Information Sciences 467 (2018) 312–322

Contents lists available at ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

Deep neural networks for bot detection

Sneha Kudugunta[a], Emilio Ferrara[b,*]

[a] Indian Institute of Technology, Hyderabad, Hyderabad, India
[b] USC Information Sciences Institute, Marina Del Rey, CA, USA

ARTICLE INFO

Article history:
Received 27 February 2018
Revised 4 August 2018
Accepted 8 August 2018
Available online 9 August 2018

Keywords:
Social media networks
Web and social media
Social bots
Deep learning
Deep neural networks

ABSTRACT

The problem of detecting bots, automated social media accounts governed by software but disguising as human users, has strong implications. For example, bots have been used to sway political elections by distorting online discourse, to manipulate the stock market, or to push anti-vaccine conspiracy theories that may have caused health epidemics. Most techniques proposed to date detect bots at the account level, by processing large amounts of social media posts, and leveraging information from network structure, temporal dynamics, sentiment analysis, etc. In this paper, we propose a deep neural network based on contextual long short-term memory (LSTM) architecture that exploits both content and metadata to detect bots at the tweet level: contextual features are extracted from user metadata and fed as auxiliary input to LSTM deep nets processing the tweet text. Another contribution that we make is proposing a technique based on synthetic minority oversampling to generate a large labeled dataset, suitable for deep nets training, from a minimal amount of labeled data (roughly 3000 examples of sophisticated Twitter bots). We demonstrate that, from just one single tweet, our architecture can achieve high classification accuracy (AUC $>$ 96%) in separating bots from humans. We apply the same architecture to account-level bot detection, achieving nearly perfect classification accuracy (AUC $>$ 99%). Our system outperforms previous state of the art while leveraging a small and interpretable set of features, yet requiring minimal training data.

1. Introduction

During the past decade, social media like Twitter and Facebook emerged as a widespread tool for massive-scale and real-time communication. These platforms have been promptly praised by some researchers for their power to democratize discussions [31], for example by allowing citizens of countries with oppressing regimes to openly discuss social and political issues. However, due to many recent reports of social media manipulation, including political propaganda, extremism, disinformation, etc., concerns about their abuse are mounting [19].

One example of social media manipulation is the use of bots (a.k.a. social bots, or *sybils*), user accounts controlled by software algorithms rather than human users. Bots have been extensively used for disingenuous purposes, ranging from swaying political opinions to perpetuating scams. Existing social media bots vary in sophistication. Some bots are very simple and merely retweet specific posts (based on some rules), whereas others are sophisticated and have the capability to even interact with human users.

* Corresponding author.
E-mail addresses: cs14btech11020@iith.ac.in (S. Kudugunta), emiliofe@usc.edu (E. Ferrara).

https://doi.org/10.1016/j.ins.2018.08.019
0020-0255/

The challenge of bot detection has been taken seriously by our research community [43]. Different approaches have been proposed to detect social media bots [17]. Supervised learning, in particular, exhibited promising results [13,38,45]: examples of activity of human users and bots, labeled as such, can be fed to machine learning algorithms; trained models are then used to classify unforeseen accounts, leveraging available data, e.g., obtained via the Twitter API. Alternatives based on unsupervised learning aimed at identify large-scale behavioral anomalies and associate them to bot accounts.

However, most, if not all, of the successful methods introduced so far detect bots at the account level. This means that, given a record of activity (e.g., a few hundred tweets posted by a user), the algorithm would determine whether the scrutinized account is likely a bot or not. This type of approaches tend to focus on the overall account's activity, e.g., the content and sentiment of user posts, the network structure, and the temporal usage patterns.

Though quite successful, account-level bot detection approaches are expensive as they require significant amounts of data for each user to be scrutinized, as well as large labeled datasets for training purposes. In contrast, most available labeled datasets have at most a few hundreds examples of tweets posted by a few thousands bots. For a comprehensive survey of bot detection methods, we direct the reader to our recent review [17].

1.1. Research questions & contributions

These fundamental limitations pose two research questions, that we try to address in this paper:

RQ1: Is it possible to accurately predict whether a given tweet has been posted by a bot or human account?
RQ2: Is it possible to enhance existing labeled datasets to produce more examples of bot and human accounts without the additional (and very expensive) data collection and annotation steps?

The contributions we provide here aim to address these challenges:

1. We advance the problem of classifying individual social media accounts from single observations, i.e., determining whether a single tweet comes from a Twitter bot or from a human user. We demonstrate that tweet-level bot detection is possible and can be very accurate: by exploiting both textual features and tweet metadata, we detect bots from single tweets and even exceed the performance of earlier works that make use of a given user's entire profile and recent posting history.
2. As a technical contribution, we introduce the concept of a *Contextual LSTM* (Long Short-Term Memory) deep neural network [23], an architecture that takes both the tweet text and the tweet metadata as an input. Related architectures that use side-information to enhance recurrent model representations have been alluded to by some authors before, primarily in the context of language models, but has never been used in the context of social media classification—to the best of our knowledge. The proposed architecture allows us to reach state-of-the-art performance in bot detection (i.e., over 96% AUC scores).
3. Finally, we introduce a technique based on the usage of *synthetic minority oversampling* [9] to enhance existing datasets by generating additional labeled examples. This will allow us to achieve near perfect classification performance on the account-level bot detection task, by leveraging only a minimal number of features and very small training datasets.

1.2. Impact of this work

A successful tweet-level bot detection approach would potentially overcome the limitations presented above, namely the need for computationally expensive models that require large numbers of features, large labeled datasets for training purposes, and access to the recent history of activity of the user profile to scrutinize.

Given the same pool of users, a tweet-based bot detection approach would have significantly more labeled examples to exploit. For instance, in the dataset we use in this paper (discussed in the next section), we have labels for 3474 human users, who overall generated 8,377,522 tweets; we also have labels for 4912 social bot accounts, who generated 3,457,344 tweets.

A tweet-level detection approach would be capable of leveraging nearly 12 million labeled datapoints, while an account-level detection system would only be able to exploit about eight thousands examples of bots and human accounts, while using those millions of tweets to learn patterns associated with the originating accounts.

Shifting to tweet-level bot detection, and thus having training data orders of magnitude larger than otherwise, makes the problem of bot detection far more amenable to the usage of deep learning models. Such techniques benefit greatly from vast amounts of labeled data, showing extremely high performance in many contexts where such resources are available [29], from image classification [28] to mastering games [35,40].

