Social Network Analysis and Mining (2022) 12:4
https://doi.org/10.1007/s13278-021-00800-9

ORIGINAL ARTICLE

GANBOT: a GAN-based framework for social bot detection

Shaghayegh Najari[1] · Mostafa Salehi[1,2] · Reza Farahbakhsh[3]

Received: 28 March 2021 / Revised: 9 September 2021 / Accepted: 14 September 2021

Abstract
Nowadays, a massive number of people are involved in various social media. This fact enables organizations and institutions to more easily access their audiences across the globe. Some of them use *social bots* as an automatic entity to gain intangible access and influence on their users by faster content propagation. Thereby, malicious social bots are populating more and more to fool humans with their unrealistic behavior and content. Hence, that's necessary to distinguish these fake social accounts from real ones. Multiple approaches have been investigated in the literature to answer this problem. Statistical machine learning methods are one of them focusing on handcrafted features to represent characteristics of social bots. Although they reached successful results in some cases, they relied on the bot's behavior and failed in the behavioral change patterns of bots. On the other hands, more advanced deep neural network-based methods aim to overcome this limitation. *Generative adversarial network (GAN)* as new technology from this domain is a semi-supervised method that demonstrates to extract the behavioral pattern of the data. In this work, we use GAN to leak more information of bot samples for state-of-the-art textual bot detection method (*Contextual LSTM*). Although GAN augments low labeled data, original textual GAN (*Sequence Generative Adversarial Net (SeqGAN)*) has the known limitation of convergence. In this paper, we invested this limitation and customized the GAN idea in a new framework called *GANBOT*, in which the generator and classifier connect by an LSTM layer as a shared channel between them. Our experimental results on a bench-marked dataset of Twitter social bot show our proposed framework outperforms the existing contextual LSTM method by increasing bot detection probabilities.

Keywords Social bot detection · Deep neural networks · Text classification · Generative adversarial networks

1 Introduction

In the past few years, social media ecosystems changed considerably by a large number of automatic entities that try to mimic human-like behavior to produce content that these entities are known as social bots (Ferrara et al. 2016). In an accurate definition, a social bot is a computer-based algorithm that automatically controls a social media account, produces content, and potentially interacts with human users on social media trying to emulate human behavior.

The social bots activities are increasing on various *Online Social Networks (OSNs)*. For example, Twitter social bots had a grave effect on the 2016 U.S. presidential election, and 50% of posts about Trump as candidates were written by automated accounts (Howard et al. 2016). Along with several positive usages of this phenomenon, such as automatically posting usefulness and emergency information, there is an increasing concern about their destructive tasks. They can influence and bias users' opinion via spreading malicious manipulated contents (Ferrara et al. 2016; Halawa et al. 2019) such as spam or fake review contents (Cresci et al. 2017; Aghakhani et al. 2018) on social media. For example, they are able to make some political, economical, and healthcare attacks to direct human's mind and activities toward special goals (Bodaghi et al. 2019). As they spread conspiracy tweets on Twitter along with spreading

✉ Mostafa Salehi
mostafa_salehi@ut.ac.ir

Shaghayegh Najari
najari.shaghayegh@ut.ac.ir

Reza Farahbakhsh
reza.farahbakhsh@it-sudparis.eu

[1] Faculty of New Sciences and Technologies, University of Tehran, Tehran, Iran

[2] School of Computer Science, Institute for Research in Fundamental Science (IPM), P.O.Box 19395-5746, Tehran, Iran

[3] Institut Polytechnique de Paris, Telecom SudParis, Evry, France

Published online: 14 November 2021

COVID-19 pandemic (Ferrara 2020; Antenore et al. 2021, 2021), or their considerable role in 2016 US elections (Bessi and Ferrara 2016; Deb et al. 2019; Ferrara 2017; Stella et al. 2018) where they were employed in democratic conversation about political candidates.

As these malicious bots have staked the reputation and power of OSNs are proposed some studies to detect them and illustrate whether a human or a bot is controlling a social account. Some of the traditional proposed solutions relied on statistical *machine learning (ML)* methods (Hochreiter and Schmidhuber 1997) that required some substantial and fixed handcrafted features. Therefore, they are not generalized and usable for all datasets. Recent studies show *Deep Neural Networks (DNN)* are able to fetch some useful features from pure data automatically during their training phase and achieved some successful results rather than statistical ML methods. (More details is described in next Sect. 2.) However, bot detection is more difficult than ever because they are also relying on advanced techniques.

GANs are a new technology of DNN that firstly was proposed for image generation and recently is customized for textual tasks and reached promising results for text generation (Yu et al. 2017; Tuan and Lee 2019; Fedus et al. 2018) that able bots to be more complex. On the other hands, there are some solutions based on GAN for detection tasks such as fake news detection (Aghakhani et al. 2018), spam detection (Stanton and Irissappane 2019), fraud review detection (Shehnepoor et al. 2020) and even social bot detection (Bin et al. 2020). However, all of them augment the samples of the small class by using traditional SeqGAN that suffering the convergence limitation. On the other words, GANs have been using to extract the distribution pattern of samples, and we try to utilize this ability to follow up the behavioral pattern of social bots. However, GAN has some limitations that make it hard to use for social bot detection; in this work, we try to fill out this gap in a new GAN-based framework as GANBOT to eliminate this barrier by using a common LSTM layer as a shared channel between generator and classifier.

As the textual content of social bots can be accessed easily in bulk, they are popular in some tasks (Kudugunta and Ferrara 2018), we also focused on detecting social bots that generate text.

Our experimental results show proposed method could outperform contextual LSTM as state-of-the-art text-based bot detection method (Kudugunta and Ferrara 2018). The main contributions of this study can summarize as follow:

- We customize the GAN approach for text-based bot detection as a new framework named *GANBOT* to leak more information of behavioral patterns of bot samples.
- We eliminate convergence limitation of traditional SeqGAN in GANBOT framework through sharing an LSTM layer between generator and classifier.
- The proposed framework increases true label probabilities as well as decreases false alarm ones (Figs. 5, 6).

The rest of this paper is organizing as follows. Section 2 presents a brief review of recent studies on social bot detection and the concept of the GAN framework. In Sect. 3 is presented our proposed method and some evaluation of the original SeqGAN. Our experimental results and findings present in Sect. 4; finally, Sect. 5 is the conclusion.

2 Literature review

There are many studies for bot detection ranging from statistical machine learning methods to deep learning-based techniques. Recently, social bots are interactive and use generative models for text generation. We categorize the recent studies in this section.

