Information Quality in Social Networks: A Collaborative Method for Detecting Spam Tweets in Trending Topics

Mahdi Washha, Aziz Qaroush, Manel Mezghani, Florence Sèdes

Information Quality in Social Networks: A Collaborative Method for Detecting Spam Tweets in Trending Topics

Mahdi Washha[1(✉)], Aziz Qaroush[2], Manel Mezghani[1], and Florence Sedes[1]

[1] IRIT - Paul Sabatier University, Toulouse, France
{Mahdi.washha,Florence.sedes}@irit.fr, Mezghani.manel@gmail.com
[2] Department of Electrical and Computer Engineering,
Birzeit University, Ramallah, Palestine
Aqaroush@birzeit.edu

Abstract. In Twitter based applications such as tweet summarization, the existence of ill-intentioned users so-called spammers imposes challenges to maintain high performance level in those applications. Conventional social spammer/spam detection methods require significant and unavoidable processing time, extending to months for treating large collections of tweets. Moreover, these methods are completely dependent on supervised learning approach to produce classification models, raising the need for ground truth data-set. In this paper, we design an unsupervised language model based method that performs collaboration with other social networks to detect spam tweets in large-scale topics (e.g. hashtags). We experiment our method on filtering more than 6 million tweets posted in 100 trending topics where Facebook social network is accounted in the collaboration. Experiments demonstrate highly competitive efficiency in regards to processing time and classification performance, compared to conventional spam tweet detection methods.

Keywords: Social spam · Social networks · Collaboration · Topics

1 Introduction

With the enormous popularity of online social networks (OSNs) over the Internet, ill-intentioned users so-called spammers have exploited OSNs for spreading spam content (e.g. advertisements, porn materials, and phishing websites) [1]. Indeed, performing spamming tasks by spammers may cause major problems in different directions, such as: (i) polluting search results by spam information; (ii) degrading statistics accuracy obtained by mining tools; (iii) consuming storage resources; (iv) and violating user's privacy. However, with these serious problems, OSNs' anti-spam mechanisms have failed to end-up the spam problem, raising real concerns about the quality of "crawled" data collections. Hence, besides the importance of OSNs data for tremendous range of areas such as search engines and research field, filtering out noisy data to have high quality information is

DOI: 10.1007/978-3-319-60045-1_24

the obvious and straight forward solution. Information quality process in social networks is generically summarized in three dependent steps [2]: (i) selecting the data collections (e.g. Facebook accounts, Tweets, Facebook posts) that need improvements; (ii) determining the noise type (e.g. spam, rumor) to be filtered out; (ii) at last, applying pre-designed algorithms depending on the chosen noise type to produce noise free data collections.

In the battle of fighting spam on Twitter, a considerable set of methods [1,3–8] has been designed for detecting spam accounts and spam campaigns with little attention dedicated toward spam tweets detection. The account-level and campaign-level detection methods are time consuming, requiring months to process large collections consisting of millions of Twitter users. The main source of high time consumption is the use of constrained REST APIs[1] to retrieve a required information (e.g. followers, followees, and user time-line) to perform such detection methods. On the other side, the existing tweet-level spam detection methods are grounded on exploiting the features extraction concept combined with supervised machine learning algorithms to build a predictive model using an annotated data-set. The main strength point of tweet-level is the fast detection in regards to time consumption since the detection process is performed on the available information in tweet object only. However, given the fact that spammers are too dynamic in the spam contents, tweet-level detection methods have drawbacks and limitations, including the followings aspects: (i) the use of non-discriminative and ineffective features such as number of words in tweet; (ii) the need for an annotated data-set to build a classification model; (iii) and the use of supervised learning algorithms produces biased models toward the training set adopted.