Traditional deep learning techniques used for text classification purposes (as well as in the broader context of language models) rely solely on textual features (e.g., characters or n-grams) [25]. A straightforward implementation of such techniques to tweet-level bot detection could be based exclusively on tweet texts as inputs for the deep neural network of choice. However, prior results in bot detection suggested that tweet text alone is not highly predictive of bot accounts [17]. Exploiting additional features such as account metadata, network structure information, or temporal activity patterns, have been found to yield more robust and accurate results [17].

To draw a parallel with recent advances in natural language processing (NLP) powered by deep learning, we here propose a novel *Contextual LSTM* architecture that utilizes both tweet text and account metadata (which are provided by the Twitter API alongside with the tweet itself, and do not require extra data collection steps) to yield a high classification accuracy.

We hypothesize that the proposed model can be used in other deep learning applications where multimodal data are available for such types of classification tasks.

A successful tweet-level bot detection framework also has interesting practical implications:

- Identifying instances of large numbers of bot-generated tweets coming from a single account would enable us to detect account that mix bot-generated tweets with human generated one.
- Since tweets are often viewed as a part of a feed containing both genuine and bot-generated tweets, it is potentially useful to be able to flag isolated tweets as possibly bot-generated.

1.3. Related work

Bots (a.k.a., social bots, or *sybil accounts*) have been found guilty of polluting social media conversations in a variety of scenarios [17]. Reports of online manipulation mediated by bots span political conversation [7,24,32,42,46,49], fake news [5,16], conspiracy theories [43], stock market manipulation [14], public health [2,3], propaganda [1,4,15,18,47], and, in some rare occasions, bots have been used for positive, rather than nefarious, purposes [36]. The research community promptly responded to the problem of the increasing pervasiveness of bots in platforms like Twitter and Facebook. A wealth of strategies and frameworks have been proposed to address the challenge of bot and campaining detection in these environments [17,46]. A recent review [17] proposed a taxonomy describing three types of approaches: *(a)* methods based on social network; *(b)* systems based on crowd-sourcing and human computation; *(c)* algorithms based on predictive features that separate bots from humans. Our framework falls in the last category.

Bots exhibit great variability and diversity in terms of behavior, capabilities, and intent: this was illustrated with a categorization scheme proposed in another recent survey [34]. Another recent white paper discussed the capabilities of bots powered by sophisticated Artificial Intelligence. Bots also attracted the attention of the cyber security research community: Sometimes, large groups of bots are controlled by the same entity, called *bot master*, acting behind the scenes in a command-and-control fashion, in analogy to traditional *botnets* used to deploy cyber attacks and other cyber-security threats, as demonstrated on Twitter as well [1].

Much work on bot detection assumes extensive access to social media data. For example, Wang and collaborators used clustering techniques to identify large-scale behavioral anomalies [48], while other authors adopted supervised learning to analyze all accounts of some platform and separate bots from humans [38,50]. Although these approaches can be useful, for example to detect large-scale bot infiltrations, they can be implemented exclusively by social media service providers with full access to data and systems. Some of them published studies showing the effectiveness of some implementations, e.g., SybilRank [8], or the Facebook Immune System [41]. To obviate the limitation of unlimited data access, other techniques have been designed to require smaller samples of user activity, and fewer labeled examples of bot and human users. Examples of such trend include the classification system proposed by Chu et al. [10], the system based on crowd-sourcing designed by Wang et al. [48], the detection techniques based on NLP presented by Clark et al. [11], and BotOrNot [13].

To allow for bot detection at the user level, all these methods still require the analysis of some historical user data, either by indirect data collection [10,11,48], or, like in the case of BotOrNot [13], by interrogating the Twitter API (which imposes strict rate limits, making it impossible to do large-scale bot detection). To the best of our knowledge, no tweet-based detection framework existed prior to this work. We filled this gap by designing a LSTM deep neural network based on combinations of textual features and user metadata that is capable of determining if a single tweet is being posted by a human or a bot with extremely high accuracy. The same architecture shows nearly-perfect accuracy in the user-level bot detection task.

2. Dataset

The dataset used in our work is the one presented by Cresci and collaborators [12], which contains an entirely new breed of social bots. We use a mixture of the groups `genuine accounts, social spambots #1, social spambots #2 and social spambots #3`.

All these subsets of data combined together account for 8386 user accounts, and 11,834,866 tweets to train on. A group-wise breakdown may be seen in Table 1.

Although many established techniques use a large number of features ([13], for example uses over 1500 features), recent research [16,17] shows that similar high performance can be obtained by using a minimal number of features. For account-level bot detection, we use the following features:

- Statuses Count
- Followers Count
- Friends Count
- Favorites Count
- Listed Count

Table 1
Breakdown of the dataset used to train our models. The dataset was obtained from Cresci and collaborators [12].

Dataset	Accounts	Tweets
genuine accounts	3474	8,377,522
social spambots #1	991	1,610,176
social spambots #2	3457	428,542
social spambots #3	464	1,418,626

- Default Profile
- Geo Enables
- Profile Uses Background image
- Verified
- Protected

Similarly, for tweet-level classification we use only 6 features, apart from the tweet textual-content itself:

- Retweet Count
- Reply Count
- Favorite Count
- Number of Hashtags
- Number of URLs
- Number of Mentions

The choice of limiting the size of the feature set is motivated by two important reasons:

- **Model efficiency:** A reduced set of features yields very efficient models that can be trained faster and are less prone to overfitting, which is a common issue in social media data mining.
- **Interpretability:** A limited set of features with a clear meaning, like the ones provided by account metadata, allows to produce interpretable models. This is a very important point, especially when combined with deep learning strategies that are notoriously hard to interpret.

Our choice goes somewhat in antithesis to that of many feature-based systems, which are designed to generate and leverage as many features as possible, but whose computational efficiency is suboptimal and whose interpretability is challenging.

3. Methods

In this study, we face two classification tasks: *(i)* account-level bot detection, and *(ii)* tweet-level bot detection. In the following, we describe the methodological approaches that we adopted to address these two challenges.

3.1. Task 1: account-level classification

Previous work on account-level classification has found that user metadata tends to be the best predictor for bot detection [16,17]. As presented in Section 2, we use a minimal number of highly interpretable features that require little to no data preprocessing. This enables us to use a multitude of out-of-the-box classical machine learning approaches as baselines, listed in Section 4.

