# A Semi-Automatic Approach for Labeling Large Amounts of Automated and Non-Automated Social Media User Accounts

Christopher Teljstedt
KTH Royal Institute of Technology
Stockholm, Sweden
Email: chte@kth.se

Magnus Rosell
Swedish Defence Research Agency (FOI)
Stockholm, Sweden
Email: magnus.rosell@foi.se

Fredrik Johansson
Swedish Defence Research Agency (FOI)
Stockholm, Sweden
Email: fredrik.johansson@foi.se

***Abstract*—Automated accounts are used for many purposes in social media, including sending spam, spreading of viruses and conducting psychological operations in political or military conflicts. While several previous attempts have been made to classify bot accounts in the spam domain, there are (to the best of our knowledge) no previous studies on detection of automated accounts in a military information operation context. Traditional machine learning approaches to bot detection rely on manual annotation of training sets from which classifiers can be learnt, which requires a large manual effort. We present a semi-automated alternative to manual annotation which significantly reduces the effort and resources needed, and hence speeds up the process of adapting classifiers to new domains. Our application of the method to Twitter data from the Russia-Ukraine conflict and our classification results suggest that good classification performance still can be obtained despite generating training sets semi-automatically rather than using manual annotation.**

## I. Introduction

Automated bot accounts sending spam have long been a problem for various online social networks, including Twitter. Traditionally, spam bots have been used for advertising and mass marketing, but is also often used for cyber crime-related activities such as attempts to spread malware, viruses, and phishing attacks. More recently, automated accounts have also been used for various kinds of information operations in political and military conflicts, including use of Twitter for spreading terrorism-related propaganda and flooding of Twitter hashtags about political protests in e.g. Syria and Russia with completely unrelated tweets.

Although quite a lot of research has been been devoted to detection of spam bots and spam messages related to illicit marketing and to scams and phishing attacks, there are to the best of our knowledge no previous attempts made to large-scale detection of the use of automated accounts for political, ideological, or military information operations. Morover, almost all research on spam classification has been relying on manual annotation of (small-scale) training sets on which machine learning-based classifiers can be learned. A problem with such an approach is that it scales badly and that the annotation process has to be redone each time the classifier should be applied to a new domain or context. In order to overcome this problem we suggest a semi-automated method for labeling automated and non-automated user accounts on social media in general, and Twitter in particular. These labels are then used for training a machine learning-based classifier. A potential advantage of this approach is that it quickly can label large amounts of training data which allows for fast development and adaptation of appropriate classifiers to new contexts and domains.

The rest of this paper is structured as follows. In Section II, we present related work. In Section III we describe our new methodology for semi-automatic labeling of social media accounts as being either automated or non-automated. In Section IV, we present the features for detection of automated accounts that have been utilized in our experiments, and in Section V we describe how it has been used for labeling a Twitter dataset collected from the Russia-Ukraine conflict. Section VI explains how we have trained a random forest classifier on the semi-automatically labeled Ukrainian conflict training data and presents the achieved results. Finally, we present some conclusions and suggest ideas for future work in Section VII.

## II. Previous Work

Most existing literature addressing characterization and identification of automation in microblogs is related to detection of spam or spammers. Spam can be viewed as a subset of the more general problem of finding automation, as automated accounts can be used for many other purposes including various types of information operations.

In [1] a classifier is developed in order to assist identification of different degrees of automation in Twitter. The model is based on (i) an entropy component that measures the tweeting behavior in form of periodic and regular timings between tweets, (ii) a spam component that measures the tweet content by checking text patterns and comparing tweets, and (iii) a user account properties component that uses properties such as account age and number of followers. They use 10-fold cross validation to train and test a random forest classifier on a manually labeled ground-truth set. The results show that the proposed classifier could accurately differentiate human, cyborgs and bots. Another example of using content-based