In this paper, we introduce a design of an unsupervised method for filtering out spam tweets existing in large-scale collections of trending topics. Our method performs collaboration with other OSNs through searching and gathering information relevant to trending topics. Then, a content matching is performed between a desired tweet and retrieved information, like Facebook relevant posts, to decide later the class label of the considered tweet. In this work, we hypothesis that the volume and the content of spam on OSNs vary depending upon the privacy rules followed by OSNs. For instance, Facebook[2] social network adopts restricted rules more than Twitter in opening new accounts such as mobile verification, which impose difficulties to create huge spam campaigns.

The remainder of the paper is organized as follows. Section 2 gives an overview about Twitter-based spam detection methods. Section 3 presents the notations, problem formalization, and design of our collaborative method used in detecting spam tweets. Section 4 describes the data-set used in validating our approach. Section 5 presents the experimental results. Section 6 concludes the paper with providing future perspectives.

[1] https://dev.twitter.com/rest/public.
[2] https://www.facebook.com/policies.

2 Related Work

Most of the existing works for fighting spam on Twitter have focused on account and campaign (bot) detection levels with little efforts spent for spam tweet-level detection.

Tweet-Level. At this level, individual tweets are checked for the existence of spam content. Benevenuto [1] extracted a set of simple statistical features from the tweet object such as number of words, number of hashtags, and number of characters. Then, a binary classifier is built on a small annotated data-set. Martinez-Romo and Araujo [9] detected spam tweets in trending topics through employing language models to extract more features such as the probability distribution divergence between a given tweet and other tweets of a trending topic. The major problem at this level of detection is derived from the lack of sufficient information that can be extracted from tweet object itself. In addition, building language models using tweets of trending topics definitely fails when having huge spam attacks. Our work overcomes these shortcomings through exploiting topic relevant information from other OSNs.

Account-Level. Methods designed in [1,3,5,10,11] work firstly through building features vector by extracting hand-designed features such as number of followers, and node betweenness. Then, supervised machine learning algorithms are applied to build a classification model on an annotated data-set. Despite of high detection rate when exploiting such features, extracting them requires significant time to collect information from Twitter's servers through using REST APIs. Indeed, these APIs are constrained to a certain and predefined number of calls, making the extraction of most features not possible in regards to time point of view, especially when treating large-scale data-set.

Campaign-Level. Chu et al. [8] proposed a spam campaign detection method through clustering accounts according to available URLs in tweets. Then, they represented each cluster by a vector of features similar to account-level detection methods. In [12], a classification model was designed to capture differences among bot, human, and cyborg. Regrettably, this level of detection has similar account-level methods drawbacks, making such solutions not scalable for large collections of users or tweets.

3 Collaborative Model Design

Our approach focuses on finding a matched information on other OSNs for a given tweet related to a certain topic. As the obvious purpose of using topic modeling in OSNs is to group similar information, the probability of finding same information talking about same topic on different OSNs is relativity high. Conversely, the probability of finding same spam content posted under the same topic is relativity low because of its dependency on spammers' goals and the openness of OSNs themselves. Therefore, instead of extracting uninformative features (e.g. number of words in tweet) to learn model using machine learning

algorithms, we rely on using statistical language model concept to detect spam tweets.

3.1 Notations and Definitions

Let $C_H = \{T_1, T_2, ...\}$ be a collection of tweets for a particular trending topic H, where $T_\bullet$ element represents the tweet object modeled as 2-tuple $T_\bullet = <Text, Actions>$. Also, we model the information retrieved about the topic, H, from defined social networks (e.g. Facebook, Instagram), $SN_\bullet$, as a finite set $S_H = \{SN_{Facbook}, SN_{Instagram}, ...\}$. Each $SN_\bullet$ is modeled as a finite set of posts $SN_\bullet = \{O_1, O_2, ...\}$ where the element $O_\bullet$ is defined by 2-tuple $O_\bullet = <Text, Actions>$. Each element inside the post O and tweet T tuple is defined as follows:

Text. As each post may consist of text, we represent the content of post as a finite set of textual words, $Text = \{w_1, w_2, ...\}$.