In this task, we found that most of these approaches have a satisfactory performance, above AUC 90%. The most successful one is based on Random Forest, yielding an AUC of 98.45%.

However, significant performance gains were observed by balancing the dataset via *oversampling techniques*, specifically the synthetic minority oversampling technique (SMOTE) [9].

The SMOTE algorithm generates samples based on the feature space of the minority examples (i.e., the class that has the fewer number of labeled datapoints), and is a powerful method that has seen successfully across many domains [21]. Specifically, we use a combination of SMOTE and two undersampling techniques. Such data enhancement techniques are used to remove any bias introduced by oversampling. Here, we combine SMOTE with data enhancement via (1) *Edited Nearest Neighbors* (ENN) and (2) *Tomek Links*. These two combinations have been found to give excellent results on imbalanced data [6]. Here, we apply SMOTE to the entire dataset and ENN/Tomek Links only to the majority class.

Although combining SMOTE and undersampling through Tomek Links (SMOTOMEK) does not improve all results, as we will discuss later, significant improvement is seen by combining SMOTE and undersampling through ENN (SMOTENN) across all models. With SMOTENN, near perfect classification accuracy is achieved with the best model being an AdaBoost Classifier at 99.81% AUC.

Our results will suggest that near-perfect accuracy bot detection at the account level can be achieved even without complex deep learning architectures. The same does not hold for the next task, i.e, that of detecting bots from single observations.

3.2. Task 2: tweet-level classification

We now introduce the new problem of determining from a single datapoint (e.g., one tweet) whether its author is a bot or not.

The approaches that use tweet content tend to use feature engineering and specific characteristics of the tweets, such as those extracted via Parts-of-Speech tagging, by counting the number of hashtags or by measuring tweet dissimilarity [17]. For bot detection in particular, there is a dearth of convincing and successful approaches based on using single observations.

As a baseline, we attempted to use an approach similar to that presented in Section 3.1 by exploiting only the features described in Section 2. Without oversampling, however, none of these methods exceeded an AUC of 77%. By means of oversampling via SMOTE, followed by undersampling through ENN, however, the results somewhat improved, as we will discuss in Section 4.1.

Many of the state-of-the-art techniques in Natural Language Processing (NLP) that use textual content tend to focus on inferring content style. These methods have been proven ineffective against more advanced social bots [12]. To overcome the limitations of traditional techniques, we use Long Short Term Memory (LSTM) models [23], a superior variant of Recurrent Neural Networks (RNNs) [26]. RNNs and their variants have been found to be extraordinarily effective in many NLP tasks, given their ability to learns relationships in sequential data [20].

To transform the tweet texts into a form amenable for processing via LSTMs, we adopt an embedding strategy: we use a pre-trained set of *Global Vectors for Word Representation* (GloVE) specific for Twitter data [37]. GloVE is a log-bilinear regression model that uses global matrix factorization and local context windows to effectively learn the substructure of natural language, by training on word co-occurrence. These resulting world vector representations capture semantic and syntactic regularities typical of natural language. Prior to this embedding strategy, we preprocess the tweets by tokenizing their text following the steps suggested by the creators of GloVE, and detailed as follows.

3.2.1. Preprocessing data

Prior to training the LSTM on the tweets, we preprocess the data by forming a string of tokens from each tweet.

- We replace occurrences of hashtags, URLs, numbers and user mentions with the tags “ < hashtag > ”, “ < url > ”, “ < number > ”, or ‘ < user > ”.
- Similarly, common emojis are replaced with specific tags (e.g., “ < smile > ”, “ < heart > ”, “ < lolface > ”, “ < neutralface > ” or “ < angryface > ”).
- For words written in upper case letters or for words containing more than 2 repeated letters, a tag denoting that is placed after the occurrence of the word. For example, the word “HAPPY” would be replaced by two tokens, “happy” and “ < allcaps > ”.
- All tokens are converted to lower case.

Then, these tokenized tweets are transformed into an embedding using the aforementioned pre-trained GloVE model. The resulting sequence of vectors is then fed to the LSTM that outputs a single 32-dimension vector that is then fed forward through 2 ReLU activated layers of size 128 and 64 to yield the output. A diagram of this simple LSTM architecture can be seen in Fig. 1. It should be noted that our model resets its state after each input, and therefore it only learns the sequential structure within each tweet and across not the sequence of tweets.

3.2.2. Limits of traditional LSTM deep nets

However, this approach only uses the text content of the tweets, and does not utilize the metadata associated with it. As observed from the results of Table 3, which will be discussed later into detail, although using metadata alone does not suffice to predict whether a user is a bot or not, metadata are weak predictors in that they significantly improve prediction accuracy if paired with other information.

However, metadata are not in the form of sequential data (unlike tweet text) that can be fed directly into the LSTM along with the text embedding. This, therefore, requires an architectural change. We propose a new *Contextual LSTM* architecture that can effectively utilize both the text and the metadata, even though these inputs have two different structures.

3.2.3. Contextual LSTM architecture

Our proposed architecture, depicted in Fig. 2, has multiple inputs and outputs. Multi-input recurrent models have been suggested before, mainly in language models for translation tasks: for example, some authors [22,33] adopted the idea of introducing auxiliary inputs to either the input, hidden, or output layer of the recurrent model to enrich the learned representations. Various types of auxiliary-information have been used, e.g., metadata such as keywords, descriptions, document titles or topic headlines [22], and even preconditioning topics learned via Latent Dirichlet Allocation (LDA) [33].

