Bot Detection: Will Focusing on Recall Cause Overall Performance Deterioration?

Tahora H. Nazer[1], Matthew Davis[1], Mansooreh Karami[1]
Leman Akoglu[2], David Koelle[3], and Huan Liu[1]

[1] Arizona State University, Tempe, AZ 85281 USA
{tahora.nazer,matt.davis,mkarami,huan.liu}@asu.edu
[2] Carnegie Mellon University, Pittsburgh, PA 15213 USA
lakoglu@andrew.cmu.edu
[3] Charles River Analytics, Cambridge, MA 02138 USA
dkoelle@cra.com

Abstract. Social bots are an effective tool in the arsenal of malicious actors who manipulate discussions on social media. Bots help spread misinformation, promote political propaganda, and inflate the popularity of users and content. Hence, it is necessary to differentiate bot accounts and human users. There are several bot detection methods that approach this problem. Conventional methods either focus on precision regardless of the overall performance or optimize overall performance, say F_1, without monitoring its effect on precision or recall. Focusing on precision means that those users marked as bots are more likely than not bots but a large portion of the bots could remain undetected. From a user's perspective, however, it is more desirable to have less interaction with bots, even if it would incur a loss in precision. This can be achieved by a detection method with higher recall. A trivial, but useless, solution for high recall is to classify every account (human or bot) as bot, hence, resulting in poor overall performance. In this work, we investigate if it is feasible for a method to focus on recall without considerable loss in overall performance. Extensive experiments with recall and precision trade-off suggest that high recall can be achieved without much overall performance deterioration. This research leads to a recall-focused approach to bot detection, REFOCUS, with some lessons learned and future directions.

Keywords: Social media · Twitter · Social Bots · Bot Detection · Recall

1 Introduction

Bots are prevalent on social media and their malicious actions have been observed repeatedly. An example of this wide-spread activity of bots was seen during the US Presidential Election in 2016. Allcott and Gentzkow [1] extensively studied this event and reported that millions of pro-Trump and pro-Clinton fake stories were shared on Facebook, in part, by bot accounts. They also provide evidence that more than 40% of fake news sources use social media to spread their content.

Thus, researchers have put great effort into understanding bots and developing methods to detect them. In supervised bot detection methods, which are the focus of this work, a labeled dataset of bots and human users is available prior to training a machine learning classifier. Using these labels, we can learn characteristics, also know as features (described in detail in Section 2), that discriminate bots from humans and use them to build classifiers that predict class labels (bot or human). The classifiers are then tested on unobserved datasets and evaluated using two prominent metrics: precision and recall.

A common theme among previous bot detection methods is attempting to maximize precision [4, 9, 16]. This is one extreme: the sole purpose is to minimize false positives and avoid mistakenly marking a human user as bot. By doing this, detection methods avoid removing human users from the site but leave many bots undetected. The other extreme is eliminating bots from social media at the price of removing human users. This approach is not preferable either. A method for finding a trade-off between precision and recall is optimizing for F_1 score which is the harmonic mean between precision and recall. Harmonic mean is dominated by the minimum of its arguments. Hence, F_1 cannot become arbitrarily large when either precision or recall is unchanged and the other metric is increased. This prevents bot detection algorithms from landing on trivial solutions (marking all users as bots or humans) to gain high F_1. However, considering the same weight for precision and recall in F_1 prevents us from having control over the final values of either precision or recall. In other words, two classifiers are considered equally good if they have the same F_1 regardless of the fact that one might result in higher recall and the other one a higher precision. The ideal case is finding a solution close to optimum F_1 that allows us to focus on precision or recall depending on the application.

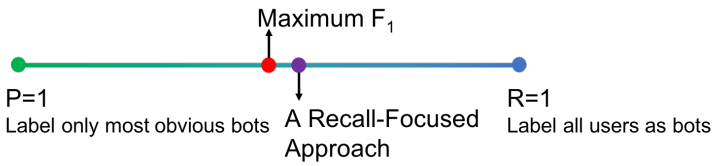


Fig. 1: Our goal is having a recall-focused approach close to the optimal F_1.

To align with corporate goals (having a large number of active users and retaining human users by avoiding accidentally suspending their accounts), bot detection models with high precision are preferable. However, from a user's perspective, both social media users and researchers alike, the preferable situation is encountering a minimum number of bots. So, in this case, high recall is preferred. In this work, we focus on developing a supervised algorithm aligned with a user's perspective: a REcall FOCUSed bot detection model, REFOCUS. We use multiple real-world datasets to show how we can find a sweet spot between blindly optimizing for F_1 or recall as shown in Fig. 1. We also compare REFOCUS with state-of-the-art bot detection models to show that focusing on recall does not necessarily result in overall performance deterioration in terms of F_1.

