12.	Variance_reply_count	Variance of number of replay to tweets
13.	Variance_retweet_count	Variance of number of retweeted tweets
14.	Mode_of_source	Source that users mostly used for tweeting
15.	Maximum_ favorite_count	Maximum number of

		favorite tweet
16.	Maximum_num_hashtags	Maximum number of hashtags in one tweet
17.	Maximum_num_urls	Maximum number of URLs in one tweet
18.	Maximum_reply_count	Maximum number of replay to one tweet
19.	Maximum_num_mentions	Maximum number of mentions in one tweet
20.	Maximum_retweet_count	Maximum number of retweeted tweet
21.	Howlong_day	Number of days that users register up to now

d) Evaluation Methodology

We have two classes in our dataset: humans and bots (spammers).

• True Positive (TP): the number of those bots recognized as bots;

• True Negative (TN): the number of those humans followers recognized as human;

• False Positive (FP): the number of those humans recognized as bots;

• False Negative (FN): the number of those bots recognized as human.

In order to evaluate the application of every single rule to the accounts in the baseline dataset, we have to consider the following standard evaluation metrics:

• Accuracy: the proportion of predicted true results (both true positives and true negatives) in the population, that is $\frac{TP+TN}{TP+TN+FP+FN}$

• Precision: the ratio of predicted positive cases that are indeed real positive, which is $\frac{TP}{TP+FP}$

• Recall: the ratio of real positive cases that are indeed predicted positive, which is $\frac{TP}{TP+FN}$

• F-Measure: the harmonic mean of precision and recall, namely $\frac{2*\text{precision}*\text{recall}}{\text{precision}+\text{recall}}$

• The area under the curve (AUC): that relates the hit rate to the false alarm rate has become a standard measure in testing the accuracy of predictive modeling.

e) Testing Our Features

To obtain the optimal classifier, this is crucial to combine the features effectively. We test our new features using k-fold Cross-validation decision tree algorithms. In k-fold cross-validation, the original sample is randomly partitioned into k equal size subsamples. Of the k subsamples, a single subsample is retained as the validation data for testing the model, and the remaining k-1 subsamples are used as training data. The cross-validation process is then repeated k times (the folds), with each of the k subsamples used exactly once as the validation data. The k results from the folds can then be averaged (or otherwise combined) to produce a single estimation. The advantage of this method is that all observations are used for both training and validation, and each observation is used for validation exactly once.

We assayed each attribute alone in 10-fold validation using a decision tree algorithm. The result is shown in Table 3.

Table 3: Testing new attributes by 10 k fold validation decision tree

#	Attribute	Accuracy	Precision	Recall	F-Measure
1.	Average_character	80.03	79.94	79.92	79.91
2.	Average_favorite_count	89.85	89.19	90.66	89.90
3.	Average_num_hashtags	82.69	82.32	83.23	82.70
4.	Average_num_mentions	91.33	95.79	86.40	90.81
5.	Average_num_urls	85.59	97.20	73.18	83.48
6.	Average_reply_count	86.00	96.71	74.54	84.15
7.	Average_retweet_count	79.95	78.21	86.38	80.99
8.	Variance_favorite_count	89.44	86.93	92.81	89.74
9.	Variance_num_hashtags	84.67	89.74	78.25	83.58
10.	Variance_num_mentions	90.26	96.06	83.86	89.51
11.	Variance_num_urls	85.64	97.33	73.19	83.52

12.	Variance_reply_count	88.28	92.54	83.49	87.66
13.	Variance_retweet_count	80.95	94.43	65.67	77.37
14.	Mode_of_source	75.23	69.44	90.14	78.40
15.	Maximum_favorite_count	89.05	88.65	89.52	89.05
16.	Maximum_num_hashtags	90.15	91.37	88.55	89.93
17.	Maximum_num_urls	84.97	97.91	71.40	82.56
18.	Maximum_reply_count	69.85	81.95	50.93	62.79
19.	Maximum_num_mentions	91.38	95.80	86.44	90.86
20.	Maximum_retweet_count	88.56	96.82	79.69	87.39
21.	Howlong_day	86.46	96.98	75.33	84.74
22.	Ratio1	96.77	97.56	95.93	96.72
23.	Ratio2	94.31	92.1	96.86	94.42
24.	Popularity_by_tweets	83.90	78.98	92.42	85.11
25.	Groth_rate	85.79	80.31	94.80	86.94

The result of our features average compares to previous work which is mentions in [2] average is shown in table 4. The result shows an observable increase in accuracy, precision, recall, and F-measure. The accuracy of 86.04% for average singe feature confirm these features are good.