Actions. Users of social networks may perform actions on posts as a reaction toward the content of posts or tweets. We define actions as a finite set of 2-tuple, $Actions = \{<a_{name_1}, a_{val_1}>, <a_{name_2}, a_{val_2}>, ...\}$, where a_{name} represents the name of action (e.g. like, share, and comment on Facebook) depending upon the considered social network, and $a_{val} \in \mathbb{N}_{\geq 0}$ is the number of times that the corresponding action performed by social network users on the considered post or tweet.

3.2 Problem Formalization

Given a collection of tweets C_H associated with a trending topic, H, and posted by a set of distinct users U_H such that $U_H \leq |C_H|$, our main problem is to filter out spam tweets in the given collection C_H without involving information requiring REST API calls. More formally, we aim at designing a function f such that it predicts the class label of each tweet in the desired collection, defined as $f(T) : T \rightarrow \{spam, non\text{-}spam\}, T \in C_H$.

3.3 Tweet Likelihood, Post Prior, and Tweet Classification

Tweet Likelihood. We leverage statistical language models [13] to estimate the relevance degree of other OSNs' posts with a given tweet to make a decision later about the tweet. Language modeling method computes the probability $P(D|Q)$ of a document D being generated by a query Q to rank a set of documents. We transform the same concept to get out the most relevant post in other social networks for a given tweet. Thus, we treat tweets as queries and posts as documents, with computing the post O probability of being generated by a tweet T as:

$$P^{SN_i}(O|T) \stackrel{\text{rank}}{=} P^{SN_i}(O).P^{SN_i}(T|O) = P^{SN_i}(O). \prod_{w \in T.Text} P^{SN_i}(w|O) \quad (1)$$

$P^{SN_i}(O)$ is the post prior probability such that $O \in SN_i$. The post prior can be viewed as tweet-independent features (i.e. features *not* extracted from tweet object) representing the probability of being non-spam content in the social network SN_i. Estimating the other probability component $P^{SN_i}(T|O)$ can be performed using different models (Jelineck Mercer, Dirichlet) [13] to compute $P^{SN_i}(w|O)$ or (Kullback-Leibler divergence) [14] to calculate the degree of dissimilarity between the tweet and post language models. In this paper, we use the uni-gram language model for representing tweets and posts because of its outstanding performance in information retrieval field. Also, we adopt Kullback-Leibler divergence (KL) method because of its fast computation time compared to others. However, the classical version of KL method cannot be exploited directly in computing the $P^{SN_i}(T|O)$ probability since the zero value of KL means that the language models of tweet and post are completely similar. Moreover, the range of KL method is unbounded, meaning that the ∞ value appears when two language models are dissimilar. Hence, we customize the current version of KL method to inverse the semantic of KL values (i.e. $0 \implies$ dissimilar and $1 \implies$ similar) with bounding its values, where the probability component $P^{SN_i}(T|O)$ is defined as:

$$P^{SN_i}(T|O) = \frac{\log|T.Text| - \sum_{w \in T.Text} P(w|M_T) * min(|\log \frac{P(w|M_T)}{P(w|M_O)}|, \log|T.Text|)}{\log|T.Text|} \quad (2)$$

where $P(w|M_T)$ and $P(w|M_O)$ are the probability of word w being generated by tweet and post language models (M_T, M_O), respectively.

Post Prior. As the retrieved posts of social network, SN_i, may consist of spam content, we estimate the probability of being non-spam through leveraging the actions performed by users on the retrieved posts set (i.e. more actions $\implies$ low probability for being spam post). We assume that actions (e.g. like, comment, and share) are independent features, and thus the general formula for calculating post prior is computed as:

$$P^{SN_i}(O) = \prod_{A \in O.Actions} P(A) \quad (3)$$

where $P(A)$ is estimated using the maximum-likelihood of performing the action A on the post O, computed as $P(A) = \frac{Count(A,O)}{Count(A,SN_i)}$. $Count(A,O) = A.val$ means that the number of times that the action A performed on the post O. $Count(A, SN_i)$ represents the summation of action A over available posts in SN_i.