2 Supervised Bot Detection Methods

To use supervised bot detection models, one must identify differences among bot and human users in terms of features such as content or activity in a labeled dataset. Then, a classifier is trained on the features and labels to distinguish bots from humans in an unobserved dataset. Different classification methods can be used for this purpose such as Support Vector Machines [11], Random Forests [9], and Neural Networks [8]. We describe some common user features below:

- *Content*: the measures in this category focus on the content shared by users. Words, phrases [16], and topics [11] of social media posts can be a strong indicator of bot activity. Also, bots are motivated to persuade real users into visiting external sites operated by their controller, hence, share more URLs in comparison to human users [4, 13, 17]. Bots are observed to lack originality in their tweets and have large ratio of retweets/tweets [14].
- *Activity Patterns*: Bots tweet in a "bursty" nature [4, 10], publishing many tweets in a short time and being inactive for a longer period of time. Bots also tend to have very regular (e.g. tweeting every 10 minutes) or highly irregular (randomized lapse) tweeting patterns over time [18].
- *Network Connections*: bots connect to a large number of users hoping to receive followers back but the majority of human users do not reciprocate. Hence, bots tend to follow more users than follow them back [4].

As bots become more complex and harder to detect [5], bot detection methods incorporate a larger number of features and datasets with more samples of humans and bots. One of the recent approaches is BotOrNot [6]. This method exploits 1,150 features from five categories: user-based, friends, network, temporal, language, and sentiment [16]. The initial model was trained on a dataset of ∼40,000 bots and humans and has been updated multiple times using seven more datasets with total of ∼87,000 samples. The strength of this method has encouraged researchers to use it to label ground-truth datasets [7]. We will compare our proposed method with BotOrNot in Section 5.

3 Data for Supervised Bot Detection

To show the robustness of our model with respect to the language, topic, time, and labeling mechanism, we use three datasets represented in Table 1.

Table 1: Statistics of the datasets used in this study.

Property	**Arabic Honeypot**	**Social Spambot 1**	**Social Spambot 2**
Tweets	637,435	4,449,395	4,257,918
Retweets	209,703	782,267	754,104
Human Accounts	2,317	1,083	1,083
Bot Accounts	1,978	991	464
Bot Ratio	46.05%	47.78%	29.99%
Labeling Approach	Honeypot	Manual	Manual

As seen in Table 1, in this work, we utilize three existing bot detection datasets. We describe how each of these raw datasets were collected and what they contain in Section 3.1. Then, we specify how we preprocessed the raw data using a content-based feature extraction method in Section 3.2.

3.1 Datasets

The first dataset is a honeypot dataset collected by Morstatter et al. [11] which we refer to as the Arabic Honeypot dataset. It was collected using a network of 9 honeypot accounts which tweeted Arabic phrases, as well as randomly following and retweeting each other. Any user who followed a honeypot was considered a bot, because bot behaviors are sporadic and provide no intelligent information to humans. For collecting a set of human users, the authors manually inspected users who tweeted same Arabic phrases as some of the bots, then crawled data for them and other users that the inspected users immediately followed; assuming that humans only follow other humans and not bots. In August 2018, we re-crawled this dataset using the tweet IDs shared by the authors.

Additionally, we employ two datasets introduced by Cresci et al. [5] in their previous work on detecting social bots on Twitter namely: `test set #1` and `test set #2`. We call these datasets Social Spambots 1 and 2, respectively, in our work. Each dataset is a combination of social spambots and human users on Twitter. To collect the human user accounts, Cresci et al. contacted random users, asked a natural language question, and manually evaluated if the user was a human. Social Spambots 1 contains these genuine accounts plus social bots that were discovered during the 2014 Mayoral election in Rome, Italy which were used to retweet a candidate within minutes of his original posting. Social Spambots 2 includes the genuine accounts and social bots that advertised products on *Amazon.com* by deceitfully spamming URLs which point to the products. We obtained these datasets directly from the BotOrNot Bot Repository [6].