2.1 Bot detection

Previous studies proposed various solutions for bot detection, some of them try to discover and use more useful features in *statistical machine learning (ML)* methods. On the other hands, there are other approaches based on *deep learning (DL)* ones that are able to extract the most precise features automatically during their training phase. In a more precise definition, ML uses a set of algorithms to interpret and learn data and make the best possible decisions based on them, while DL is a subset of ML that use multiple layers in order to create an artificial neural network, this neural network can learn from the data and also make intelligent decisions.

There are some reviews that introduce various taxonomies (Ferrara et al. 2016; Orabi et al. 2020) based on different aspects. We categorize them as ML-based and DL-based methods that is summarized in Table 1.

2.1.1 Machine learning-based methods

These methods learn to identify bots based on a set of input features, by training them on a set of datasets. Thus, the performance of these models is dependent on the value of used features.

Different classes of features are commonly employed to capture users' behaviors that can categorize them into two main categories as *Content-based* and *User-based* features. Content-based features are extracting from post's contents; some works use an embedding layer to translate raw texts into corresponding vectors by using *linguistic* features such

Table 1 Review on methods employed for bot detection problem including two categories: (1) statistical machine learning as ML-based methods and (2) deep learning as DL-based methods

Method	Features						Year	References
	User-based			Content-based				
	Metadata	Network	Temporal	Embedding	Linguistic	Semantic		
ML-based					✓		2016	Igawa et al. (2016)
	✓	✓	✓		✓	✓	2016	Davis et al. (2016)
	✓		✓				2017	Gilani et al. (2017)
	✓						2018	Chen and Subramanian (2018)
				✓			2018	Jr et al. (2018)
						✓	2018	Wang et al. (2018).
		✓	✓				2018	Dorri and Dadfarnia (2018)
		✓	✓	✓			2019	Abu-El-Rub and Mueen (2019)
			✓				2019	Luceri et al. (2019)
	✓	✓					2019	Hurtado et al. (2019)
	✓						2020	Rodríguez-Ruiz et al. (2020)
		✓					2021	Bebensee et al. (2021)
DL-based			✓	✓			2017	Cai et al. (2017b)
			✓	✓		✓	2017	Cai et al. (2017a)
	✓		✓	✓			2018	Ping and Qin (2018)
	✓		✓	✓			2018	Chavoshi and Mueen (2018)
	✓			✓			2018	Kudugunta and Ferrara (2018)
	✓			✓	✓		2019	Daouadi et al. (2019)
	✓	✓	✓	✓		✓	2019	Beskow and Carley (2019)
	✓		✓	✓			2019	Mazza et al. (2019)
	✓						2020	Yang et al. (2020)
	✓	✓					2020	Zhao et al. (2020)
	✓	✓	✓	✓			2020	Bin et al. (2020)

as *Lexicon-based Coefficient Attenuation* (Igawa et al. 2016), *part-of-speech tagging* (Barbon et al. 2018) or tweets similarity features (Chen and Subramanian 2018) are packed in this category. On the other hands, latent features such as *sentimental* ones can be extracted through *semantic* algorithms (Wang et al. 2018). Beside content-based features, there are some other features based on *user's profile, behavior and network's features.* Profile features also are known as *metadata* including language, age, gender, geographic locations and account creation time. Behavioral features are based on period or time of posting contents (Velayutham and Tiwari 2017; Gilani et al. 2017) and network-based features (Hurtado et al. 2019; Shehnepoor et al. 2017) are such as account popularity (Gilani et al. 2017), *clustering coefficient*, *community properties* (Abu-El-Rub and Mueen 2019; Ranjbar et al. 2018) and *homophily properties* (Dorri and Dadfarnia 2018).

There are many statistical machines to learn these features automatically as bot detection methods, such as *random forest* (Varol et al. 2017; Gilani et al. 2017), *AdaBoost* (Abu-El-Rub and Mueen 2019; Andriotis and Takasu 2018), *K-nearest neighbor (KNN)* (Valliyammai and Devakunchari 2019) and *support vector machine (SVM)* (Dorri and Dadfarnia 2018).

2.1.2 Deep learning-based models

This way can feed both user-based or content-based features into a deep neural network and extract better features.

Long short-term memory (LSTM) and *convolutional neural networks (CNN)* are the most popular neural networks that achieved promising results in various domains including bot detection challenge (Kudugunta and Ferrara 2018; Zhao et al. 2020; Ping and Qin 2018; Beskow and Carley 2019; Daouadi et al. 2019; Orabi et al. 2020). In most of these methods, raw text representation is used in the corresponding vector by transfer learning from pre-trained linguistic models as *Global Vectors Word Representation (GloVE)* (Pennington et al. 2014). Recently, on the same use case, Wu et al. leveraged GANs as a semi-supervised task to improve bot detection task via data augmentation (Bin et al. 2020).

2.2 Bot generation

Bots have been existing since the early age of Internet in different shapes ranging from partially controlled such as spam generators (Ferrara 2017), to fully automated algorithms such as chatbots which are initially designed to hold a conversation with a human, as envisioned by Alan Turing in the 1950s (TURING 1950). In this era, recent advancements of natural language processing from simple neural networks to transfer learning methods enable bots to interact with real users so that they cannot find if a text is generated by a bot or human (Alarifi et al. 2016). (For a recent survey of Text Generation tools see Iqbal and Qureshi (2020)).

GAN is one of the most popular generative models that shows considerable advancement in generation tasks, at the first, GAN was proposed for continuous data (image generation) (Goodfellow et al. 2014), but soon it is extended to discrete and textual data (Yu et al. 2017). Due to the discrete nature of text samples, text generation is modeled as a *reinforcement learning (RL)* process, where the state is previously generated tokens (words), the action is the next token to be generated, and the generator net is a stochastic policy that maps current state to a distribution over the action space (Yu et al. 2017). After the whole text is generated, the generated text samples are fed to the discriminator network, a classifier that is trained to distinguish generated text samples from realistic ones, to get reward signals for updating generator network (Goodfellow et al. 2014). Generator and discriminator play a minimax game, and each of them tries to maximize its gain and in contrast and minimize its adversary gain.