In our work, we augment the output layer with an auxiliary input. Similar to the previously described tweets-only model, the main input of the proposed model is the tweet text that is tokenized and transformed into a set of GloVE vectors prior

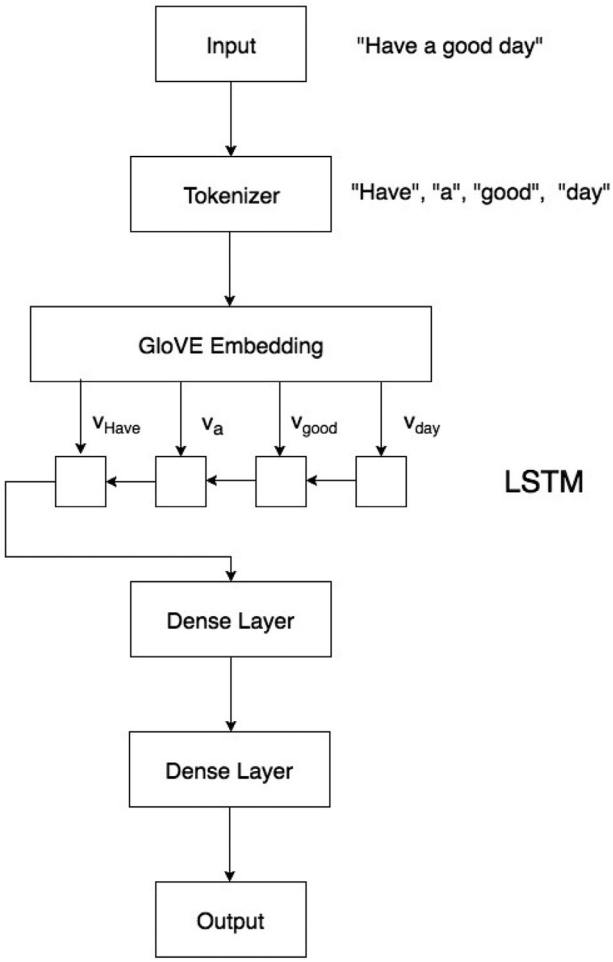

Fig. 1. Architecture of model for tweet-level bot detection that takes only the tweet content as its input.

to feeding them into the LSTM. This again results in an output vector that is concatenated with our auxiliary input and then given as input to a 2-layer neural network with ReLU activations to yield the output. The exact sizes of layers are the same as that of our previous model.

As a regularization mechanism, we introduce an auxiliary output, whose target is also the classification label, alongside with the LSTM output. Such a mechanism has been successfully used before in Inception Nets [44]. The total loss is a weighted average of the auxiliary output loss and the main output loss (0.2:0.8 was found to be optimal via hyper-parameter search).

4. Results

In this section, we summarize and discuss the results yielded by the approaches outlined in the previous section. The success of each method is measured on a variety of metrics, namely Precision, Recall, F1-Score, Accuracy and *Area Under the Receiver Operating Characteristic Curve* (AUC/ROC).

4.1. Task 1: account-level classification

Table 2 shows the results of our experiments on the first prediction task, namely account-level classification.

The first batch of five classifiers shows the results by solely using the data without any oversampling methods: *Random Forest Classifier* and *AdaBoost Classifier* yield the best results, with an AUC $>98\%$. Overall, all classifiers perform well, suggesting that the account-level classification task, at least for the dataset under analysis here, is not particularly daunting.

The second batch of models reported in Table 2 shows the results obtained after using oversampling with SMOTE, followed by undersampling with ENN (SMOTENN): the metrics improve across all classifiers, with AdaBoost achieving near-

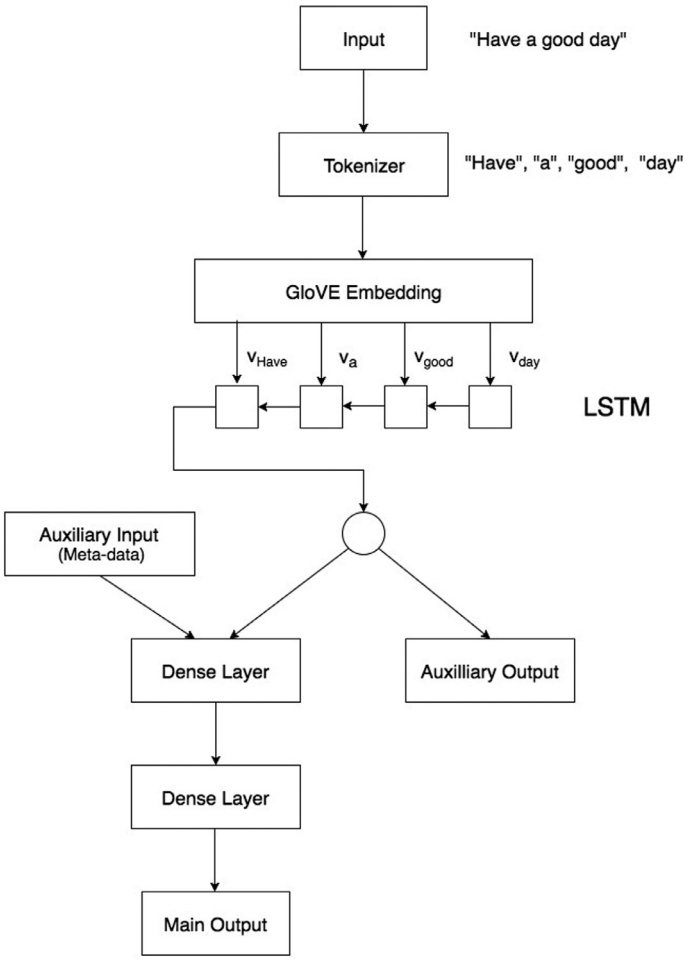

Fig. 2. Architecture of the proposed contextual LSTM that uses both tweet content and the metadata that comes with the tweet.

Table 2
Classification performance of various models on the account-level (user) bot detection task. The first batch of models represent traditional off-the-shelf baseline approaches, that already exhibit very accurate performance. The second and third batches of models are enhanced by means of synthetic minority oversampling techniques, to illustrate how it is possible to achieve nearly perfect account-level bot detection without the need for complex deep architectures. For each batch of models we highlighted the best accuracy and AUC/ROC performing ones: AdaBoost consistently provides the top (or nearly the top) performance across all account-level bot detection benchmarks.