Tweet Classification. When doing inference for a given tweet over a set of posts in SN_i, we obtain a vector of probability values where each represents the degree of matching between a post and a given tweet. We exploit these values to make a decision about the class label of a given tweet. To do so, we define a thresholded decision function that labels tweets as non-spam in case of finding at least one post on any social network having probability above a fixed threshold.

Table 1. Statistics of Twitter and Facebook crawled data-sets.

Twitter		Facebook	
Property	Value	Property	Value
# of accounts	2,088,131 (4.9% spammers)	# of users	3,122
# of tweets	6,470,809 (11.8% spam)	# of posts	6,880
# of replied tweets	76,393	# of comments	2,398,611
# of re-tweeted tweets	3,129,237	# of reactions	64,083,457

Formally, we define the crisp decision function as follows:

$$F(T, S_H) = \begin{cases} non\text{-}spam & max\{\frac{P^{SN_i}(O|T)}{Sum(SN_i,T)} | SN_i \in S_H, O \in SN_i\} \geq \Delta \\ spam & \text{otherwise} \end{cases} \quad (4)$$

where the function $Sum(SN_i, T) = \sum_{O \in SN_i} P^{SN_i}(O|T)$ normalizes the probability of each post retrieved from a certain social network SN_i, making their summation equals to one. Δ is a threshold interpreted as the minimum probability (i.e. matching degree) required to classify the considered tweet T as non-spam.

4 Data-Set Description and Ground Truth

As various social networks available over web, in this paper, we experiment our method through performing collaboration with Facebook social network only. Hence, in this section, we describe the Twitter and Facebook data-sets that have been exploited in validating our method.

Twitter Data-Set. The data-sets used at tweet level detection [1,9] are not publicly available for research use. Also, Twitter's polices allow to publish only the IDs of accounts and tweets of Twitter data-sets. Indeed, in context of social spam problem, using ID is not a solution since Twitter might already have deleted the corresponding object (account or tweet) and thus no information is available to retrieve. Hence, we developed a crawler to collect tweets using real-time streaming method provided by Twitter. Then, we launched our crawler for five months, from 1/Jan/2016 to 31/May/2016, with storing the topics that were trending in the specified period. Afterward, we clustered the crawled tweets based on the available topics in the text of tweets, with discarding the tweets that don't have a trending topic. As thousands of topics available in our tweets collection, we selected the tweets of 100 trending topics randomly sampled to validate our approach. To build an annotated data-set consisting of spam and non-spam tweets, we leverage a widely followed annotation process in the social spam detection researches, named as "Twitter Suspended Spammers (TSS)" [9]. 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. We performed this process in 1/Nov/2016 to gain large set of spam tweets, annotated around 763,555 as spam tweets and about 102,318 as spammers (spam accounts), as reported in Table 1.

Facebook Data-Set. For the selected 100 trending topics, we crawled the corresponding Facebook posts that contain those topics and posted during the period 1/Jan/2016 to 31/May/2016. It is important to mention that Facebook community has stopped recently post searching APIs in the latest version, v2.8, of Graph API[3] released on August 2016. Thus, we overcome this obstacle through developing a Facebook crawler that searches for a particular topic using a normal Facebook account and then parses the HTML tags of the retrieved posts. We automate this process through using open source Selenium web browser automation tool[4]. In total, as reported in Table 1, we crawled more than 6,880 Facebook posts generated by about 3,122 different users in less than one hour.

5 Results and Evaluations

5.1 Experimental Setup

Performance Metrics. As the ground truth class label about each tweet is available, we exploit accuracy, precision, recall, F-measure, average precision, average recall, and average F-measure, computed according to the confusion matrix of Weka tool [15], 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 "spam" class, while the average metrics combines both classes based on the fraction of each class (e.g. 11.8% * "spam precision" + 88.2% * "non-spam precision").