Formally in this game, first of all the generator model G_θ parameterized with θ pre-train over real-world sentences to produce a sequence of tokens as $S_{1:T} = (S_1, \dots, S_t, \dots, S_T)$, $S_t \in S$, where S is the set of vocabulary of sentences. The objective of generative model is to find an approximated distribution of real train set data conditioned on generator parameters as $G_\theta(S_{1:T}|\theta)$. The generator produces each token of samples based on preceding ones as shown in Eq. 1.

$$G_\theta(S_{1:T}|\theta) = \prod_{t:1,2,\dots,T} G_\theta(S_t|S_{1:t-1},\theta) \tag{1}$$

During pre-training step, each of neural networks tries to minimize cross-entropy over train set examples. The cross-entropy between two distributions p, q over a given support set $X = x_1, \dots, x_n$ is shown in Eq. 2.

$$CrossEntropy_{pq} = -\sum_{X=x_1,x_2,\dots,x_n} p(x_i)\log q(x_i) = -E_p[log(q(x))] \tag{2}$$

This equation can formulate using the Kullback–Leibler divergence from p of q, $D_{KL}(p||q)$, as the relative entropy of q relative to$p$.

Specifically, the generator G_θ try to maximize the likelihood formulated on Eq. 1, this formula is the same as minimizing the cross-entropy by updating weights through formulated loss in Eq. 3 on a sequence of tokens ($S_{1:T}$) extracted one by one from train sets sentences.

$$J^{G_\theta} = -\sum_{t:1,2,\dots,T} \log G_\theta(S_t|S_{1:t-1},\theta) \tag{3}$$

Once the generator pre-trained, we can start pretraining the discriminator as follows:

$$J^D = -E_{S_{1:T}\sim p_{data}} \log D(S_{1:T}) - E_{S_{1:T}\sim P_{G_\theta}} \log(1 - D(S_{1:T})) \tag{4}$$

The discriminator gets a batch of negative samples generated by the generator model and trains to discriminate them from positive realistic samples $S_{1:T}$ with prior distribution P_{data} via the loss function formulated in Eq. 6. Consequently, the discriminator output identifies how likely their input sentence is realistic or synthetic (drawn from the generator model). As the text generation task is a sequential task, we can use recurrent neural networks (RNN) such as long short-term memory (LSTM) or gated recurrent unit (GRU) for whether generator or discriminator.

In adversarial training phase, the generator produces sentences token by token; therefore, it can model as a reinforcement learning (RL) algorithm to produce best action in each time step to maximize the expected rewards of policy gradients derived from discriminator as a binary feedback for generated samples (Yu et al. 2017).

Since the discriminator backward reward for finished sequence while the generator model generates sentences token by token, on the other hands, sending the finished sentence into discriminator lead to miss intermediate information, to solve this issue traditional idea is using Mont Carlo tree search (MCTS) with Roll out Policy to sample reminded tokens (Yu et al. 2017) and complete sentence. In fact, the discriminator as an action function sends feedback into the generator for each produced action, in where the current state is produced tokens $S_{1:t}$ and action is selecting the next token so that maximize the averaged reward of the discriminator.

Generally, there are some recent methods as StepGAN (Tuan and Lee 2019) and MaskGAN (Fedus et al. 2018) that try to improve SeqGAN using the actor-critic method to optimize the policy. However, they have other problems; for example, MaskGAN does not support sentences longer than length 40.

Formally speaking, in a GAN-based framework, the generator and discriminator can play a game with each other in a three different games as minimax or non-saturating or maximum likelihood estimation (MLE) (Goodfellow et al. 2014); in each of them discriminator task is to predict how likely a sample is generated by generator or sampled from train set (6), but the generator loss based on game type is the difference, in a minimax game, the generator loss flip a sing behind the discriminator loss as $J^G = -J^D$. In this loss function, the generator has the problem of vanishing gradient and defines another loss function for maximum likelihood estimation as Eq. 5.

$$J^{G_\theta}_{MLE} = -E_X \exp \sigma^{-1}(D_\beta(G_\theta(X))) \tag{5}$$

all above formula borrowed from two main references as Goodfellow et al. (2014); Yu et al. (2017).

3 GAN-based bot detection

In this section, we first present the traditional seqGAN (Yu et al. 2017) framework to plug in bot text samples. This method as shown in previous studies (Shehnepoor et al. 2020; Guo et al. 2018; Bin et al. 2020) suffering the convergence limitations, we show this problem in our bot detection task. As this task is classification and not generation, we can use this idea in a way other than data augmentation. We try to obviate the convergence limitation of this original framework in a new framework called *GANBOT*. The contextual LSTM as same as shown discriminator in Fig. 5a is the state-of-the-art bot detection method (Kudugunta and Ferrara 2018) that we try to improve that.

3.1 Original Text-GAN: SeqGAN (Yu et al. 2017)

In this section, we try traditional SeqGAN framework to generate more samples of class bot and investigate the limitation of this task.

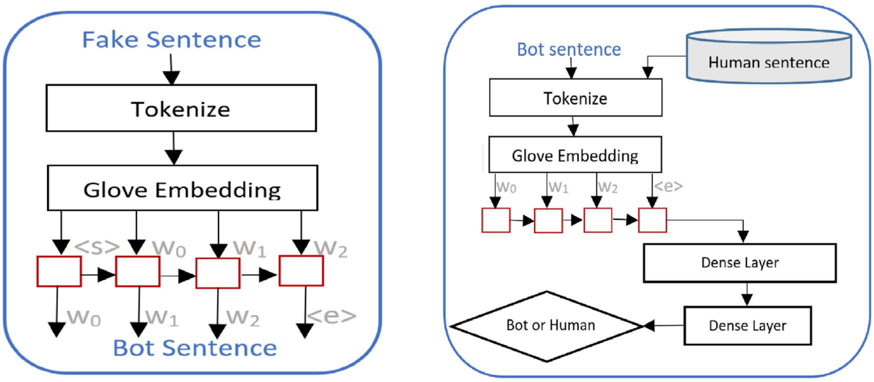
In fact, a text-gan framework is including two neural networks as generator and discriminator to compete or cooperate with each other for producing more accurate samples, as our input data are textual we use LSTM network as a sequential model for generator and discriminator as shown in Fig. 1. In left side sub-Figure 3a, the generator aims to mimic the distribution of real text from a collection of writings by bots available in train set then produce a new sentence based of their knowledge using stochastic policy agent in reinforcement learning. Also, in right-side sub-figure 3d, the discriminator has a dense layer on the output of last LSTM cell to classify input sentence based on last hidden state, in this game, generator for it's generated samples receive signaling feedback based on discriminator output as reward (Eq. 5).

As our goal is text classification, we compare the backwarded reward of the discriminator to the generator on SeqGAN framework (Eq. 6) as a binary feedback signal with derived loss of discriminator during its training step (Eq. 2). Our experimental results show the derived reward from binary feedback signals is very lower than discriminator loss values that are led to vanishing gradient phenomena in generator back-warding, as shown in Fig. 2 that vertical axis shows loss values and horizontal ones show these values for training batches with the size 64.