Model	Precision	Recall	F1-Score	Accuracy	AUC/ROC
Logistic Regression	0.94	0.93	0.93	0.9066	0.8891
SGD Classifier	0.87	0.87	0.87	0.8726	0.8680
Random Forest Classifier	0.98	0.98	0.98	**0.9839**	**0.9845**
AdaBoost Classifier	0.98	0.98	0.98	0.9823	0.9823
2-layer NN (500,200,1) RelU + Adam	0.95	0.95	0.95	0.9496	0.9475
Logistic Regression (With SMOTENN)	0.99	0.99	0.99	0.9859	0.9862
SGD Classifier (With SMOTENN)	0.95	0.94	0.94	0.9433	0.9443
Random Forest Classifier (With SMOTENN)	0.99	0.99	0.99	0.9937	0.9938
AdaBoost Classifier (With SMOTENN)	1.00	1.00	1.00	**0.9981**	**0.9981**
2-layer NN (300,200,1) RelU + Adam (With SMOTENN)	0.99	0.99	0.99	0.9878	0.9879
Logistic Regression (With SMOTOMEK)	0.92	0.91	0.91	0.9094	0.9098
SGD Classifier (With SMOTOMEK)	0.90	0.90	0.90	0.9030	0.9031
Random Forest Classifier (With SMOTOMEK)	0.99	0.99	0.99	0.9859	0.9859
AdaBoost Classifier (With SMOTOMEK)	0.99	0.99	0.99	**0.9865**	**0.9865**
2-layer NN (300,200,1) RelU + Adam (With SMOTOMEK)	0.95	0.95	0.95	0.9391	0.9489

Table 3
Classification performance of various models, including the proposed Contextual LSTM, on the tweet-level bot detection task. The first batch of models represent traditional off-the-shelf baseline approaches: their accuracy and AUC/ROC scores range between 71% and 80%. The second and third batches of models are enhanced by means of synthetic minority oversampling techniques, showing better classification performance between 76% and 90% accuracy and AUC/ROC scores. The LSTM architecture that we propose is presented in the fourth batch of models: our results outperform the other approaches by a significant margin, averaging above 96% accuracy and AUC/ROC scores, to demonstrate that tweet-level bot detection can be achieved with extremely high accuracy, small number of features, and limited-size training datasets. For each batch of models we highlighted the best Accuracy and AUC/ROC performing ones. Among the traditional machine learning models (i.e., excluding the proposed approach that significantly outperforms all the baselines and their variants using synthetic minority oversampling), both Random Forest and AdaBoost classifiers appear to consistently deliver the best performance across all tweet-level bot detection benchmarks.

Model	Precision	Recall	F1-Score	Accuracy	AUC/ROC
Logistic Regression (Metadata-only)	0.80	0.80	0.79	0.8008	0.7633
SGD Classifier (Metadata-only)	0.76	0.76	0.75	0.7625	0.7191
Random Forest Classifier (Metadata-only)	0.80	0.80	0.80	**0.8042**	**0.7765**
AdaBoost Classifier (Metadata-only)	0.80	0.80	0.79	0.7991	0.7618
Logistic Regression (Metadata-only + SMOTENN)	0.92	0.92	0.92	0.9188	0.8820
SGD Classifier (Metadata-only + SMOTENN)	0.91	0.90	0.90	0.8992	0.8860
Random Forest Classifier (Metadata-only + SMOTENN)	0.92	0.92	0.92	0.9233	0.8806
AdaBoost Classifier (Metadata-only + SMOTENN)	0.93	0.92	0.93	**0.9234**	**0.9065**
Logistic Regression (Metadata-only+SMOTOMEK)	0.79	0.77	0.76	0.7666	0.7667
SGD Classifier (Metadata-only+SMOTOMEK)	0.78	0.77	0.76	0.7664	0.7664
Random Forest Classifier (Metadata-only+SMOTOMEK)	0.79	0.77	0.77	**0.7747**	**0.7748**
AdaBoost Classifier (Metadata-only+SMOTOMEK)	0.79	0.77	0.77	0.7715	0.7716
LSTM (Tweet-only + 50D GloVE)	0.96	0.96	0.96	0.9553	0.9567
Contextual LSTM (25D GloVE)	0.96	0.96	0.96	0.9567	0.9585
Contextual LSTM (50D GloVE)	0.96	0.96	0.96	0.9618	0.9627
Contextual LSTM (100D GloVE)	0.96	0.96	0.96	0.9618	0.9626
Contextual LSTM (200D GloVE)	0.96	0.96	0.96	**0.9633**	**0.9643**

perfect accuracy and AUC of 99.81%. This suggests that our strategy of synthetic minority oversampling is really effective when dealing with this type of account-level classification tasks.

The third batch of models shows the results of oversampling with SMOTE followed by undersampling with TOMEK (SMOTOMEK): with this approach the results improve as well, though not as systematically as by using ENN.

Overall, we demonstrated the feasibility of high-accuracy account-level bot detection by means of unsophisticated off-the-shelf algorithms in combination with synthetic minority oversampling techniques employed to enhance training data.

4.2. Task 2: tweet-level classification

We now illustrate the proposed tweet-level bot detection task. Table 3 shows four batches of models: the first three show the baseline models provided by the off-the-shelf classifiers, specifically in their naive implementation (first batch), and by using again synthetic minority oversampling in combination with ENN (second batch) or TOMEK (third batch); finally, the fourth and last batch shows the performance of the proposed LSTM deep architectures, showing different configurations.

The first batch of models that only uses the tweet metadata does not work particularly well. Without data augmentation, none of the results cross an AUC of 78%.

By means of synthetic minority oversampling, in combination with ENN, results drastically improve: the second batch of models shows that all models approach an accuracy between 88% and 90%.

However, no such improvement is seen by using SMOTE in combination with TOMEK. Although the reasons of such a difference with respect to the previous task are not obvious, and warrant further investigation, we hypothesized that ENN works better because tweets originated by bots exhibit identical or very similar metadata thus a nearest-neighbor strategy works best.

The fourth batch of results presents our architecture and deserves an in-depth discussion. By using only the text of the tweets, our LSTM model provides a classification accuracy of 95.53%, see "LSTM (Tweets-only + 50D GloVE)". This result is very promising: the use of our proposed deep net architecture yields a lift in performance in the order of 5%, even using only the tweet texts.