Baselines. We define two baselines to compare our method with: (i) baseline "A" which represents the results when classifying all tweets as non-spam directly without doing any kind of classification; (ii) baseline "B" which reflects the results obtained when applying supervised machine learning algorithms on state of the art "tweet" features described in Table 2. As many learning algorithms provided by Weka tool, we exploit Naive Bayes, Random Forest, J48, and support vector machine (SVM) as well-known supervised learning methods to evaluate the performance of the mentioned state of the art features.

Parameter Setting. In computing the post prior probability, $P^{SN_i}(O)$, we adopt "Likes", "Shares", "Comments", "Wow", "Love", "Sad","Haha", and "Angry" as actions. In our method, Δ is the main variable in classifying tweets and thus we study the impact of changing its value through performing experiments at different values of $\Delta \in [0.1, 1.0]$ with 0.1 increment step. For the Naive Bayes method, we set the "useKernelEstimator" and "useSupervisedDiscretization" options to false value as default values set by Weka. For Random Forest, we set the option max depth to 0 (unlimited), with studying the effect of changing

[3] https://developers.facebook.com/docs/graph-api/using-graph-api.
[4] http://docs.seleniumhq.org/.

number of trees $\in \{100, 500\}$. For $J48$ method, we set the minimum number of instances per leaf to 2, number of folds to 3, and confidence factor to 0.2. For the SVM method, we use the LibSVM [17] implementation integrated with Weka tool with setting the kernel function to Radial Basis and examining the impact of gamma $\in \{0.5, 1\}$, where the rest parameters are set to the default ones.

Experiment Procedure. For the baseline "B" experiments, we use the concept of cross validation along the 100 trending topics in our data-set, summarized in the following steps: (i) for each topic, we build a feature vector space using the state of the art features described in Table 2; (ii) then, a feature vector space of a selected topic (training set) only is used to build a predictive model using a chosen learning algorithm; (iii) the feature vector spaces of rest topics (i.e. 99 topics for testing) are validated on the built classification model in the previous step; (iv) the validation results in terms of true positive, true negative, false positive, false negative are extracted and stored; (v) the steps from ii to iv are repeated on each topic in the collection; (vi) at last, using the validation results obtained for each single topic, we calculate the performance metrics mentioned above. It is important to mention that the experiment procedure for the baseline "B" simulates exactly the real scenarios in detecting spam tweets.

In experimenting our method, for each topic we perform the following steps: (i) for a certain value of classification threshold Δ, the designed classification model in Sect. 3 is applied on the considered topic tweets using the correspond-

Table 2. Description of the state of the art "tweet" features used in building supervised classification models [1,9,16].

Feature name	Description
Number of hashtags	Counts the number of hashtags available in the tweet text
Number of spam words	Counts the number of words that listed as spam words in the tweet text
Hashtags ratio	Ratio of number of hashtags with respect to the number of words in the tweet
URLs ratio	Ratio of number of URLs posted in the tweet with respect to the number of tweet words
Number of words	Counts the number of words in the tweet
Number of numeric characters	Counts the number of numeric characters in the tweet text
Number of URLs	Counts the number of URLs posted in the tweet
Number of mentions	Counts the number of accounts (users) mentions in the tweet
Replied tweet	Checks whether the tweet is a replied tweet or not
Tweet and URL content similarity	Measures the similarity between the tweet text and the text of URL posted in Tweet

ing topic Facebook posts to predict the class label of tweets; (ii) then, the results in terms of true positive, true negative, false positive, false negative are extracted and stored for final results computations; (iii) the previous two steps are performed on each topic in the data-set; (iv) in the last step, the results of whole topics are summed together to compute the performance results using the mentioned metrics.