This plot shows, the spars guiding signals of discriminator are not informative for generator sufficiently and are required more information, while in this framework, discriminator cannot leak more information of generator performance for that. On the other hands, this non-informative binary feedback requires a huge number of training steps and generated samples to improve the generator and even could result in mode collapse problems as shown in Fig. 3.

These results are get on test set of bot and human samples after training on 50 batch sample of bot and 1000 batch of human samples that generator try to generate more samples of bot class. As shown in the figures, after 10 epoch, an epoch means training the neural network with all the training data for one cycle., discriminator is not accurate in splitting human and bot probability distributions and needs to see more samples of them. But, after 20 epochs in Fig. 3.b is shown that bot samples get low fixed probabilities in

Fig. 1 Components of the original SeqGAN framework with a LSTM network as generator and discriminator network as same as contextual LSTM (Kudugunta and Ferrara 2018) proposed for bot detection task



(a) Generator

(b) Discriminator (Contextual LSTM[36])

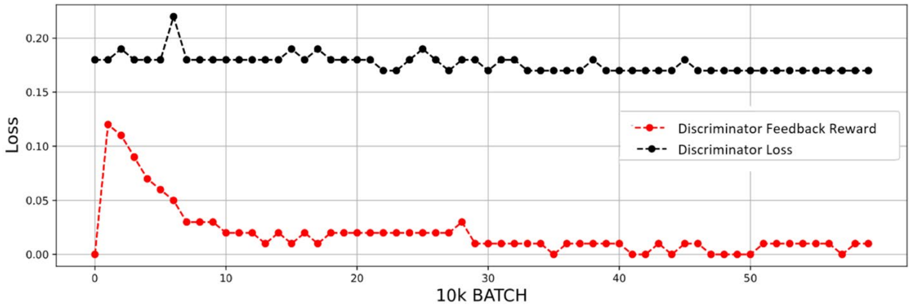
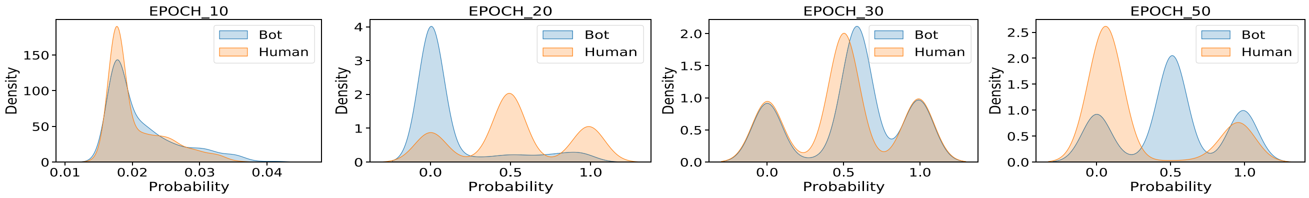


Fig. 2 Compare values of feedback reward and derived loss of discriminator training in SeqGAN framework



(a) Epoch 10th (b) Epoch 20th (c) Epoch 30th (d) Epoch 50th

Fig. 3 GAN training steps on how discriminate two classes of bot and human after **a** Epoch 10th, **b** Epoch 20th, **c** Epoch 30th and **d** Epoch 50th

distribution, while human samples are distributing toward two sides the main reason of this bad training of bot samples is generated samples of bot class, because generator samples are not qualified as same as real bot samples and need more training feedback to reach a proper quality.

In Fig. 3c and d, we also see discriminator is failed and means the game can finish but that is not able to reach a successful result in splitting probability distribution of two bot and human classes. Mathematically, the reason of this event is provided in Pascanu et al. (2013) and Zhenan et al. (2020).

3.2 Our proposed framework: GANBOT

As GAN is developed for image generation in the first has some limitations in discrete sequence generation. The main problem is known as non-differentiability due to discrete generation tasks such as text generation. They are required a sampling process to generate each token of the sentence. To solve this problem is proposed three solutions in the literature as using *Gumbel Softmax* (Kusner and Hernández-Lobato 2016), *Continues Estimation* (Gulrajani et al. 2017) and *Reinforcement Learning (RL)* (Yu et al. 2017) that RL is more popular in research studies, we also used this way.

As LSTM networks are using for both generation and classification tasks. Along with this ability, we used a common LSTM layer shared between generator and classifier. A notable issue in this model is that the LSTM layer should do classification more accurately rather than generation task for two main reasons; firstly, we showed in Fig. 2 that in a GAN framework implemented by LSTM, the loss values derived from signals feedbacked are not sufficiently informative, thereby is required a huge number of training steps to converge the outputs (Fig. 3). On the other hands, when this LSTM layer used as a discriminator or classifier model reached more informative loss values (Fig. 2) and the classifier dominates. Secondly, not only are these low feedbacks derived from discriminator effective in the long term for generation task, but they are also informative for classifier model to grab up more information from behavioral patterns of bot class (is proofed in the next section).

Practically, each LSTM cell of GAN's generator produces a bot sample that should sample a token by using the softmax layer in LSTM's output. Then, the generated samples with some real bot's samples send into the discriminator to differentiate between real and generated samples and in coming back send a feedback signal to the generator to generate better-qualified samples to be more similar to real bot samples. In which, an LSTM layer not only trains to classify bot samples (fake) from human (real) but also trains the pattern of bot sample's distribution through using the GAN framework. In fact, the LSTM layer learns human and bot distribution patterns through classification tasks, and the GAN framework.

