Better Safe Than Sorry: an Adversarial Approach to improve Social Bot Detection

Stefano Cresci, Marinella Petrocchi
IIT-CNR, Pisa, Italy
[name.surname]@iit.cnr.it

Angelo Spognardi
Dept. of Computer Science,
Sapienza University of Rome, Italy
spognardi@di.uniroma1.it

Stefano Tognazzi
IMT School for Advanced Studies
Lucca, Italy
stefano.tognazzi@imtlucca.it

ABSTRACT

The arm race between spambots and spambot-detectors is made of several cycles (or generations): a new wave of spambots is created (and new spam is spread), new spambot filters are derived and old spambots mutate (or *evolve)* to new species. Recently, with the diffusion of the adversarial learning approach, a new practice is emerging: to manipulate on purpose target samples in order to make stronger detection models. Here, we manipulate generations of Twitter social bots, to obtain - and study - their possible future evolutions, with the aim of eventually deriving more effective detection techniques. In detail, we propose and experiment with a novel genetic algorithm for the synthesis of online accounts. The algorithm allows to create synthetic *evolved* versions of current state-of-the-art social bots. Results demonstrate that synthetic bots really escape current detection techniques. However, they give all the needed elements to improve such techniques, making possible a proactive approach for the design of social bot detection systems.

KEYWORDS

Social bots; online social networks security; adversarial classifier evasion; genetic algorithms; Twitter

ACM Reference Format:
Stefano Cresci, Marinella Petrocchi, Angelo Spognardi, and Stefano Tognazzi. 2019. Better Safe Than Sorry: an Adversarial Approach to improve Social Bot Detection. In *11th ACM Conference on Web Science (WebSci '19), June 30-July 3, 2019, Boston, MA, USA.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3292522.3326030

1 INTRODUCTION

A worrying peculiarity of spammers and bots (or spambots) is that they *evolve* over time, adopting sophisticated techniques to evade well-established detection systems [12, 49]. In the context of Online Social Networks (ONSs), newer social spambots often feature advanced characteristics that make them way harder to detect with respect to older ones, since capable of mimicking human behaviors and interaction patterns better than ever before [12, 20]. These automated accounts represent – to the best of the literature knowledge – the third and most novel generation of social bots, following the original wave dated back in the 00s, and passing through a second generation dated around 2011 [49]. The latest social bots are capable of sharing (credible) fake news, inflating the popularity of OSN users, and reshaping political debates [40, 41]. Given this picture, it is not surprising that evolution mechanisms (together with coordination and synchronization ones) represent one of the key factors that currently allow malicious accounts to massively tamper with our social ecosystems [14].

Despite malicious accounts evolution representing a sort of Pandora's box, little to no attention has been posed towards studying – and possibly anticipating – such evolution. In fact, in past years, as social bots gradually became clever in escaping detection, scholars and OSNs administrators tried to keep pace (i.e., *reacted*), by proposing ever more complex detection techniques, as described in the Related Work section. The natural consequence of this *reactive* approach – according to which a new technique is designed only after having collected evidence of new mischiefs of evolved bots – is that researchers and OSN admins are constantly one step behind the bot developers [43].

The classification task of recognising if an online account is genuine or not is *adversarial* in nature, being focused on distinguishing *bad* samples from *good* ones. Nonetheless, a new approach is gaining momentum in the wide field of artificial intelligence, leveraging the concept of *adversarial learning*: the automatic learning within a hostile environment [28]. This allows to both discover vulnerabilities in learning algorithms and to test algorithmic techniques which yield more robust learning [44]. Intuitively, the evolution of new waves of social bots can be seen as a problem of *adversarial classifier evasion*, where the attacker changes the generated samples to evade detection. Thus, the core idea of this paper is to manipulate *on purpose* target samples in order to produce stronger detection models. Inspired by the adversarial learning approach, for the first time to the best of our knowledge, we carry out an exploratory investigation to define and implement a *proactive* technique to study and detect evolving social bots.