5.2 Experimental Results

According to the results of the baselines reported in Table 3, the supervised classification models have strong failure in filtering out the spam tweets existing in the 100 trending topics. This failure can be easily captured from the low spam recall values (4^{th} column) where the highest value is obtained by $NaiveBayes$ learning algorithm. The 10.5% of spam recall obtained by $NaiveBayes$ means that less than 80,000 of spam tweets can be detected from around 736,500 spam tweets. The low spam precision values also give an indication that a significant number of "non-spam" tweets has been classified into "spam" ones. Subsequently, as spam F-measure is dependent on recall and precision metrics, the values of spam F-measure are definitely low. The accuracy values of baseline "B" are close to the accuracy value of baseline "A". However, given the low values of spam precision and spam recall, the accuracy metric in this case is not an indicative and useful metric to judge on the supervised learning as winner approach. More precisely, the supervised learning approach does not add significant contribution in increasing the quality of the 100 trending topics tweets. The key idea of using different machine learning algorithms with playing in their parameters is

Table 3. Performance results of baseline A and baseline B in terms of different metrics.

Learning algorithm	Accuracy	Precision	Recall	F-measure	Avg. precision	Avg. recall	Avg. F-measure
Baseline (A): All tweets labeled as non-spam							
————	88.2%	0.0%	0.0%	0.0.%	88.2%	88.2%	88.2%
Baseline (B): Supervised machine learning approach							
Naive Bayes	81.2%	13.7%	10.5%	11.9%	79.0%	81.2%	80.1%
Random Forest (#Trees = 100)	86.4%	13.2%	2.8%	4.6%	79.0%	86.4%	80.1%
Random Forest (#Trees = 500)	86.5%	12.6%	2.6%	4.7%	79.4%	86.5%	82.8%
J48 (Confidence Factor = 0.2)	86.4%	13.8%	2.9%	4.9%	79.6%	86.4%	82.5%
SVM (Gamma = 0.5)	87.2%	15.7%	0.2%	0.4%	78.3%	87.2%	82.5%
SVM (Gamma = 1.0)	87.0%	15.9%	0.1%	0.3%	77.9%	87.0%	82.2%

Table 4. Our collaborative method performance results in terms of different metrics, showing the impact of post prior probability component when performing collaboration with Facebook social network.

Model (Δ)	Accuracy	Precision	Recall	F-measure	Avg. precision	Avg. recall	Avg. F-measure
Uniform post prior probability							
$\Delta = 0.1$	49.8%	10.8%	48.3%	17.7%	79.7%	49.8%	61.3%
$\Delta = 0.2$	32.3%	10.8%	69.4%	18.7%	79.1%	32.3%	45.9%
$\Delta = 0.3$	26.2%	10.8%	77.0%	18.9%	78.6%	26.2%	39.3%
$\Delta = 0.4$	22.8%	10.9%	82.3%	19.2%	78.5%	22.8%	35.3%
$\Delta = 0.5$	21.0%	11.0%	85.3%	19.4%	78.7%	21.0%	33.2%
$\Delta = 0.6$	19.4%	11.0%	87.9%	19.6%	78.8%	19.4%	31.2%
$\Delta = 0.7$	18.7%	11.1%	89.3%	19.7%	79.1%	18.7%	30.3%
$\Delta = 0.8$	17.5%	11.1%	90.9%	19.8%	79.3%	17.5%	28.7%
$\Delta = 0.9$	17.2%	11.1%	91.5%	19.8%	79.2%	17.2%	28.3%
$\Delta = 1.0$	17.2%	11.1%	91.6%	19.8%	79.4%	17.2%	28.3%
Non-Uniform post prior probability							
$\Delta = 0.1$	80.7%	17.0%	18.8%	17.8%	81.4%	80.7%	81.0%
$\Delta = 0.2$	80.6%	17.2%	19.3%	18.2%	81.5%	80.6%	81.0%
$\Delta = 0.3$	79.3%	15.8%	19.6%	17.5%	81.2%	79.3%	80.2%
$\Delta = 0.4$	77.8%	15.0%	21.1%	17.5%	81.1%	77.8%	79.4%
$\Delta = 0.5$	73.4%	13.5%	24.9%	17.4%	80.8%	73.4%	77.1%
$\Delta = 0.6$	64.0%	12.3%	36.4%	18.5%	80.7%	64.0%	71.4%
$\Delta = 0.7$	57.7%	11.9%	43.4%	18.7%	80.6%	57.7%	67.2%
$\Delta = 0.8$	51.9%	11.5%	49.0%	18.6%	80.3%	51.9%	63.0%
$\Delta = 0.9$	42.2%	11.0%	59.0%	18.6%	79.8%	42.2%	55.2%
$\Delta = 1.0$	34.79%	10.7%	66.0%	18.5%	79.1%	34.79%	48.3%