978-1-4673-7592-4/15 $31.00 © 2015 IEEE
DOI 10.1109/ENIC.2015.31

features such as measuring repetitiveness in tweets and entropy features to separate spam accounts from legitimate accounts is found in [2]. When evaluating their classifiers on two data sets, their best models obtain a 96% detection rate and a 0.8% false positive rate. Similar to the entropy models used in [1] and [2], behavior-based features are used in [3] to detect spammers on Twitter. In their paper, timing patterns extracted from timestamp information associated with each tweet is used to test non-uniform tweeting behavior. A $\chi^2$-test is used to assess whether tweets from an account appears to be drawn uniformly across the seconds-of-the-minute and minutes-of-the-hour distributions, since this is what can be expected from human users who are not using automation software. Distributions deviating strongly from uniformity can therefore be good indicators of automated behavior. Duplicate tweet content and links is also used for detecting automated behavior.

In [4] it is examined how graph-based and content-based features can be combined in order to facilitate spam detection. Two sets of features are suggested: (i) graph-based features such as number of followers and friends and (ii) content-based features such as duplicate content and number of links. Several learning methods are used in their study, including decision trees and naïve Bayes classifiers. The ground-truth used for training and evaluation purposes is a manually labeled training dataset consisting of 2,000 users and a test dataset with 500 users. A naïve Bayes classifier is shown to have the best performance with an accuracy of 93.5% and precision, recall and F-measures above 90% obtained using 10-fold cross validation. In [5], a classification model is built to classify spammers and non-spammers on a manually labeled dataset. The results show that the model succeeded in correctly classifying 70% of the spammers and 96% of the non-spammers. Furthermore, the paper highlights the most important features for spam detection in their study: fraction of tweets with URLs, age of the user account, average number of URLs per tweet, fraction of followers per followees, fraction of tweets the user had replied, number of tweets the user replied to, number of tweets to which the user receives a reply, number of followees, number of followers, and average number of hashtags per tweet.

To summarize, most previous studies on spam and bot detection in online social networks have relied on manually annotated training data [1], [6], [4], [5] A large number of different classifiers and features have been utilized in previous work. In studies by [2] and [3] the annotation problem has been approached in an alternative way by taking advantage of that Twitter users often manually report spam accounts. In this way, the researchers have assumed that all accounts being reported to be spam accounts are indeed spam bots.

## III. Method for semi-automatic labeling

Existing machine learning-based classifiers are in general created from small manually annotated datasets. A problem with such approaches is that manual annotation is a tedious task which demand considerable efforts in order to create large representative samples to train on. Additionally, this task has to be done from scratch each time a classifier has to be relearned, e.g, due to changes in bot behavior or due to a need to adapt it to a new domain or a new social media platform.

In this section we present a novel methodology for semi-automatic labeling of large sets of user generated content, significantly reducing the time and effort required for creation of new datasets from which classifiers can be learnt.

As we saw in the previous section, there are many potential features that can be used for bot detection. The same features are appropriate for detection of any automated account. Let us first consider a simple such feature: the frequency of a user account's tweets measured as the number of tweeets per second: tweets/s. With only this feature at our disposal, a simplistic method for deciding if an account is automated or not would be to use a cut-off value $C$: any account having more than $C$ tweets/s is considered being an automated account. If we choose $C$ to be high enough we will only assign the label "automated account" to accounts that are indeed automated. Obviously a lot of automated accounts will not be found and if we choose a too low $C$ we will misclassify accounts as automated. Hence, we will have to set $C$ considering the application at hand.

Using our simplistic method we can gather a lot of examples of automated accounts. Supposing that these also exhibit other traits of automation besides being frequent tweeters, we can use this data set to train a machine learning-based method for automated account detection. These other traits will potentially be captured via other features and our detector will learn to find automated accounts by looking not only at the original feature. In this case, we would obviously have to be cautious and monitor the detector so it does not generalize too much in any respect. One indication of how it handles new instances would be to evaluate it on data held out from the data set.