Following the intuition that metadata can enrich the information available for classification purposes, the last four models show the results of our *Contextual LSTM* combining metadata and tweet text features. Our Contextual LSTM shows a boost in performance yielding a classification accuracy of about 96.33% for the best model with 200-dimension GloVE embedding, see "Contextual LSTM (200D GloVE)". It is worth noting that different configurations of GloVE, obtained by varying the dimensionality of the word embedding space, do not exhibit drastically different performance: the evidence seems to suggest that higher dimensionality yields increasingly slightly-better performance. In conclusion, from our analysis we derive that, even though metadata is just a weak predictor of the account's nature, as seen in the baseline results, our proposed

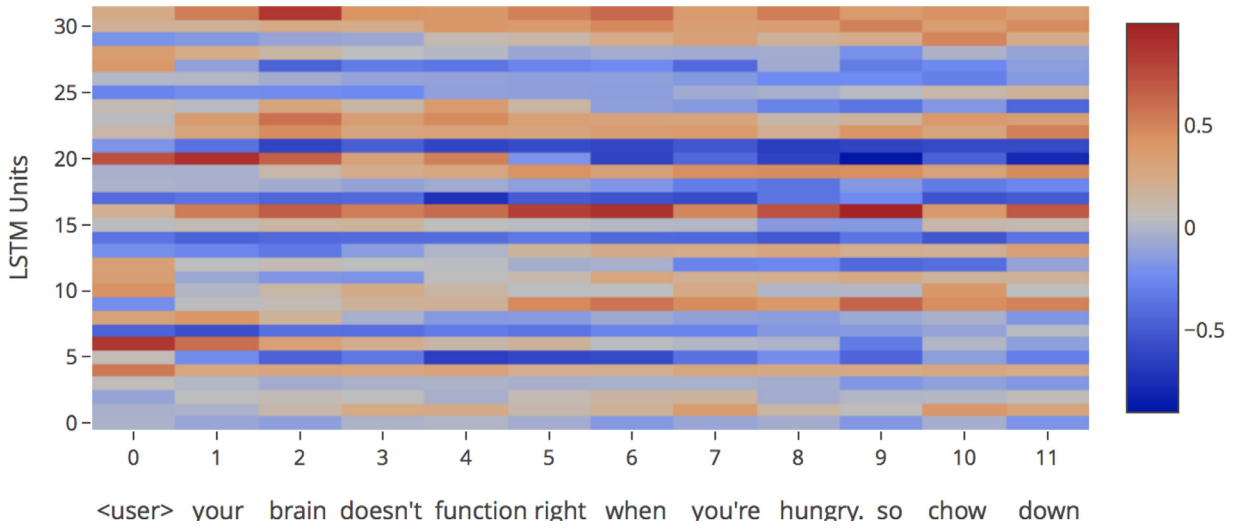

Fig. 3. Representations over time of LSTM 32 hidden units activations for a tweet generated by a human. Each column corresponds to LSTM outputs at each time step. Cells correspond to the 32 dimensions of the representation at each timestep.

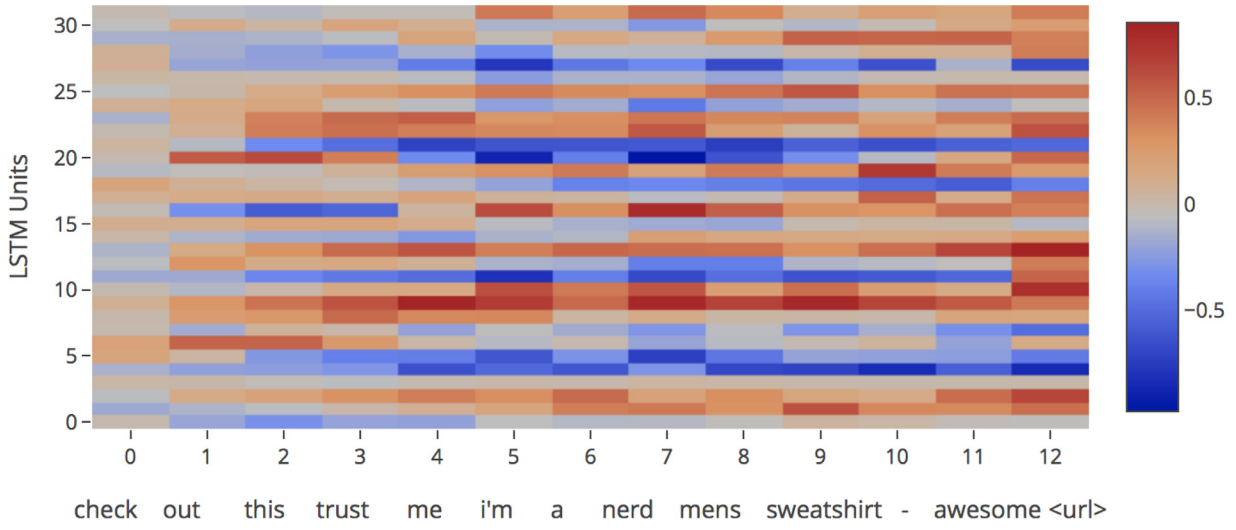

Fig. 4. Representations over time of LSTM 32 hidden units activations for a tweet generated by a bot. Each column corresponds to LSTM outputs at each time step. Cells correspond to the 32 dimensions of the representation at each timestep.

architecture uses this extra information to yield more accurate prediction results. It is worth noting that metadata have not been oversampled with the Contextual LSTM models.

4.3. Interpretation of LSTM models

Deep neural network models are often touted as uninterpretable black boxes. Although they provide extraordinary results in many practical applications, various researchers, practitioners, policy makers, and funding agencies demand attention to the matter of AI's interpretability [39]. Recently, several studies aimed at shedding light on the inner states of recurrent models by visualizing their hidden state dynamics. Karpathy and collaborators, for example, examined inner states of language models [27], while Li et al. [30] introduced strategies to help interpreting deep neural network results in NLP tasks. With this in mind, we here attempt to understand and interpret the effectiveness of our methods by analyzing the hidden state activations of our neural network model.

We first visualize the change in the representation, i.e., the output over time of the LSTM hidden units. We give examples of tweets generated by a human and by a bot in Figs. 3 and 4 respectively. We observe significant differences in the activations between the examples, suggesting that the LSTM hidden units may correspond to different linguistic features.

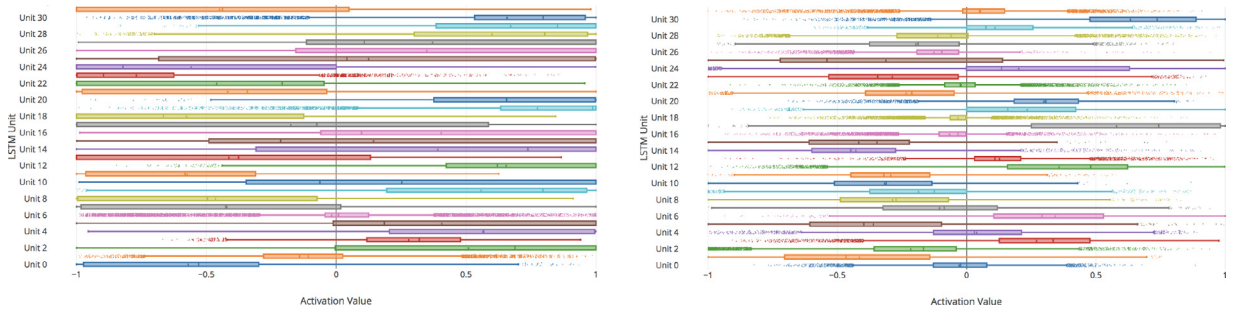

Fig. 5. Distribution of activation values of 32 LSTM hidden units on tweets generated by humans (left) and bots (right).