Table 4: Average of Single feature comparison

Average of Single Feature	Accuracy	Precision	Recall	F-Measure
Camisani-Calzolari	64.4	71.3	58.18	54.02
Van Den Beld	41.36	41.36	68.12	18.65
Social Bakers	50.2	66.26	4.9	69.2
Our features	86.04	89.77	82.54	85.28

We test our 4 ratios together by 10-fold validation and different machine learning algorithms. We did this because our 4 ratios are extracted from profile information and need less time to acquire.

Table 5: Result of all rules together for 10 fold validation

#	Algorithm	Accuracy	Precision	Recall	F-Measure
1.	Decision Tree	98.31	98.92	97.69	98.30
2.	Random Forest	97.87	99.11	96.59	97.83
3.	AdaBoost	98.38	98.97	97.82	98.39
4.	Naïve Bayes	95.18	97.98	92.34	95.05

Although we only use profile information in our ratio, the results are remarkable. We achieve the accuracy of 98.38% by using profile information and AdaBoost algorithm. We want to go farther and add our extracting timeline features to this ratio to make it even better. In the next section, we describe it more.

f) Finding Best Feature Set

After testing attributes one by one now it is time to find the best feature set for getting the best result. We used “Forward Selection” algorithms in RapidMiner software to find the best-combined features.

The Forward Selection operator starts with an empty selection of attributes and, in each round, it adds each unused attribute of the given Example Set. For each added attribute, the performance is estimated using the inner operators, in this case, cross-validation. Only the attribute giving the highest increase in performance is added to the selection. Then a new round is started with the modified selection. This implementation avoids any additional memory consumption besides the memory used originally for storing the data and the memory which might be needed for applying the inner operators. The stopping behavior parameter specifies when the iteration should be aborted. There are three different options:

Without increase: The iteration runs as long as there is an increase in performance.

Without an increase of at least: The iteration runs as long as the increase is at least as high as specified, either relative or absolute. The minimal relative increase parameter is used for specifying the minimal relative increase if the use relative increase parameter is set to true. Otherwise, the minimal absolute increase parameter is used for specifying the minimal absolute increase.

Without significant increase: The iteration stops as soon as the increase is not significant to the level specified by the alpha parameter.

We gave all our extracted features to the forward selection algorithm and choose stopping behavior to be iteration without a significant increase. The output of this operator will be a list of all attributes with weights 0 or 1, which 1 means good to select and 0 means not have a significant effect on the validation result and not selected. Table 6, shows a list of the most effective attributes for the detection model.

Table 6: Selected Attributes Using a Forward Selection Algorithm.

#	Attributes	Weights
1	Average_reply_count	1
2	Ratio1	1
3	Ratio2	1
4	Popularity_by_tweets	1
5	Variance_favorite_count	1

V. RESULT

In Table 7, we can see the result of our feature set on the base dataset.

Table 7: Result of our feature set by 10 fold validation

#	Algorithm	Accuracy	precision	Recall	F-Measure	AUC
1.	Decision Tree	99.18	99.32	99.02	99.17	0.993
2.	Random Forest	98.87	99.11	98.59	98.85	0.998
3.	AdaBoost	98.54	98.52	98.56	98.53	0.796
4.	Naïve Bayes	72.66	99.06	45.23	62.08	0.989
5.	K Nearest Neighbors	92.05	92.23	91.85	92.02	0.951

In comparison to [10], we got the same result with fewer features. We only used profile information and timeline features and we do not imply relationship features to our feature set. It means that the time of crawling for features is much less than [10] research.

Table 8 is comparing the result of two previous research [10], [7] that have been mentioned on [2] by the same dataset that we used in our work. Our detection system has higher accuracy than [7].

Table 8: Compare the result to previous works

Algorithms	Suggested by	Accuracy	Precision	Recall	F-Measure	AUC
Random Forest	Stringhini	.981	.983	.979	.981	.995
	Yang	.991	.991	.991	.991	.998
	Our	.9887	.9911	.9859	.9885	.998
Decision Tree	Stringhini	.979	.984	.974	.979	.985
	Yang	.990	.991	.989	.990	.997
	Our	.9918	.9932	.9902	.9917	.993
Adaptive Boost	Stringhini	.968	.965	.970	.968	.995
	Yang	.988	.989	.937	.988	.999
	Our	.9854	.9852	.9856	.6208	.989
K-NN	Stringhini	.954	.961	.946	.953	.974
	Yang	.966	.966	.966	.966	.983
	Our	.9205	.9223	.9185	.9202	.951

In [2], they used features from the profile category proposed by others which contains 23 features after that they used all features without considering the timing category which contains 49 features. Table 9 reports the result of k fold validation on our suggested five features and compare it to 23 features of profile category and 49 features from all categories presented in [2].