3.2 Feature Extraction

Bot accounts are created by malicious actors to serve specific purposes. Thus, their content can be a strong indicator to expose such potentially automated accounts. The problem with using content for bot detection is that the raw text features are of high dimensionality and sparse. Inspired by the recent advances of topic modeling, we adopt latent Dirichlet allocation (LDA) [3] to obtain a topic representation of each user. LDA, which treats each document as a distribution over topics and each topic as a distribution over the vocabulary in the dataset, has been proven useful for extracting latent semantics of documents. As such, we use LDA to extract features from users' tweet content. In this work, each user is considered one document and the content of that document is his tweets. We trained separate LDA models on each of the three datasets and develop classifiers independently. We follow the assumption that, since bots are naturally more interested in certain topics, denoting each user as a distribution over different topics may help to better identify them from regular accounts [11].

4 A Recall-Focused Approach – REFOCUS

A bot detection classifier generates the probability of being a bot (belonging to the positive class) for each instance in the dataset. To assign a binary label to users, a classifier uses a threshold (commonly set to 0.5 [12]) to decide; if the probability of being a bot is more than the classification threshold then the user is labeled as a bot, otherwise as a human. Based on the assigned labels, classifiers can be evaluated using precision (P) and recall (R) defined bellow:

$$P = \frac{tp}{tp + fp}, \quad R = \frac{tp}{tp + fn} \tag{1}$$

Precision and recall can be independently maximized easily. A trivial approach for increasing the recall is lowering the classification threshold and classifying more users as bots. Alternatively, increasing the classification threshold results in labeling most users as humans, with only the unquestionably obvious bot users labeled as bots, and causes a trivial increase in precision. However, precision and recall are not independent from each other: increasing one might result in decreasing the other. One approach for finding a trade-off between precision and recall is using the F_β score. F_β is the *weighted* harmonic mean of precision and recall [15] and is defined as follows:

$$F_\beta = \frac{(1 + \beta^2)PR}{\beta^2 P + R} \tag{2}$$

With β values greater than one, β times more weight is put on recall and for values less than one, β times more weight is associated with precision.

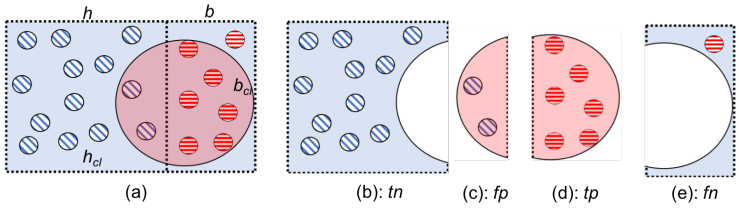


Fig. 2: Illustration of true negative - (b): tn, false positive - (c): fp, true positive - (d): tp, and false negative - (e): fn for a classifier trained on dataset (a) when the classifier labels a subset of users as bots (positive class) - b_{cl} - and the rest as humans (negative class) - h_{cl}.

4.1 Searching for a Trade-off: Selecting β

Our goal is optimizing for recall, hence, we utilize F_β with $\beta > 1$ to find the best classification threshold: a sweet spot between where F_1 (overall performance) is maximized and where $R = 1$. The framework of our recall focused approach is presented in Fig. 3. We divide the dataset to 90% $Train$ and 10% $Test$. Then, for ten iterations, we divide Train to 90% $Train_i$ and 10% Val_i which are training and validation sets respectively; $Train_i$ is 81% of the whole data and Val_i is

9%. In each iteration, we train a classifier C_i on $Train_i$, change the classification threshold between 0.1 and 0.9 with 0.1 steps, and find the threshold that results in the highest F_β score on Val_i; we call this threshold t_i. After the tenth iteration, we get an average of the thresholds t_1 to t_{10} to find the average threshold t. Then we train a classifier, C, on $Train$ and using t, we find the precision, recall, and F_1 score. We repeat this process ten times and report the average of precision, recall, and F_1 scores as our final results.

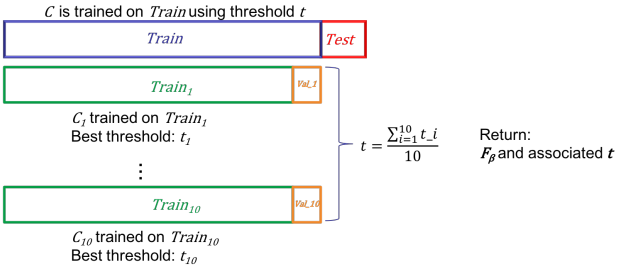


Fig. 3: Framework for the proposed bot detection model, REFOCUS.