to highlight the badness of the state of the art tweet features. Overall, the results obtained by the learning models draw various conclusions: (i) the state of the art features are not discriminative among non-spam and spam tweets, ensuring the dynamicity of spam content; (ii) spammers tend to publish tweets almost similar to non-spam ones; (iii) adopting a supervised approach to perform training on an annotated data-set of trending topics and applying the classification model on future or not annotated trending topics is *not* the solution at all.

Taking a look at our method performance results in Table 4, the behavior is completely different in recalling (classifying) "spam" tweets, especially when the value of Δ gets higher. The recall results are completely consistent with the Eq. 4 designed for classifying tweets. For high values of Δ, the major difficulty is in finding at one high matched Facebook post to classify the considered tweet as

"non-spam". Thus, this explains the dramatic degradation in the accuracy when increasing the value of Δ. Although of high recall values, the spam precision values of our method are almost similar to the supervised learning approach ones.

Uniform vs. Non-uniform Post Prior. The role of post prior probability component is obvious in detecting spam tweets. Working on the assumption that each Facebook post has same probability (uniform) for being non-spam increases the spam recall values when the value of Δ gets higher, leading to detect most spam tweets. On contrary, a significant number of "non-spam" tweets has been predicted as "spam" ones. We interpret this behavior because of the small value of post prior probability when working on the uniform probability assumption. Indeed, this problem is reduced when considering the actions performed on Facebook post to compute the post prior probability component. Thus, the spam recall has increased without high degradation in the accuracy values. Although of low values of spam precision, the high values of average precision mean that little tweets have been classified as "non-spam" where they are truly "spam".

High Quality vs. False Positive. In email spam filtering, the efforts are directed for 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" tweet as "spam" is not a serious 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. tweet summarization) has high priority to work on noise free collections. Also, the computational time aspect is significant when targeting large-scale collections. Hence, our method 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 Facebook and applying our model. At last, as various experiments are given for different Δ values where no optimal value can satisfy all performance metrics, the selection is mainly dependent on the desired requirements of the final collection. For instance, high Δ value is recommended to have too high quality collection with having high probability to lose not noisy information.

6 Conclusion and Future Directions

In this paper, we study the impact of performing collaboration with social networks to filter out spam tweets in large-scale collections of trending topics. We propose an unsupervised method grounding on the language model concept to find out similar information in other social networks considered in the collaboration. Our method outperforms conventional detection spam methods in regards to time consumption, requiring few hours to process around 6 millions tweets posted in 100 trending topics. With this novel idea in battle of fighting spam, we plan as a future work to study the effect of performing collaboration with

additional social networks such as Instagram. Also, we intend to improve the classification performance through extracting more features from users' comments such as sentiment features. Moreover, we plan to study the behavior of using different language models with their estimations.