It has been observed by Cresci et al. [12] that using simple textual features reveals ineffective against the more sophisticated social bots we study in this work. We plot the distributions of the final LSTM representations of genuine tweets and bot-generated tweets in Fig. 5, left and right respectively. We see that the majority of hidden units have a significant difference in the distribution of activation values for genuine tweets versus to bot-generated tweets. For example, Unit 0 (bottom blue box) has a broad activation range centered around -0.6 for tweets generated by humans, as opposed to a narrow activation range centered around 0 for tweets generated by bots. Similarly, Unit 31 (top orange box) is activated in the negative $[-1,0]$ range for genuine tweets, but again narrowly around 0 for bot-generated tweets. In general, it appears that the activation ranges are much broader for genuine tweets, and much narrower for bot-generated ones, maybe suggesting more heterogeneity and nuances in the signature of human language. These examples suggest the mechanism behind the ability of LSTM to learn more complex, non-linear features, which are effective in identifying bot-generated tweets even from single observations.

4.4. Considerations and limitations

We developed bot detection models requiring a minimal number of features as well as one single observation of a tweet generated by the account to be scrutinized. For the account-level bot detection, we showed that using oversampling and data enhancement techniques could yield models that perform nearly perfectly.

Our main contribution, however, is presenting to the best of our knowledge the first bot detection model that detects the nature of an account from a single tweet. First, we find that simply using the tweet metadata does not suffice. We also show that, while simply using the content of the tweet works quite well with our proposed LSTM architecture, we can improve the performance by effectively utilizing the information available in the metadata.

5. Conclusions

Given the prevalence of sophisticated bots on social media platforms such as Twitter, the need for improved, inexpensive bot detection methods is apparent. We proposed a novel Contextual LSTM architecture allowing us to use both tweet content and metadata to detect bots at the tweet level. From a single tweet, our model can achieve an extremely high accuracy exceeding 96% AUC.

We show that the additional metadata information, though a weak predictor of the nature of a Twitter account per se, when exploited by LSTM decreases the error rate by nearly 20%. In addition to this, we propose methods based on synthetic minority oversampling that yield a near perfect account-level detection accuracy ($>99\%$ AUC). Both these methods use a very minimal number of features that can be obtained in a straightforward way from the tweet itself and its metadata, while surpassing prior state of the art.

In the future, we plan to make our framework open source, and to implement a Web service (for example, an API) to allow the research community to perform tweet-level bot detection using it.

From a research standpoint, we plan to use the proposed framework to scrutinize social media conversation in different contexts, in order to determine the extent of the interference of bots with public discourse, as well as to understand how their capabilities and sophistication evolve over time.

Acknowledgements

The authors gratefully acknowledge support by the Air Force Office of Scientific Research (AFOSR #FA9550-17-1-0327), and by the Defense Advanced Research Projects Agency (DARPA #W911NF-17-C-0094). The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright annotation thereon. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of AFOSR, DARPA, or the U.S. Government.