As shown in our proposed framework in Fig. 4, human and bot samples from trainset feedforward into an embedding and common LSTM layer to classify input samples based on the last hidden state, respectively; also, this LSTM layer as a generator model tries to generate new samples using observed input samples and similar to real bot samples to fool the discriminator. The discriminator backward signaling feedback relied on itself's detection. Furthermore, the generator update by the signaling feedbacks derived from

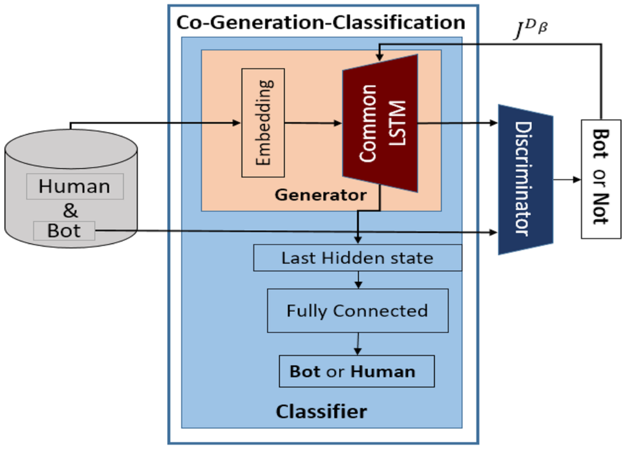


Fig. 4 The proposed GANBOT framework

the GAN's discriminator to leak more information about bot samples behavior, and the classifier also trains via loss values rely on inputs separation. In this way, we finish training steps as soon as classifier training is matured because it is dominant against generator; in fact, more training steps may lead to better classification but cannot improve generator's output like traditional SeqGAN.

parameters (λ) update through two gradients ($\nabla_\alpha J^C_\alpha$,$\nabla_\beta J^D_\beta$) derived from classifier and signaling feedback of discriminator as following formula.

$$\lambda \leftarrow \lambda + \nabla_\alpha J^{C_\alpha} + \nabla_\beta J^{D_\beta} \tag{7}$$

As the discriminator's input is including the generated bot samples and trainset bot samples, so the type of input is a sentence. The structure of the discriminator layers is similar to the classifier, in which the input sentences pass from an embedding layer and then LSTM layer and finally some dense layers to differentiate the inputs by labels.

The flow of our proposed framework is shown in algorithm 1. As we concatenate generator and classifier, we can use a discriminator loss value for generator training, too; because we solve non-differentiable back propagation through using continuous probabilities instead of generated samples as the discriminator input as IWGAN (Gulrajani et al. 2017).

```
Algorithm 1 GANBOT
Require: Dataset X contain human and bot text samples, Generator G, Classifier C, discriminator D and a
  Common LSTM layer sharing between G and C
  Initialize G, D, C parameters as $\theta$, $\beta$, $\alpha$ with the random weights
  Pre-process and tokenize X's sentences
  Load Glove embedding layer into LSTM layer
  Pre-train G to generate bot samples as eq.3
  for Training Epochs do
    for C - Training Steps do
      Train C through defined loss in eq.2
    end for
    for GD - Adversarial Training Steps do
      Generate a batch sample as a fake bot sample.
      Feed real and fake bot samples into D, and update D parameters to minimize defined cross entropy loss in
      eq.2.
      Train G by use signaling feedback of discriminator, through loss eq.6.
    end for
  end for
```

As the classifier is trained only based on the last LSTM hidden state, the objective function of that is defined in Eq. 2, J^{C_α} parameterized with α and discriminator objective function parameterized with β as follows:

$$J^{D_\beta} = -E_{S_{1:T}\sim p_{data}} \log D_\beta(S_{1:T}|\beta) - E_{S_{1:T}\sim P_{G_\theta}} \log(1 - D_\beta(S_{1:T}|\beta)) \tag{6}$$

In where, α and β, respectively, are the parameters of the classifier and discriminator; furthermore, the LSTM

4 Experimental results

In this section, we represent the results of our GANBOT framework and compare them with RNN bot detector known as contextual LSTM that is proposed by Kudugunta et al. (2018).

For evaluating, as same as contextual LSTM bot detector, we use the dataset of the work Cresci et al. (2017), which contains two groups of accounts. One group is genuine accounts that are human-operated the second group is social bot contains three subgroups as social spambots 1 (retweeters of an Italian political candidate), social spambots 2 (spammers of paid apps for mobile devices), and social spambots 3 (spammers of products on sale at Amazon.com) that are crawled from Twitter as like the Kudugunta and Ferrara (2018) we combine these tree groups in one social bot group. The dataset information is described in Table 2.

As shown in the flow of our proposed framework in algorithm 1, to transform our tweets into the LSTM network, we need to normalize and pre-process inputs for that. Furthermore, we normalize and pre-process tweets through tokenizing them using the methods proposed for Global Vectors for Word Representation (GloVE) (Pennington et al. 2014); we replace hashtags, URLs, numbers and user mentions with the unit tags as $< hashtag >$, $< url >$,$< number >$, or $< user >$, respectively, various emojis replaced with the corresponding tags such as $< hear >$, $< smile >$, $< lolmode >$, $< neutralmode >$ or $< angrymode >$, other pre-process steps are done similar to Kudugunta and Ferrara (2018).

After pre-processing steps, we need to use an embedding layer to embed token words into a corresponded vector. There are various pre-trained models for vector representation, and one of the most popular embeddings is the glove. In the glove algorithm, training is performed on aggregated global word-word co-occurrence statistics from a corpus, we use a pre-trained glove on Twitter tweets. We evaluate our results for four different dimensions of vectors as 25d, 50d, 100d, and 200d (Fig. 3).

Table 2 Main characteristic of the used datasets for evaluation, Collected by Cresci et al. (2017)

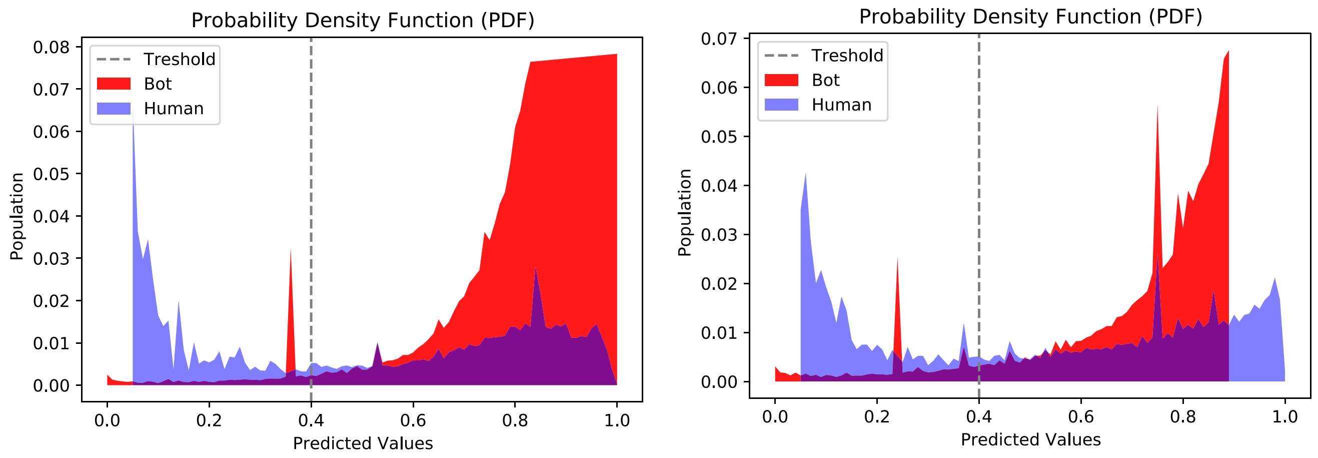
Dataset	#Accounts	#Tweets
Genuine accounts	3474	8,377,522
Social spambots 1	991	1,610,176
Social spambots 2	3457	428,542
Social spambots 3	464	1,418,626

To evaluate our model, we split our dataset into trainset, test set, and validation set, respectively, as 80, 20, 20 percentage of dataset size. We did the implementation part of our study in python language and torch library on the Graphics processing unit of Google.