There may be several features that on their own separate automated and not automated accounts to some degree. Therefore, we generalize the original idea and consider several features $F_i$, where $i \in \{0, 1, 2, \ldots I\}$. For each of these features we in some manner decide a cut-off value $C_i$, hereby splitting an original set of accounts $I$ times, giving us for each feature the accounts labeled "automated" $A_i$, and those labeled "not automated" $N_i$. Obviously, not all features are suitable to split the accounts in two. Of those that are, some of the automated accounts may have a value lower than the cut-off. Further, some may be split by some other, more complex, principle. So instead of a cut-off value, we may need some function $G_i$. However, the more complicated the function, the more manual work will likely be needed. How to decide the cut-off value (or the function) for each feature is dependent on the particular data and is part of the manual labour that has to be put into the application of the method.

### A. Data Sets

The labeling of accounts using several features and cut-off values (or functions) could be compared to a manual annotation following several predetermind hard criteria. Each account is evaluated by each of the criteria. If an account is deemed as being automated considering one criterion we label it as such. Following this scheme we create one set of automated accounts $A = \bigcup_{i \in I} A_i$ and one set of non-automated accounts $N = \bigcap_{i \in I} N_i$. This gives us a labeled data set $D = A \cup N$, which is only limited in size by the amount of data we have available. It has a distribution over the classes following the actual data, in so far as we believe in our labels. Both of these properties are very appealing: the first for obvious reasons, the second compared to manual data usually being annotated for equal amounts of the classes, thus not reflecting the real distribution.

We can now divide $D$ into the ordinary training, validation, and evaluation data sets. As the presented method allows for easy creation of large amounts of data, we can tailor the data sets to our particular needs. For instance, if the machine learning method of our choice performs better with balanced training data we can produce such training data by under-sampling. To further monitor the developed detector we create one more data set to apply it to after the evaluation. Using this data set, the monitor data set, we do an ordinary evaluation. In addition to this, we also manually check the results considering the confusion matrix. Consider the case of a really skewed distribution of positive and negative examples, as for automated accounts under normal circumstances. We are particularly keen to consider the false positives, as we do not want a lot of false alarms in a real-world application. The ideal outcome of the presented method is a rather high false positive count, that when studied manually are found to be true positives (bots or automated accounts) to a large extent. This would give us a detector that has a high precision and has generalized from the data.

## IV. Used features

In our experiments two main types of features have been used when deciding whether an account is to be labeled automated or not, namely Twitter-specific features and more general features applicable to most online social networks. In the following, we briefly describe all features that have been extracted and used in our experiments. The interested reader is encouraged to follow the given references for a more complete explanation of the used features.

### A. *Twitter-specific features*

A tweet may in addition to the regular content contain mentions, hashtags and URLs. These ”amplifiers” can be used to spread content and reach targets more efficiently. Since automation often is a result of having the need to distribute content widely to a large audience without incurring manual work, automated Twitter accounts might exploit these amplifiers to become more effective. The following features are based on the assumption that tweets from automated Twitter accounts contain more repetitiveness and manifest a different behavior in using hashtags, mentions and URLs compared to non-automated users.

*1) Mentions-per-tweet Ratio (MTR):* This feature measures at which rate a user account is mentioning other users by calculating the ratio between the total number of mentions and the total number of tweets. Given the assumption that an automated account have some kind of purpose of spreading information to a large audience, this feature can give an indication of at which rate the user account is trying to reach out to other users. More formally, this metric is computed as:

$$MTR = \frac{total\ mentions}{total \text{ tweets}}$$

*2) Hashtags-per-tweet Ratio (HTR):* This feature measures at which rate an account is trying to reach out using various hashtags, and can be computed in following way:

$$HTR = \frac{total\ hashtags}{total \text{ tweets}}$$

*3) URLs-per-tweet Ratio (UTR):* The URLs-per-tweet ratio measures the rate at which an account is trying to redirect visitors to various URLs:

$$UTR = \frac{total \text{ URLs}}{total \text{ tweets}}$$

*4) Duplicate-URLs Ratio (DUR):* This feature quantifies the repetition of URLs by comparing the ratio between the number of unique URLs and all URLs sent from the user account:

$$DUR = \begin{cases} 1 - \frac{unique \text{ URLs}}{total \text{ URLs}} & \text{for } total \text{ URLs} > 0 \\ 0 & \text{for } total \text{ URLs} = 0 \end{cases}$$

A high ratio could indicate that the account is attempting to promote a specific URL.