We need to test different values of β in the training phase to find the best classification threshold using F_β. As we increase β, precision has a non-increasing trend and recall has a non-decreasing trend. This happens because as we increase β we put more weight on recall in comparison to precision. More formally

$$R_{\beta_i} \geq R_{\beta_j} \quad and \quad P_{\beta_i} \leq P_{\beta_j} \qquad if \quad \beta_i > \beta_j \tag{3}$$

Due to this non-increasing pattern of precision with increase of β, we prefer to maintain a low β as long as we do not sacrifice the chance of achieving a higher recall with minor loss in precision. To find the right β, we start from $\beta = 1$ and in each step we choose the current β as β_{opt} if

$$(R^{\beta} - R^{\beta_{opt}}) > (F_1^{\beta_{opt}} - F_1^{\beta}) \tag{4}$$

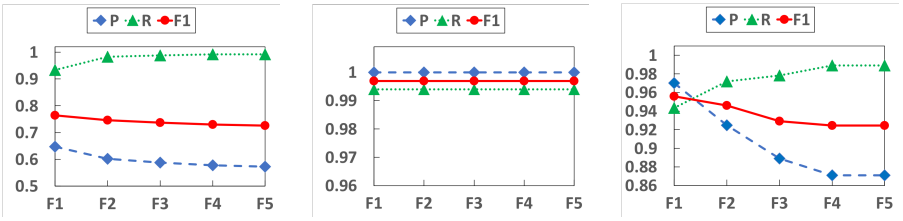
Meaning that we choose a larger β if the gain in R is more than the loss in F_1.

5 Experiments

In this section we empirically investigate the performance of our proposed approach. First, we investigate the effect of β and then, we compare REFOCUS with baseline bot detection models in terms of P, R, and F_1.

5.1 Searching for the Right β

It is intuitive that using a F_β when $\beta > 1$ for training a classifier helps us find the classification threshold that results in higher recall as compared to when $\beta = 1$. However, it raises two questions: (1) what is best value of β and can we increase it indefinitely to reach the highest recall possible? (2) Does the model trained using $\beta > 1$ still perform well in terms of F_1 or we will drastically lose precision? We answer the first question here and the second one in Section 5.2.



(a) Arabic Honeypot (b) Social Spambots 1 (c) Social Spambots 2

Fig. 4: Effect of β on precision (P), recall (R), and overall performance (F_1). In each dataset, we change β from 1 to 5, use F_β for finding the best classification threshold in the training phase and report P, R, and F_1 on the test set.

We test our model on three datsets: Arabic Honeypot, Social Spambot 1, and 2. The results are shown in Fig. 4. In Social Spambots 1, we do not observe any change in the overall performance in terms of F_1 as we change the β. This can be due to the way this dataset was collected resulting in humans and bots being quite distinct from each other. This distinction causes the classifier perform well no matter what the threshold is. Hence, any of the F_β scores can be used to find the best classification threshold. In Arabic Honeypot and Social Spambot 2, we see some variations in precision, recall, and overall performance. $\beta = 2$ gives us the best trade-off between precision and recall because the loss in the overall performance is smaller that the gain in recall; in other words, the slope of recall line is larger than the slope of F_1 line. Further increase in β does not provide enough gain on recall in comparison to the loss in the overall performance, hence, we stop at F_2.

5.2 Testing the Overall Performance

For comparing the overall performance of REFOCUS with other bot detection methods, we need to decide on the number of topics in LDA and the classification model. Due to the similarity between our feature extraction and the one by Morstatter et al. [11] we follow their observation that 200 topics generated the highest F_1 in the Arabic Honeypot dataset and set number of topics to 200.

We test multiple classification algorithms that are observed to have high performance in the problem of bot detection [2] to find the best fit for REFOCUS:

Table 2: Performance of REFOCUS when implemented using different classifiers.