4.1 Impact of GANBOT on probability density function (PDF)

The plots in Fig. 5 show the probability density function (PDF) for GANBOT (Fig. 5a) and contextual LSTM classifier (Fig. 5b), in which the horizontal axis shows the predicted probabilities for both Genuine and Bot accounts (shown with blue and red colors, respectively); also, the vertical axis shows the population percent of each probability and dashed line showing the threshold value.

Plots in Fig. 5a and b show the PDF for GANBOT and contextual LSTM method, the left side plot (GANBOT model) shows sketched PDF plot toward higher probabilities, which meant GANBOT predict bots with higher confidence. As testset data were very large, these two plots were not comparable, furthermore, we evaluate our model on some randomly selected batch sample of test data to show this comparison clearly. In these plots, as a large part of bots and humans' predicted probabilities is closely near to highest and lowest values, they are attached to the vertical axis, and by plotting the area under these curves are not observable that we ignore them. However, if we compare these two plots in



(a) Proposed method of GANBOT

(b) Contextual LSTM method as baseline [36]

Fig. 5 Compare probability density function for the contextual LSTM method and our proposed framework called GANBOT

a high probability for example in 0.8 that is clear the GANBOT shift bot probabilities toward the right side of the plot and higher probabilities.

In this model, we can reach a different number of True Positive, True Negative, False Positive, and False Negative samples by displacing the threshold value, we show these fluctuations in Fig. 7. As shown in Fig. 5 to control sequence the best threshold that holds the balance between these values is 0.4.

4.2 True predictions vs. false alarms

In this part, we evaluate the performance of GANBOT by changing the trainset size. Plots 6 is shown the predicted values on the vertical axis and the set of train set sizes are the horizontal axis, we show two different plots for the false predicted labels as False Negative (FN) and False Positive (FP) in left side Fig. (6a) and the true predicted labels as True Negative (TN) and True Positive (TP) are shown in right-side Fig. (6b). In each of these two plots, the dashed and bright color lines are for predictions of contextual LSTM and the filled and dark color lines are for GANBOT model.

As shown in the false predicted plot (Fig. 6a), for most of the trainset's sizes, the GANBOT values are fewer than contextual LSTM; in contrast with, the right-side plot (Fig. 6b) figure out the true predicted labels in both GANBOT and contextual LSTM. These results show a positive property of GANBOT model rather than contextual LSTM, because true predicted labels (TP and TN in the right-side plot) by GANBOT are nearer to higher probabilities, and in the contest, false predicted labels (FP and FN in the left side plot) by GANBOT are more near to lower probabilities, over than these properties for contextual LSTM (Fig. 7).

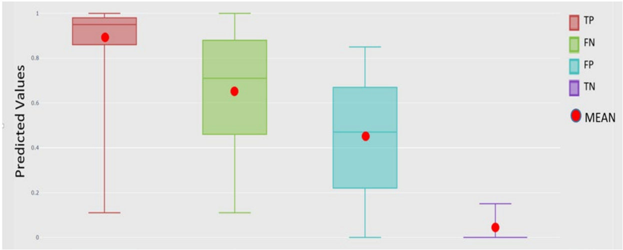
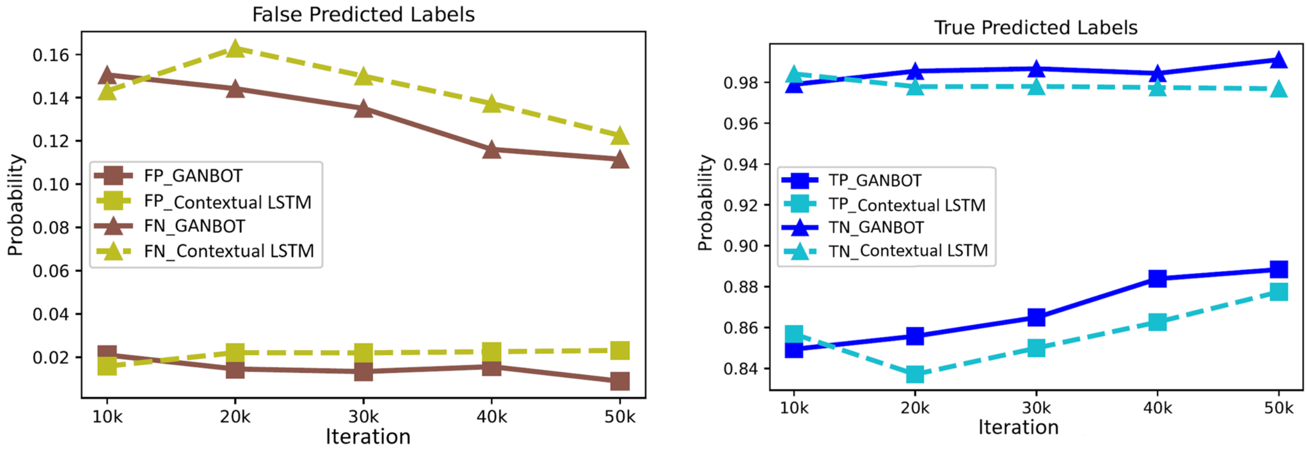


Fig. 7 Fluctuations of True Positive (TP), True Negative (TN), False Negative (FN) and False Positive (FP) when threshold are changing in range (0–0.5) for GANBOT method

In this section, we evaluate our results for both true and false prediction measures when the size of the trainset is changing from 10k to 50k that is equal to the horizontal axis in Fig. 6.