Table 9: Result of Our 5 Features Compare to 23 Features and 49 Features

Algorithms	Features	Accuracy	Precision	Recall	F-Measure	AUC
Random Forest	49 features	.994	.997	.990	.994	.999
	23 Features	.987	.993	.980	.987	.995
	Our5 Features	.9887	.9911	.9859	.9885	.998
Decision Tree	49 features	.992	.991	.992	.992	.993
	23 Features	.983	.987	.979	.983	.983
	Our5 Features	.9918	.9932	.9902	.9917	.993

Adaptive Boost	49 features	.987	.988	.987	.987	.999
	23 Feature	.972	.975	.969	.972	.995
	Our5 Features	.9854	.9852	.9856	.6208	.989
K-NN	49 features	.971	.963	.979	.971	.990
	23 Features	.957	.961	.953	.957	.978
	Our5 Features	.9205	.9223	.9185	.9202	.951

We realize from table 9 that our suggested features have more accuracy than the 23 profile features suggested before. Although our features come with slightly less accuracy by comparing them to 49 features, we should consider the time for gathering relationship features is much more from features from profile and timeline.

VI. FUTURE WORKS

In the future, we can aim for new features based on text mining and analyzing tweets, and if tweets meaning are related to the hashtag, they are making or not. Also, some new behavioral features, such as the speed of replying to tweets and making trending hashtags can be considered.

REFERENCES

[1]'Compete Site Comparison' <http://siteanalytics.compete.com/facebook.com+myspace.com+twitter.com> accessed 11 June 2016.

[2] Cresci, S., Di Pietro, R., Petrocchi, M., Spognardi, A., Tesconi, M. (2015). Fame for sale: Efficient detection of fake Twitter followers. Decision Support Systems, 80, 56-71. http://dx.doi.org/10.1016/j.dss.2015.09.003

[3] '388 Amazing Twitter Statistics And Facts' (expandedramblings, 2015) <http://siteanalytics.compete.com/facebook.com+myspace.com+twitter.com> accessed 10 May 2016.

[4] Ghosh, S., Korlam, G., Ganguly, N. (2011). Spammers' networks within online social networks. Proceedings Of The 20Th International Conference Companion On World Wide Web - WWW '11. http://dx.doi.org/10.1145/1963192.1963214

[5] Lee, K., Caverlee, J., Webb, S. (2010). Uncovering social spammers. Proceeding Of The 33Rd International ACM SIGIR Conference On Research And Development In Information Retrieval - SIGIR '10. http://dx.doi.org/10.1145/1835449.1835522

[6] McCord, M., Chuah, M. (2011). Spam Detection on Twitter Using Traditional Classifiers. Lecture Notes In Computer Science, 175-186. http://dx.doi.org/10.1007/978-3-642-23496-5_13

[7] Stringhini, G., Kruegel, C., Vigna, G. (2010). Detecting spammers on social networks. Proceedings Of The 26Th Annual Computer Security Applications Conference On - ACSAC '10. http://dx.doi.org/10.1145/1920261.1920263

[8] Wang, A. (2010). Detecting Spam Bots in Online Social Networking Sites: A Machine Learning Approach. Lecture Notes In Computer Science, 335-342. http://dx.doi.org/10.1007/978-3-642-13739-6_25

[9] Weiss, G., Provost, F. (2003). Learning When Training Data are Costly: The Effect of Class Distribution on Tree Induction.

[10] Yang, C., Harkreader, R., Gu, G. (2013). Empirical Evaluation and New Design for Fighting Evolving Twitter Spammers. IEEE Transactions On Information Forensics And Security, 8(8), 1280-1293. http://dx.doi.org/10.1109/tifs.2013.2267732

[11] Yardi, S., Romero, D., Schoenebeck, G., Boyd, D. (2009). Detecting spam in a Twitter network. First Monday, 15(1). http://dx.doi.org/10.5210/fm.v15i1.2793

[12] Yu, H., Kaminsky, M., Gibbons, P., Flaxman, A. (2006). SybilGuard. ACM SIGCOMM Computer Communication Review, 36(4), 267. http://dx.doi.org/10.1145/1151659.1159945

[13] Yu, S., Kak, S. (2012). A Survey of Prediction Using Social Media.long.