*5) Duplicate-Domains Ratio (DDR):* Similarly to DUR, the duplicate-domains ratio measures the repetition in domain names of URLs:

$$DDR = \begin{cases} 1 - \frac{unique \text{ domains}}{total \text{ domains}} & \text{for } total \text{ domains} > 0 \\ 0 & \text{for } total \text{ domains} = 0 \end{cases}$$

*6) Duplicate-Mentions Ratio (DMR):* This feature measures the repetitive nature of accounts mentioning other users by comparing the ratio between the unique and total number of mentions in tweets. High repetitiveness would indicate that the account is targeting specific users only, while a very low ratio would indicate that a user is trying to reach out to a wide variety of users.

$$DMR = \begin{cases} 1 - \frac{unique \text{ mentions}}{total \text{ mentions}} & \text{for } total \text{ mentions} > 0 \\ 0 & \text{for } total \text{ mentions} = 0 \end{cases}$$

*7) Duplicate-Hashtags Ratio (DHR):* Similarly, duplicate hashtags ratio measures the amount of repetition in the use of hashtags. A high ratio would indicate that the user is trying to reach out on very specific topics while a low ratio indicates that the user is trying to reach out on a wider variety of topics:

$$DHR = \begin{cases} 1 - \frac{unique \text{ hashtags}}{total \text{ hashtags}} & \text{for } total \text{ hashtags} > 0 \\ 0 & \text{for } total \text{ hashtags} = 0 \end{cases}$$

### B. General features

Here we describe features that are, unlike the Twitter-specific features, designed to capture automation in most types of OSNs where user generated content is associated with a timestamp.

*1) Inter-tweet-content Similarity (ITCS):* One way to find automation is to calculate the similarity among tweets by measuring the average distance between all tweets. We represent each tweet using a term frequency-based bag-of-words representation. In this work we have selected to use Jaccard similarity and cosine similarity for measuring the inter-tweet-content similarity.

*2) Pearson's Chi-Square Statistics of Uniformity (PCU):* This statistic measures the uniformity in the timing distributions for different granularities based on the Pearson's Chi-Square Test. The uniformity timing distributions were calculated by binning the timestamps of tweets posted by a user into histograms of $Q$ bins for second-of-the-minute, minute-of-the-hour, hour-of-the-day, and day-of-the-week.

*3) Rao's Spacing Statistics of Uniformity (RSU):* Similarly to PCU, Rao's Spacing Statistics [7], [8] measures the uniformity in successive inter-tweet-delays for different granularities. The rationale behind this metric is that if the underlying distribution is uniform, then the successive timestamps should be approximately evenly spaced on a circle. This approach is not dependent on binning in the same way as Pearson's $\chi^2$-test and does not suffer as much from few samples.

*4) Inter-Tweet-Content Shape (ITC-S):* The inter-tweet-content shape features [9] measure the first-order entropy of the content of tweets posted by a user. Each character (including special characters such as white space, commas, etc.) of a tweet is represented by its index in the ASCII-table, thus each tweet can be represented as a sequence of numbers. The entropy is then calculated on the concatenated sequence of all tweets ordered by their timestamp. Since the numbers of the sequence are upper-bounded by the ASCII alphabet we used $Q = 255$ quantization levels with an equal-width quantization strategy so that no coarse graining would be applied to the dynamics of the sequence. The phase space reconstruction was made with a time delay of $\tau = (\text{total number of chars})/\text{tweets}$. For most accounts this roughly corresponds to $\tau \approx 60$.

*5) Inter-Tweet-Content Regularity (ITC-R):* This feature [9] measures a user's tweeting behavior by computing the higher-order entropy of the tweets posted by a user. Here, we used $Q = 256$ quantization levels with an equal-width quantization strategy to reconstruct the state space. The phase space reconstruction was made with a time delay of $\tau = (\text{total number of chars})/\text{tweets}$.