Resulted in these shown results, better prediction for four evaluation categories as TP, TN, FP, FN follows better results for corresponding evaluation metrics such as accuracy, precision, recall, F1-score, and area under the receiver operating characteristic curve. Furthermore, we compare the results of GAN-BOT framework with the contextual LSTM method for four different dimensions of GloVE embedding as 25D, 50D, 100D, 200D after 5 adversarial training epochs, this convergence speed is very lower than original GAN (Yu et al. 2017). The results shown in Table 3 are comparing contextual LSTM and GANBOT models that confirm the earlier statement on the performance of GANBOT Framework; in where, the bold values show the results of our proposed method. As shown, along with increasing all evaluation metrics, the F1-score results are more improved rather than other ones, the reason for this event is increasing



(a) False Predicted Labels

(b) True Predicted Labels

Fig. 6 Compare GANBOT method with contextual LSTM bot detector when the size of train set is changing in range of (10k : 50k) for false predicted labels and true predicted labels

Table 3 Evaluation the accuracy, recall, precision, F1-sore and AUC metrics for contextual LSTM vs. GANBOT method (presented as Contextual LSTM / GANBOT)

Contextual LSTM/ GANBOT	Accuracy	Recall	Precision	F1-score	AUC
25D GloVE	.944/**.946**	.920/**.921**	.977/**.982**	.948/**.950**	.975/**.980**
50D GloVE	.947/**.948**	.919/**.924**	.979/**.983**	.948/**.955**	.985/**.986**
100D GloVE	.948/**.949**	.922/**.925**	.980/**.986**	.951/**.955**	.989/**.989**
200D GloVE	.949/**.951**	.924/**.932**	.982/**.987**	.951/**.958**	.993/**.995**

the true predicted labels (Fig. 6b) and decreasing false predicted ones (Fig. 6b).

5 Conclusion and future work

To address the social bot detection problem has proposed several interesting methods ranging from statistical machine learning to recent neural networks such as recurrent or conventional neural networks. Aligned with this direction, we used GAN (Generative Adversarial Network) technology and customized that in a new framework named as *GANBOT* to be appropriate for classification using a common LSTM layer. In which, GAN aims to leak more information of behavioral patterns of bot samples for the classifier. Additionally, our experimental results exhibit this model has a positive effect on increasing true positive and also reducing false alarms. It is concluded that using GAN is appropriate to aim bot detection by leaking more information of behavioral bot pattern. We now intend to apply the GAN framework to the bot detection task because of its compatibility with our bot detection problem. A bot acts as same as GAN's generator to mimic the behavioral pattern of real human so that fool the bot detector; also a bot detector right as GAN's discriminator is trying to distinguish real samples from fake ones. In addition, we aim to use the results of these experiments to further our understanding of the behavioral pattern of bots, to enhance the performance of the bot detection algorithm.

Acknowledgements Mostafa Salehi is supported by a grant from IPM, Iran (No. CS1400-4-268).

References

Abu-El-Rub N, Mueen A (2019) Botcamp: Bot-driven interactions in social campaigns. In: The world wide web conference, pp 2529–2535

Aghakhani H, Aravind M, Shirin N, Christopher K, Giovanni V (2018) Detecting deceptive reviews using generative adversarial networks. In: 2018 IEEE security and privacy workshops (SPW). IEEE, pp 89–95

Alarifi A, Alsaleh M, Al-Salman AM (2016) Twitter turing test: identifying social machines. Inform Sci 372:332–346

Andriotis P, Atsuhiro T (2018) Emotional bots: content-based spammer detection on social media. In: 2018 IEEE international workshop on information forensics and security (WIFS). IEEE, pp 1–8

Antenore M, Camacho-Rodriguez JM, Panizzi E (2021) A comparative study of bot detection techniques methods with an application related to covid-19 discourse on twitter. arXiv preprint arXiv:2102.01148

Bebensee B, Nazarov N, Zhang B-T (2021) Leveraging node neighborhoods and egograph topology for better bot detection in social graphs. Social Netw Anal Mining 11(1):1–14

Beskow DM, Carley KM (2019) Its all in a name: detecting and labeling bots by their name. Comput Math Organ Theory 25(1):24–35

Bessi A, Emilio F (2016) Social bots distort the 2016 us presidential election online discussion. First Monday 21(11–7)

Bin W, Liu L, Yang Y, Zheng K, Wang X (2020) Using improved conditional generative adversarial networks to detect social bots on twitter. IEEE Access 8:36664–36680

Bodaghi A, Goliaei S, Salehi M (2019) The number of followings as an influential factor in rumor spreading. Appl Math Comput 357:167–184

Cai C, Li L, Zeng D (2017) Detecting social bots by jointly modeling deep behavior and content information. In: Proceedings of the 2017 ACM on conference on information and knowledge management, pp 1995–1998

Cai C, Li L, Zengi D (2017) Behavior enhanced deep bot detection in social media. In: 2017 IEEE international conference on intelligence and security informatics (ISI). IEEE, pp 128–130

Chavoshi N, Mueen A (2018) Model bots, not humans on social media. In: 2018 IEEE/ACM international conference on advances in social networks analysis and mining (ASONAM). IEEE, pp 178–185

Chen Z, Subramanian D (2018) An unsupervised approach to detect spam campaigns that use botnets on twitter. arXiv preprint arXiv:1804.05232

Cresci S, Pietro RD, Petrocchi M, Spognardi A, Tesconi M (2017) 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, pp 963–972

Daouadi KE, Rebaï RZ, Amous I (2019) Bot detection on online social networks using deep forest. In: Computer science on-line conference. Springer, pp 307–315

Davis CA, Varol O, Ferrara E, Flammini A, Menczer F (2016) Botornot: a system to evaluate social bots. In: Proceedings of the 25th international conference companion on world wide web, pp 273–274

Deb A, Luceri L, Badaway A, Ferrara E (2019) Perils and challenges of social media and election manipulation analysis: the 2018 us midterms. In: Companion proceedings of the 2019 world wide web conference, pp 237–247

Dorri A, Abadi M, Dadfarnia M (2018) Socialbothunter: Botnet detection in twitter-like social networking services using semi-supervised collective classification. In: 2018 IEEE 16th international conference on dependable, autonomic and secure computing, In: 16th International conference on pervasive intelligence and computing, In: 4th International conference on big data intelligence and computing and cyber science and technology congress (DASC/PiCom/DataCom/CyberSciTech). IEEE, pp 496–503

Fedus W, Goodfellow I, Dai AM (2018) Maskgan: better text generation via filling in the_. arXiv preprint arXiv:1801.07736, 2018

Ferrara E (2017) Disinformation and social bot operations in the run up to the 2017 french presidential election. arXiv preprint arXiv:1707.00086

Ferrara E (2020) What types of covid-19 conspiracies are populated by twitter bots? First Monday 25(6)