References

[1] N. Abokhodair, D. Yoo, D.W. McDonald, Dissecting a social botnet: growth, content and influence in twitter, in: Proc. of the 18th ACM Conf. on Computer Supported Cooperative Work & Social Computing, ACM, 2015, pp. 839–851.
[2] J.-P. Allem, E. Ferrara, The importance of debiasing social media data to better understand e-cigarette-related attitudes and behaviors, J. Med. Internet Res. 18 (8) (2016).
[3] J.-P. Allem, E. Ferrara, S.P. Uppu, T.B. Cruz, J.B. Unger, E-Cigarette surveillance with social media data: social bots, emerging topics, and trends, JMIR Public Health Surveill. 3 (4) (2017).
[4] A. Badawy, E. Ferrara, The rise of jihadist propaganda on social networks, J. Comput. Social Sci. (2018).
[5] A. Badawy, E. Ferrara, K. Lerman, Analyzing the digital traces of political manipulation: the 2016 Russian interference twitter campaign, arXiv:1802.04291 (2018).
[6] G.E. Batista, R.C. Prati, M.C. Monard, A study of the behavior of several methods for balancing machine learning training data, ACM Sigkdd Explor. Newslett. 6 (1) (2004) 20–29.
[7] A. Bessi, E. Ferrara, Social bots distort the 2016 us presidential election online discussion, First Monday 21 (11) (2016).
[8] Q. Cao, M. Sirivianos, X. Yang, T. Pregueiro, Aiding the detection of fake accounts in large scale social online services, in: 9th USENIX Symp on Netw Sys Design & Implement, 2012, pp. 197–210.
[9] N.V. Chawla, K.W. Bowyer, L.O. Hall, W.P. Kegelmeyer, Smote: synthetic minority over-sampling technique, J. Artif. Intell. Res. 16 (2002) 321–357.
[10] Z. Chu, S. Gianvecchio, H. Wang, S. Jajodia, Detecting automation of twitter accounts: are you a human, bot, or cyborg? IEEE Trans. Depend. Secure Comput. 9 (6) (2012) 811–824.
[11] E. Clark, J. Williams, C. Jones, R. Galbraith, C. Danforth, P. Dodds, Sifting robotic from organic text: a natural language approach for detecting automation on twitter, J. Comput. Sci. 16 (2016) 1–7.
[12] S. Cresci, R. Di Pietro, M. Petrocchi, A. Spognardi, M. Tesconi, The paradigm-shift of social spambots: evidence, theories, and tools for the arms race, in: Proceedings of the 26th International Conference on World Wide Web Companion, International World Wide Web Conferences Steering Committee, 2017, pp. 963–972.
[13] C.A. Davis, O. Varol, E. Ferrara, A. Flammini, F. Menczer, Botornot: a system to evaluate social bots, in: Proceedings of the 25th International Conference Companion on World Wide Web, International World Wide Web Conferences Steering Committee, 2016, pp. 273–274.
[14] E. Ferrara, Manipulation and abuse on social media, ACM SIGWEB Newslett. (Spring) (2015) 4.
[15] E. Ferrara, Contagion dynamics of extremist propaganda in social networks, Inf. Sci. 418 (2017) 1–12.
[16] E. Ferrara, Disinformation and social bot operations in the run up to the 2017 french presidential election, First Monday 22 (8) (2017).
[17] E. Ferrara, O. Varol, C. Davis, F. Menczer, A. Flammini, The rise of social bots, Commun ACM 59 (7) (2016) 96–104.
[18] E. Ferrara, W.-Q. Wang, O. Varol, A. Flammini, A. Galstyan, Predicting online extremism, content adopters, and interaction reciprocity, in: International conference on social informatics, Springer, 2016, pp. 22–39.
[19] D. Gayo-Avello, Social media won't free us, IEEE Internet Comput. 21 (4) (2017) 98–101.
[20] Y. Goldberg, A primer on neural network models for natural language processing., J. Artif. Intell. Res.(JAIR) 57 (2016) 345–420.
[21] H. He, E.A. Garcia, Learning from imbalanced data, IEEE Trans. Knowl. Data Eng. 21 (9) (2009) 1263–1284.
[22] C.D.V. Hoang, T. Cohn, G. Haffari, Incorporating side information into recurrent neural network language models., in: HLT-NAACL, 2016, pp. 1250–1255.
[23] S. Hochreiter, J. Schmidhuber, Long short-term memory, Neural Comput. 9 (8) (1997) 1735–1780.
[24] P.N. Howard, B. Kollanyi, Bots,# strongerin, and# brexit: computational propaganda during the uk-eu referendum, Browser Download This Paper(2016).
[25] V. John, A survey of neural network techniques for feature extraction from text, arXiv:1704.08531 (2017).
[26] R. Jozefowicz, W. Zaremba, I. Sutskever, An empirical exploration of recurrent network architectures, in: Proceedings of the 32nd International Conference on Machine Learning (ICML-15), 2015, pp. 2342–2350.
[27] A. Karpathy, J. Johnson, L. Fei-Fei, Visualizing and understanding recurrent networks, arXiv:1506.02078 (2015).
[28] A. Krizhevsky, I. Sutskever, G.E. Hinton, Imagenet classification with deep convolutional neural networks, in: Advances in Neural Information Processing Systems, 2012, pp. 1097–1105.
[29] Y. LeCun, Y. Bengio, G. Hinton, Deep learning, Nature 521 (7553) (2015) 436–444.
[30] J. Li, X. Chen, E. Hovy, D. Jurafsky, Visualizing and understanding neural models in nlp, arXiv:1506.01066 (2015).
[31] B.D. Loader, D. Mercea, Networking democracy? Social media innovations and participatory politics, Inf., Commun. Soc. 14 (6) (2011) 757–769.
[32] P.T. Metaxas, E. Mustafaraj, Social media and the elections, Science 338 (6106) (2012) 472–473.
[33] T. Mikolov, G. Zweig, Context dependent recurrent neural network language model., SLT 12 (2012) 234–239.
[34] S. Mitter, C. Wagner, M. Strohmaier, A categorization scheme for socialbot attacks in online social networks, in: Proc. of the 3rd ACM Web Science Conference, 2013.
[35] V. Mnih, K. Kavukcuoglu, D. Silver, A.A. Rusu, J. Veness, M.G. Bellemare, A. Graves, M. Riedmiller, A.K. Fidjeland, G. Ostrovski, et al., Human-level control through deep reinforcement learning, Nature 518 (7540) (2015) 529–533.
[36] B. Mønsted, P. Sapieżyński, E. Ferrara, S. Lehmann, Evidence of complex contagion of information in social media: an experiment using twitter bots, PLoS ONE 12 (9) (2017).
[37] J. Pennington, R. Socher, C.D. Manning, Glove: global vectors for word representation, in: Empirical Methods in Natural Language Processing (EMNLP), 2014, pp. 1532–1543.
[38] I. Pozzana, E. Ferrara, Measuring bot and human behavioral dynamics, arXiv:1802.04286 (2018).
[39] W. Samek, T. Wiegand, K.-R. Müller, Explainable artificial intelligence: understanding, visualizing and interpreting deep learning models, arXiv:1708.08296 (2017).
[40] D. Silver, A. Huang, C.J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, et al., Mastering the game of go with deep neural networks and tree search, Nature 529 (7587) (2016) 484–489.
[41] T. Stein, E. Chen, K. Mangla, Facebook immune system, in: Proc. of the 4th Workshop on Social Network Systems, ACM, 2011, p. 8.
[42] M. Stella, E. Ferrara, M. De Domenico, Bots sustain and inflate striking opposition in online social systems, arXiv:1802.07292 (2018).
[43] V. Subrahmanian, A. Azaria, S. Durst, V. Kagan, A. Galstyan, K. Lerman, L. Zhu, E. Ferrara, A. Flammini, F. Menczer, The darpa twitter bot challenge, Computer 49 (6) (2016) 38–46.
[44] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, A. Rabinovich, Going deeper with convolutions, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2015, pp. 1–9.
[45] O. Varol, E. Ferrara, C. Davis, F. Menczer, A. Flammini, Online human-bot interactions: Detection, estimation, and characterization, in: International AAAI Conference on Web and Social Media, 2017, pp. 280–289.
[46] O. Varol, E. Ferrara, F. Menczer, A. Flammini, Early detection of promoted campaigns on social media, EPJ Data Sci. 6 (1) (2017) 13.
[47] C. Wagner, S. Mitter, C. Körner, M. Strohmaier, When social bots attack: modeling susceptibility of users in online social networks, Making Sense Microposts (# MSM2012) 2 (4) (2012) 1951–1959.
[48] G. Wang, M. Mohanlal, C. Wilson, X. Wang, M. Metzger, H. Zheng, B.Y. Zhao, Social turing tests: crowdsourcing sybil detection, in: Proc. of the 20th Network & Distributed System Security Symposium (NDSS), 2013.
[49] S.C. Woolley, Automating power: social bot interference in global politics, First Monday 21 (4) (2016).
[50] Z. Yang, C. Wilson, X. Wang, T. Gao, B.Y. Zhao, Y. Dai, Uncovering social network sybils in the wild, ACM Trans. Knowl. Discovery Data 8 (1) (2014) 2.