*6) Inter-Tweet-Delay Shape (ITD-S):* This feature [9] measures the first-order entropy of the delays between the tweets' timestamps. The delay can be measured with different granularities, such as delay by whole seconds, minutes, hours and days, which in itself generates four different features. Here, we used $Q = 5$ with an equal-width quantization strategy. The phase space reconstruction was made with a time delay of $\tau = 1$.

*7) Inter-tweet-delay Regularity (ITD-R):* This feature measures the periodic or regular timing of a user's tweeting behavior by computing the entropy rate of the delays between the tweets' timestamps. The delay can be measured with different granularities, such as delay by whole seconds, minutes, hours and days, which in itself generates four different features. Here, we used $Q = 5$ quantization levels with an equiprobable quantization strategy. The phase space reconstruction was made with a time delay of $\tau = 1$.

## V. Dataset creation

We have collected tweets related to the Russia-Ukraine conflict, even before the Russian annexation of Crimea took place. From this large collection of tweets we have selected 2014-04-01 to 2014-05-15 to be of extra interest from an information operation's perspective. During this time period more than 4,000,000 tweets matching our selected keywords from over 700,000 different Twitter accounts were collected. Since we are interested in studying the use of automation, we have from the collected dataset selected all user accounts which during the period sent at least one tweet or more per day on average, resulting in approximately 13,000 user accounts. A small subset of these accounts were sat aside for initial manual analysis (e.g., for selecting suitable parameter settings) and for manual verification purposes, while the rest of the users and their associated collected tweets were used for semi-automatic labeling. The semi-automatic labeling process was configured so that each feature would contribute to approximately 3.7% of the accounts being labeled as ”automated”. The chosen cut-off values resulted in 3958 accounts being labeled as automated and 6901 accounts being labeled as non-automated. The labeled data instances were then divided into a well-balanced training set (obtained using under-sampling) consisting of 3079 automated accounts and 3079 non-automated accounts, and an unbalanced test set consisting of 519 accounts being labeled as automated and 980 accounts being labeled as non-automated.

## VI. Experimental results

Based on the previously described training set, we have made use of 10-fold cross validation in order to train a random forest classifier. We have then evaluated this classifier on the remaining test set which was held out from the training. Training and test errors have been calculated and are summarized using confusion matrices in Tables I and II.

| | | Predicted | | |
|---|---|---|---|---|
| | | Automation | Non-Automation | Total |
| Actual | Automation | 3120 (tp) | 10 (fn) | 3130 |
| | Non-Automation | 53 (fp) | 3077 (tn) | 3130 |

TABLE I

THE CONFUSION MATRIX OBTAINED ON THE TRAINING SET.

Calculating the precision, recall and $F_1$-score from these matrices give us a precision of 0.974 (0.983 on the training set), a recall of 0.996 (0.997 on the training set), and a $F_1$-score of 0.985 (0.990 on the training set). Although these numbers are promising, there is an obvious risk that these are biased estimates of the true performance of this classifier since the ground truth for the test set has been obtained in the same way as the automated labeling of the training set. For this reason we have also made use of manual annotation and verification in order to give a more unbiased estimate of how well the classifier works ”in reality”. In a first attempt to assess the quality of the semi-automatic labeling and to investigate whether the classifier has been able to learn anything beyond the quite simple ”rules” utilized to label the data, we have manually checked all accounts which were misclassified on the semi-automatically labeled test set. Manual verification of the two false negative instances confirmed that the semi-automatically generated labels were correct, those accounts indeed were automated. However, among the 14 false positives, the manual analysis revealed that at least 12 out of these were in fact automated. The two remaining accounts were not easily classified as either automated or non-automated, but when looking for more recent tweets sent from these accounts it was found that both have been suspended by Twitter. This makes us believe that also these accounts have been used for some kind of automated behavior, although it was not obvious from the manual analysis of the tweets sent during the time period of interest. From these findings we can infer that the classifier has been able to generalize beyond the semi-automatic labelling of the training data.

|  |  | Predicted |  |  |
|---|---|---|---|---|
|  |  | Automation | Non-Automation | Total |
| Actual | Automation | 523 (tp) | 2 (fn) | 525 |
|  | Non-Automation | 14 (fp) | 960 (tn) | 974 |

TABLE II

THE CONFUSION MATRIX OBTAINED ON THE TEST SET. THE LABELS FROM THE SEMI-AUTOMATIC LABELING METHOD WERE USED AS GROUND TRUTH.