Specifically, we aim at answering the following critical, yet unexplored, research questions:

RQ1 – *Can we develop an analytical framework for simulating spambot evolutions?*

RQ2 – *Can we use such framework for synthesizing new generations of spambots? And, most importantly, are these evolved spambots capable of going undetected by state-of-the-art techniques?*

RQ3 – *Can we leverage this proactive and adversarial study of spambot evolutions to improve current detection techniques?*

Approach. Our methodological approach to answer the three questions stems from the so-called *digital DNA* technique [11], where the behavioral lifetime of an account is encoded as a sequence of

WebSci '19, June 30-July 3, 2019, Boston, MA, USA

ACM ISBN 978-1-4503-6202-3/19/06…$15.00
https://doi.org/10.1145/3292522.3326030

characters, built according to the chronological sequence of actions performed by the account. The adoption of this behavioral modeling technique offers a DNA-like representation for the lifetime of each account, including new social spambots. We feed the DNA sequences to a custom-designed genetic algorithm, so as to study possible evolutions of the accounts [36]. The customized genetic algorithm iteratively selects the *best* evolutions, so as to converge towards synthetic bots capable of resembling behavioral characteristics of legitimate accounts. Furthermore, by constraining possible evolutions within the algorithm, the approach allows to obtain synthetic accounts capable of performing specific tasks (e.g., viral marketing, message spamming, mass retweeting, etc.). Notably, the digital DNA behavioral modeling technique has been exploited in the past as the building block of a state-of-the-art detection system [13]. Thus, we can also apply such detection technique on the synthetically evolved accounts, to evaluate whether they are capable of evading detection.

Contributions. This work contributes along several dimensions. Firstly, (i) we propose GenBot, a novel genetic algorithm specifically designed for social spambot evolutions. By employing a cost function that quantifies the difference between a new generation of spambots and a group of legitimate accounts, GenBot is capable of generating spambots whose behavior is similar to that of the legitimate ones. Notably, our results outperform previous attempts to simulate the behavior of human accounts [16]. Then, (ii) we discuss the design of an analytical framework for simulating possible social spambot evolutions that leverages both the digital DNA behavioral modeling technique and the genetic algorithm previously defined. We experiment with this framework to synthesize a novel generation of evolved spambots, with the aim of producing adversarial samples. Furthermore, (iii) we assess the extent to which the newly synthesized social bots are detected by 3 state-of-the-art techniques. Results show that, with the proposed framework, it is possible to create an adversarial behavioral fingerprint that allow bots to escape detection. Finally, (iv) by studying the characteristics of the synthetic evolved spambots, we draw useful insights into which account features could be considered in order to improve the detection of real evolving spambots.

Broadening the approach. We ground our study on a recently-proposed proactive approach to spambot detection [43]. In order to carry out extensive experimentation on real-world data, without loss of generality, here we implement it by focusing on the behavior (i.e., the sequences of actions that accounts perform) of spambots and legitimate accounts. This choice opens up the possibility to leverage, for our experiments, the *digital DNA* behavioral modeling technique [11] as well as the *social fingerprinting* spambot detection technique [13]. However, despite this particular implementation of the proactive approach, similar analyses could be carried out, by relying on different modeling and spambot detection techniques, such as those based on network/graph analysis and those based on content analysis.

Reproducibility. Both the data[1] and the code[2] used in this study are publicly available for scientific purposes.

[1] http://mib.projects.iit.cnr.it/dataset.html
[2] http://sysma.imtlucca.it/tools/digdna-genetic-algorithm/

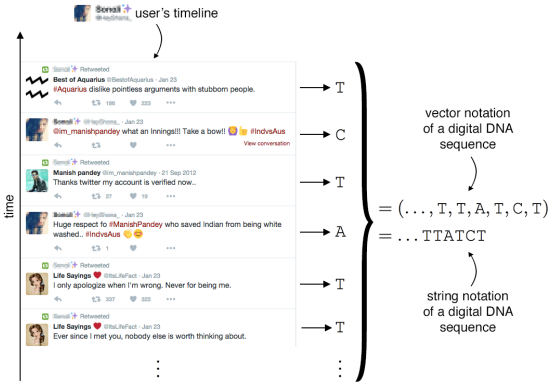

Figure 1: Excerpt of a digital DNA extraction process for a Twitter user with the alphabet $\mathbb{B} = \{\mathtt{A},\mathtt{C},\mathtt{T}\}$, where T is assigned to every tweet, C to every reply, and A to every retweet.

2 BACKGROUND AND NOTATION

Here, we provide a succinct description of the main concepts related to digital DNA sequences and genetic algorithms, as well as notations and metrics adopted in the remainder of this study.