	Arabic Honeypot			Social Spambot 1			Social Spambot 2		
Classifier	P	R	F_1	P	R	F_1	P	R	F_1
Decision Tree	0.738	0.746	0.741	0.995	0.996	0.996	0.933	0.956	0.944
Random Forest	0.460	1.0	0.630	0.992	0.996	0.994	0.854	0.947	0.897
Logistic Regression	0.603	0.984	0.748	1.0	0.996	0.998	0.984	0.919	0.950
SVM	0.601	0.983	0.746	1.0	0.993	0.996	0.924	0.971	0.945

Decision Tree, Random Forest, Logistic Regression, and SVM. We use Python Scikit-learn package [12] for implementation with default settings except $max\_depth = 1$ for Random Forest and $max\_iter = 1$ for Logistic Regression to avoid overfitting. As shown in Table 2, all classifiers achieve very similar (difference less than 0.5%) F_1 score except for Random Forest that has lower performance. We choose SVM for the rest of our experiments because it has similar or higher R and similar F_1 . Worth mentioning that our method can be built on top of any classifier to help improve recall without sacrificing the overall performance.
We compare our proposed approach, REFOCUS, with two baselines:

- *SVM*: REFOCUS uses SVM to train multiple classifiers on subsamples of the dataset and learns the best recall-precision trade-off using F_β. Hence, we compare our method with SVM when its parameters are set to default and it generates the class labels (1 or -1) using 0.5 as threshold. Users are represented with 200 LDA topics and we use 10-fold cross validation.
- *BotOrNot* [6, 16]: this supervised bot detection model exploits 1150 features in six categories: user-based, friends, network, temporal, content and language, and sentiment. The model uses a Random Forest classifier and is trained on multiple publicly available datasets. BotOrNot has been used for generating ground-truth due to its performance.

We perform two sets of experiments. In the first one, we use the Arabic Honeypot dataset. We use an LDA model with 200 topics to extract features from the dataset then we apply REFOCUS and report the results. However, using this dataset raises the concern that our approach might not perform as well on non-Arabic tweets. Hence we also perform the second experiment. We follow the same procedure but use the datasets that were collected by Cresci et al. [5]. These datasets (as explained in Section 3) have three advantages: they are among the most recent publicly available labeled datasets for bots and include newer bots, they use manual labeling which is different from the honeypot dataset, and a majority of the tweets are in English. Hence, by testing our approach on Cresci's datasets, we show that our model performs well regardless of the language of tweets and is resilient to new bots that emerge on social media.

The results are presented in Table 3. For the experiments on Cresci's datasets, we do not balance the classes due to small size of the data. Hence, we also include the ROC AUC in our results. The ROC AUC for a classifier that randomly assigns labels to instances is 0.5 regardless of the class balance and is a helpful metric to assess classifiers when the samples of one class are more than the other. Reserving the class imbalance is also helpful to mimic the real world scenario where bots are a small portion of all users on social media [16].

In the first experiment, on the Arabic Honeypot dataset, SVM has higher precision and lower recall in comparison to REFOCUS. The reason is that SVM only labels a user as bot if the predicted probability of being a bot for that user is over 0.5. However, our method learns the best threshold for optimizing recall while reaching a high F_1. Hence REFOCUS chooses a lower threshold (0.35 in this case). This choice results in 2% lower F_1, however, we are willing to tolerate this loss due to 6% gain in recall. BotOrNot performs considerably worse in this

Table 3: Comparison between REFOCUS and baseline bot detection methods.

Dataset	Method	P	R	F_1	ROC
Arabic Honeypot	SVM	0.655	0.919	0.765	0.849
	BotOrNot	0.472	0.523	0.496	0.514
	REFOCUS	0.601	0.983	0.746	0.849
Social Spambot 1	SVM	1.0	0.991	0.995	0.997
	BotOrNot	0.963	0.961	0.962	0.969
	REFOCUS	1.0	0.993	0.996	0.997
Social Spambot 2	SVM	0.986	0.915	0.949	0.996
	BotOrNot	0.954	0.939	0.946	0.957
	REFOCUS	0.924	0.971	0.945	0.996

dataset in comparison to the Social Spambot datasets. The reason is that Social Spambot datasets have been used in training BotOrNot and it is expected for classifiers to have lower performance on unseen datasets (e.g. Arabic Honeypot).

In the second experiment, we test our method on two non-Arabic datasets which are obtained using a manual annotation method to show that our results are robust to variations in datasets such as language. In Social Spambots 1, SVM and our proposed approach perform almost identically with an slightly better recall in REFOCUS. The reason is that the differences between instances in human and bot classes are well captured by the classifiers to the extent that the classifier (either SVM or REFOCUS) are very confident in the labeling. Hence, each instance gets a high probability of being in its actual class and changing the threshold does not change the classification results much. We also observe that our approach outperforms BotOrNot. On Social Spambots 2, SVM and BotOrNot outperform our approach in precision and have lower recall, similar to the Arabic Honeypot dataset, because they are not is not designed to optimize on recall. F_1 of our approach is similar to the baselines.