References

1. Benevenuto, F., Magno, G., Rodrigues, T., Almeida, V.: Detecting spammers on Twitter. In: Collaboration, Electronic messaging, Anti-abuse and Spam Conference (CEAS), p. 12 (2010)
2. Agarwal, N., Yiliyasi, Y.: Information quality challenges in social media. In: International Conference on Information Quality (ICIQ) (2010)
3. Wang, A.H.: Don't follow me: spam detection in Twitter. In: Proceedings of the 2010 International Conference on Security and Cryptography (SECRYPT), pp. 1–10, July 2010
4. Yardi, S., Romero, D., Schoenebeck, G., danah boyd: Detecting spam in a Twitter network. First Monday **15**(1) (2009)
5. Stringhini, G., Kruegel, C., Vigna, G.: Detecting spammers on social networks. In: Proceedings of the 26th Annual Computer Security Applications Conference (ACSAC 2010), pp. 1–9. ACM, New York (2010)
6. Yang, C., Harkreader, R.C., Gu, G.: Die free or live hard? Empirical evaluation and new design for fighting evolving Twitter spammers. In: Sommer, R., Balzarotti, D., Maier, G. (eds.) RAID 2011. LNCS, vol. 6961, pp. 318–337. Springer, Heidelberg (2011). doi:10.1007/978-3-642-23644-0_17
7. Amleshwaram, A.A., Reddy, N., Yadav, S., Guofei, G., Yang, C.: Cats: characterizing automation of Twitter spammers. In: 2013 Fifth International Conference on Communication Systems and Networks (COMSNETS), pp. 1–10. IEEE (2013)
8. Chu, Z., Widjaja, I., Wang, H.: Detecting social spam campaigns on Twitter. In: Bao, F., Samarati, P., Zhou, J. (eds.) ACNS 2012. LNCS, vol. 7341, pp. 455–472. Springer, Heidelberg (2012). doi:10.1007/978-3-642-31284-7_27
9. Martinez-Romo, J., Araujo, L.: Detecting malicious tweets in trending topics using a statistical analysis of language. Expert Syst. Appl. **40**(8), 2992–3000 (2013)
10. McCord, M., Chuah, M.: Spam detection on Twitter using traditional classifiers. In: Calero, J.M.A., Yang, L.T., Mármol, F.G., García Villalba, L.J., Li, A.X., Wang, Y. (eds.) ATC 2011. LNCS, vol. 6906, pp. 175–186. Springer, Heidelberg (2011). doi:10.1007/978-3-642-23496-5_13
11. Cao, C., Caverlee, J.: Detecting spam URLs in social media via behavioral analysis. In: Hanbury, A., Kazai, G., Rauber, A., Fuhr, N. (eds.) ECIR 2015. LNCS, vol. 9022, pp. 703–714. Springer, Cham (2015). doi:10.1007/978-3-319-16354-3_77
12. Chu, Z., Gianvecchio, S., Wang, H., Jajodia, S.: Detecting automation of Twitter accounts: are you a human, bot, or cyborg? IEEE Trans. Dependable Secure Comput. **9**(6), 811–824 (2012)
13. Ponte, J.M., Croft, W.B.: A language modeling approach to information retrieval. In: Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 275–281. ACM (1998)
14. Kullback, S.: The Kullback-Leibler distance. Am. Stat. **41**(4), 340–341 (1987)
15. Hall, M., Frank, E., Holmes, G., Pfahringer, B., Reutemann, P., Witten, I.H.: The Weka data mining software: an update. SIGKDD Explor. Newsl. **11**(1), 10–18 (2009)

16. McCord, M., Chuah, M.: Spam detection on Twitter using traditional classifiers. In: Calero, J.M.A., Yang, L.T., Mármol, F.G., García Villalba, L.J., Li, A.X., Wang, Y. (eds.) ATC 2011. LNCS, vol. 6906, pp. 175–186. Springer, Heidelberg (2011). doi:10.1007/978-3-642-23496-5_13
17. Chang, C.-C., Lin, C.-J.: LIBSVM: a library for support vector machines. ACM Trans. Intell. Syst. Technol. **2**, 27:1–27:27 (2011). http://www.csie.ntu.edu.tw/cjlin/libsvm