As a final evaluation, 50 randomly selected user accounts predicted to be automated and 50 randomly selected user accounts predicted to be non-automated were subject to human inspection and annotation. Among the 50 accounts predicted to be automated by the classifier, 49 accounts turned out to be true positives according to the manual annotation. Ten of these accounts were hard to label for the annotator, but the final judgement was based on the fact that these accounts had been suspended by Twitter on a later occasion. This yields a precision of 0.98 on the manually annotated test data. Among the 50 accounts predicted to be non-automated, 19 were correctly classified and 31 accounts were incorrectly classified according to the manual annotation. Five out of the 31 misclassified accounts were very challenging to label for the human annotator, so the final call was based on that these accounts had been suspended by Twitter on a later occasion. This yields a rather low recall of 0.38, suggesting that many truly automated accounts may be missed if deploying the classifier in real-world applications. As discussed previously in this paper, the precision is for many purposes more important than a high recall. The recall can probably be increased by adding more features or ajusting the cut-off parameter, but this is left as future work.

## VII. Conclusions and Future Work

We have in this work collected a large set of tweets related to the ongoing Russia-Ukraine conflict using the Twitter Search API. A subset of this tweet dataset has been used for training a machine learning-based random forest classifier after semi-automatic labeling of the data using a newly developed labeling method. The classifier has been evaluated on a semi-automatic labeled test set held out from the training, as well as a small human-annotated test set. The obtained results indicate that good enough precision and recall have been achieved for the classifier to be usable for real-world classification of automated and non-automated Twitter accounts.

As ongoing and future work we intend to use the learned classifier for real-time classification of Twitter accounts tweeting about topics related to the Russia-Ukraine conflict. In this way we hope to be able to get a better understanding for how automated accounts are contributing to psychological operations in the conflict. Detection of automated bots is just one, but an important, piece of the complex puzzle when trying to understand how information operations are carried out on social media in relation to modern political and military conflicts.

## Acknowledgements

This work was supported by the R&D programme of the Swedish Armed Forces.

## References

[1] Z. Chu, S. Gianvecchio, H. Wang, and S. Jajodia, “Detecting automation of twitter accounts: Are you a human, bot, or cyborg?” *Dependable and Secure Computing, IEEE Transactions on*, vol. 9, no. 6, pp. 811–824, 2012.

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

[3] C. Grier, K. Thomas, V. Paxson, and M. Zhang, “@ spam: the underground on 140 characters or less,” in *Proceedings of the 17th ACM conference on Computer and communications security*. ACM, 2010, pp. 27–37.

[4] A. H. Wang, “Detecting spam bots in online social networking sites: a machine learning approach,” in *Data and Applications Security and Privacy XXIV*. Springer, 2010, pp. 335–342.

[5] F. Benevenuto, G. Magno, T. Rodrigues, and V. Almeida, “Detecting spammers on twitter,” in *Collaboration, electronic messaging, anti-abuse and spam conference (CEAS)*, vol. 6, 2010, p. 12.

[6] C. M. Zhang and V. Paxson, “Detecting and analyzing automated activity on twitter,” in *Passive and Active Measurement*. Springer, 2011, pp. 102–111.

[7] G. S. Russell and D. J. Levitin, “An expanded table of probability values for rao’s spacing test,” *Communications in Statistics-Simulation and Computation*, vol. 24, no. 4, pp. 879–888, 1995.

[8] J. Rao, “Some contributions to the analysis of circular data,” 1969.

[9] S. Gianvecchio, M. Xie, Z. Wu, and H. Wang, “Measurement and classification of humans and bots in internet chat.” in *USENIX security symposium*, 2008, pp. 155–170.