6 Conclusion and Future Directions

The dominant trend among the previously proposed methods for bot detection is solely focusing on precision, making sure that no human user is marked as a bot, or optimizing for F_1. In this work, we showed that we can focus on recall of a bot detection model without sacrificing the overall performance. We tested our method on three real-word datasets and observed that using F_2 score in the training phase results in finding the best classification threshold for optimizing recall and having high overall performance in terms of F_1. In the future, we wish to explore the robustness of our method on translated datasets and also measure its effectiveness in discriminating different types of bots in a dataset.

7 Acknowledgements

Support was provided, in part, by National Science Foundation grant 1461886 and the Office of Naval Research through N000141310835 and N000141612257. We would like to thank anonymous reviewers for their valuable feedback.

References

1. Allcott, H., Gentzkow, M.: Social media and fake news in the 2016 election. Journal of Economic Perspectives **31**(2), 211–36 (2017)
2. Alothali, E., Zaki, N., Mohamed, E.A., Alashwal, H.: Detecting Social Bots on Twitter: A Literature Review. In: IIT. pp. 175–180. IEEE (2018)
3. Blei, D.M., Ng, A.Y., Jordan, M.I.: Latent dirichlet allocation. Journal of machine Learning research **3**(Jan), 993–1022 (2003)
4. Chu, Z., Gianvecchio, S., Wang, H., Jajodia, S.: Who is Tweeting on Twitter: Human, Bot, or Cyborg? In: ACSAC. pp. 21–30. ACM (2010)
5. Cresci, S., Di Pietro, R., Petrocchi, M., Spognardi, A., Tesconi, M.: The Paradigm-Shift of Social Spambots: Evidence, Theories, and Tools for the Arms Race. In: The Web Conference. pp. 963–972 (2017)
6. Davis, C.A., Varol, O., Ferrara, E., Flammini, A., Menczer, F.: BotOrNot: A System to Evaluate Social Bots. In: The Web Conference. pp. 273–274 (2016)
7. Khaund, T., Al-Khateeb, S., Tokdemir, S., Agarwal, N.: Analyzing Social Bots and Their Coordination During Natural Disasters. In: SBP-BRiMS. pp. 207–212. Springer (2018)
8. Kudugunta, S., Ferrara, E.: Deep Neural Networks for Bot Detection. Information Sciences **467**, 312–322 (2018)
9. Lee, K., Eoff, B.D., Caverlee, J.: Seven Months with the Devils: A Long-Term Study of Content Polluters on Twitter. In: ICWSM. pp. 185–192. AAAI (2011)
10. Lee, S., Kim, J.: Early Filtering of Ephemeral Malicious Accounts on Twitter. Computer Communications **54**, 48–57 (2014)
11. Morstatter, F., Wu, L., H. Nazer, T., Carley, K.M., Liu, H.: A New Approach to Bot Detection: Striking the Balance between Precision and Recall. In: ASONAM. pp. 533–540. IEEE (2016)
12. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, E.: Scikit-learn: Machine Learning in Python. Journal of Machine Learning Research **12**, 2825–2830 (2011)
13. Ratkiewicz, J., Conover, M., Meiss, M., Gonçalves, B., Patil, S., Flammini, A., Menczer, F.: Truthy: Mapping the Spread of Astroturf in Microblog Streams. In: The Web Conference. pp. 249–252. ACM (2011)
14. Ratkiewicz, J., Conover, M., Meiss, M., Gonalves, B., Flammini, A., , Menczer, F.: Detecting and Tracking Political Abuse in Social Media. In: ICWSM. pp. 297–304. AAAI (2011)
15. Rijsbergen, C.J.V.: Information Retrieval. Butterworth-Heinemann, Newton, MA, USA, 2nd edn. (1979)
16. Varol, O., Ferrara, E., Davis, C.A., Menczer, F., Flammini, A.: Online Human-Bot Interactions: Detection, Estimation, and Characterization. In: ICWSM. pp. 280–289. AAAI (2017)
17. Xie, Y., Yu, F., Achan, K., Panigrahy, R., Hulten, G., Osipkov, I.: Spamming Botnets: Signatures and Characteristics. ACM SIGCOMM Computer Communication Review **38**(4), 171–182 (2008)
18. Zhang, C.M., Paxson, V.: Detecting and Analyzing Automated Activity on Twitter. Passive and Active Measurement (PAM) **LNCS 6579**, 102–111 (2011)