Ferrara E, Varol O, Davis C, Menczer F, Flammini A (2016) The rise of social bots. Commun ACM 59(7):96–104

Gilani Z, Farahbakhsh R, Tyson G, Wang L, Crowcroft K (2017) Of bots and humans (on twitter). In: Proceedings of the 2017 IEEE/ACM international conference on advances in social networks analysis and mining, pp 349–354

Gilani Z, Kochmar E, Crowcroft J (2017) Classification of twitter accounts into automated agents and human users. In: Proceedings of the 2017 IEEE/ACM international conference on advances in social networks analysis and mining, pp 489–496

Goodfellow I, Pouget-Abadie J, Mirza M, Xu B, Warde-Farley D, Ozair S, Courville A, Bengio Y (2014) Generative adversarial nets. In: Advances in neural information processing systems, pp 2672–2680

Gui J, Sun Z, Wen Y, Tao D, Ye J (2020) A review on generative adversarial networks: algorithms, theory, and applications. arXiv preprint arXiv:2001.06937

Gulrajani I, Ahmed F, Arjovsky M, Dumoulin V, Courville A (2017) Improved training of wasserstein gans. arXiv preprint arXiv:1704.00028

Guo J, Lu S, Cai H, Zhang W, Yu Y, Wang J (2018) Long text generation via adversarial training with leaked information. In: Thirty-second AAAI conference on artificial intelligence

Halawa H, Beznosov K, Coskun B, Liu M, Ripeanu M (2019) Forecasting suspicious account activity at large-scale online service providers. In: International conference on financial cryptography and data security. Springer, pp 569–587

Hochreiter S, Schmidhuber J (1997) Long short-term memory. Neural Comput 9(8):1735–1780

Howard PN, Kollanyi B, Woolley S (2016) Bots and automation over twitter during the us election. In: Computational propaganda project: Working paper series, pp 1–5

Hurtado S, Ray P, Marculescu R (2019) Bot detection in reddit political discussion. In: Proceedings of the fourth international workshop on social sensing, pp 30–35

Igawa RA, Barbon Jr S, Paulo KCS, Kido GS, Guido RC, Proença Júnior ML, da Silva IN (2016) Account classification in online social networks with lbca and wavelets. Inform Sci 332:72–83

Iqbal T, Qureshi S (2020) The survey: text generation models in deep learning. J King Saud Univ Comput Inform Sci

Barbon Jr S, Campos GFC, Tavares GM, Igawa RA, Proença Jr ML, Guido RC (2018) Detection of human, legitimate bot, and malicious bot in online social networks based on wavelets. ACM Trans Multim Comput Commun Appl TOMM), 14(1s):1–17

Kudugunta S, Ferrara E (2018) Deep neural networks for bot detection. Inform Sci 467:312–322

Kusner MJ, Hernández-Lobato JM (2016) Gans for sequences of discrete elements with the gumbel-softmax distribution. arXiv preprint arXiv:1611.04051

Luca L, Ashok D, Silvia G, Emilio F (2019) Evolution of bot and human behavior during elections. First Monday 24(9)

Mazza M, Cresci S, Avvenuti M, Quattrociocchi W, Tesconi M (2019) Rtbust: exploiting temporal patterns for botnet detection on twitter. In: Proceedings of the 10th ACM conference on web science, pp 183–192

Orabi M, Mouheb D, Aghbari ZA, Kamel I (2020) Detection of bots in social media: a systematic review. Inform Process Manage 57(4):102250

Pascanu R, Mikolov T, Bengio Y (2013) On the difficulty of training recurrent neural networks. In: International conference on machine learning, pp 1310–1318

Pennington J, Socher R, Manning CD (2014) Glove: global vectors for word representation. In: Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP), pp 1532–1543

Ping H, Qin S (2018) A social bots detection model based on deep learning algorithm. In: 2018 IEEE 18th international conference on communication technology (ICCT). IEEE, pp 1435–1439

Ranjbar V, Salehi M, Jandaghi P, Jalili M (2018) Qanet: Tensor decomposition approach for query-based anomaly detection in heterogeneous information networks. IEEE Trans Knowl Data Eng 31(11):2178–2189

Rodríguez-Ruiz J, Mata-Sánchez JI, Monroy R, Loyola-González O, López-Cuevas A (2020) A one-class classification approach for bot detection on twitter. Comput Sec 91:101715

Shehnepoor S, Salehi M, Farahbakhsh R, Crespi N (2017) Netspam: A network-based spam detection framework for reviews in online social media. IEEE Trans Inform Foren Security 12(7):1585–1595

Shehnepoor S, Togneri R, Liu W, Bennamoun M (2020) Gangster: a fraud review detector based on regulated gan with data augmentation. arXiv preprint arXiv:2006.06561

Stanton G, Irissappane AA (2019) Gans for semi-supervised opinion spam detection. arXiv preprint arXiv:1903.08289

Stella M, Ferrara E, De Domenico M (2018) Bots increase exposure to negative and inflammatory content in online social systems. Proc Natl Acad Sci 115(49):12435–12440

Tuan Y-L, Lee H-Y (2019) Improving conditional sequence generative adversarial networks by stepwise evaluation. IEEE/ACM Trans Audio Speech Language Process 27(4):788–798

Turing AM (1950) Computing machinery and intelligence. Mind 59(236):433

Valliyammai C, Devakunchari R (2019) Distributed and scalable sybil identification based on nearest neighbour approximation using big data analysis techniques. Cluster Computing 22(6):14461–14476

Varol O, Ferrara E, Davis CA, Menczer F, Flammini A (2017) Online human-bot interactions: detection, estimation, and characterization. In: Eleventh international AAAI conference on web and social media

Velayutham T, Tiwari PK (2017) Bot identification: helping analysts for right data in twitter. In: 2017 3rd international conference on advances in computing, Communication & automation (ICACCA) (Fall). IEEE, pp 1–5

Wang Y, Wu C, Zheng K, Wang X (2018) Social bot detection using tweets similarity. In: International conference on security and privacy in communication systems. Springer, pp 63–78

Yang K-C, Varol O, Hui P-M, Menczer F (2020) Scalable and generalizable social bot detection through data selection. In: Proceedings of the AAAI conference on artificial intelligence, vol 34, pp 096–1103

Yu L, Zhang W, Wang J, Yu Y (2017) Seqgan: aequence generative adversarial nets with policy gradient. In: Thirty-first AAAI conference on artificial intelligence

Zhao J, Liu X, Yan Q, Li B, Shao M, Peng H (2020) Multi-attributed heterogeneous graph convolutional network for bot detection. Inform Sci 537:380–393