2.1 Digital DNA

Digital DNA sequences. We define a digital DNA sequence $\mathbf{s}$ as a row-vector of characters (i.e., a string),

$$\mathbf{s} = (b_1, b_2, \ldots, b_n) \quad b_i \in \mathbb{B} \;\; \forall \;\; i = 1, \ldots, n$$

Characters b_i in $\mathbf{s}$ are also called the (DNA) *bases* and are drawn from a finite set $\mathbb{B}$, called *alphabet*,

$$\mathbb{B} = \{B_1, B_2, \ldots, B_N\} \quad \forall\, i \neq j : B_i \neq B_j$$

Online users' behaviors can be represented by encoding each user action, in chronological order, with an appropriate base. In this way, we obtain the sequence of characters that makes up the digital DNA sequence of the user. For example, Figure 1 shows the process of extracting the digital DNA sequence of a Twitter user, by scanning its timeline according to the alphabet $\mathbb{B} = \{\mathtt{A}, \mathtt{C}, \mathtt{T}\}$, where T is assigned to every tweet, C to every reply, and A to every retweet. A digital DNA sequence can be represented, then, with a compact string like $\mathbf{s} = \ldots \mathtt{TTATCT}$. Additional details about the theoretical foundations of digital DNA can be found in [11, 13].

Similarity between digital DNA sequences. In order to analyze groups of users rather than single users, we need to study multiple digital DNA sequences as a whole. A group $\mathbf{A}$ of $M = |\mathbf{A}|$ users can be described by the strings representing the digital DNA sequences of the M users.

To perform our analyses on digital DNA sequences, we can rely on recent advances in the fields of bio-informatics and string mining [23]. One of the possible means to quantify similarities between sequential data representations is the *longest common substring* [3]. Given two strings, $\mathbf{s}_i$ of length n and $\mathbf{s}_j$ of length m, their longest common substring (henceforth LCS) is the longest string that is a substring of both $\mathbf{s}_i$ and $\mathbf{s}_j$. For example, given

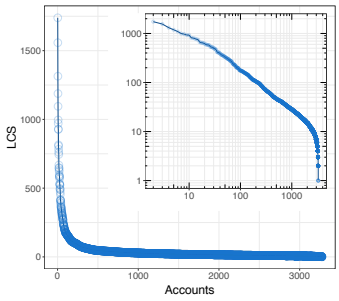

Figure 2: LCS curve of a group of legitimate (human-operated) accounts.

$\mathbf{s}_i$ = MASSACHUSETTS and $\mathbf{s}_j$ = PARACHUTE, their LCS is the string ACHU and the LCS length is 4. The extended version of this problem, which considers an arbitrary finite number of strings, is called the *k-common substring* problem [9]. In this case, given a vector $\mathbf{A} = (\mathbf{s}_1, \ldots, \mathbf{s}_M)$ of M strings, the problem is that of finding the LCS that is common to at least k of these strings, for each $2 \leq k \leq M$. Notably, both the *longest common substring* and the *k-common substring* problems can be solved in linear time and space, by resorting to the generalized suffix tree and by implementing state-of-the-art algorithms, such as those proposed in [3]. Given that, in the *k-common substring* problem, the LCS is computed for each $2 \leq k \leq M$, it is possible to plot a *LCS curve*, showing the relationship between the length of the LCS and the number k of strings [13].

Figure 2 depicts the LCS curve computed for a group of legitimate (human-operated) Twitter accounts, via the alphabet $\mathbb{B} = \{\mathtt{A}, \mathtt{C}, \mathtt{T}\}$. On the x axis is reported the number k of accounts (corresponding to the k digital DNA sequences used to compute LCS values) and on the y axis the length of the LCS common to at least k accounts. Therefore, each point in a LCS curve corresponds to a subset of k accounts that share the longest substring (of length y) among all those shared between all the other possible subsets of k accounts.

A LCS curve is a representation of the behavioral similarities among a group of users, since it is an ordered sequence of substring lengths. To obtain a single value as *measure* of similarity for the whole group, we can compute the area under the LCS curve (AUC) [19, 21]. Since LCS curves are discrete functions defined over the $[2, M]$ range, their AUC can be computed straightaway, without approximations, with the following trapezoid rule,

$$\text{AUC} = \sum_{k=3}^{M} \frac{(\text{LCS}[k-1] + \text{LCS}[k])\Delta_k}{2} \quad (1)$$

Compared to LCS, the definition of AUC, given in Equation (1), allows to quantitatively and directly compare the overall behavioral similarity among different groups. This notion of AUC is exploited in Section 5 to evaluate the results of this study.

2.2 Genetic algorithms

Genetic algorithms [36] represent a popular meta-heuristic technique to solve optimization problems. It is inspired by the natural evolution, in which only the best candidates of some species survive across several subsequent generations. Starting from a random sample of candidate solutions (a so-called *population* of *individuals*), only the best ones are elected to evolve. Similarly to what happens in nature, through a mechanism of recombination and mutation of those individuals, a new *generation* is obtained. This evolved generation is expected to have a better quality with respect to the previous one. The quality of individuals is evaluated by associating a *fitness score* to all of them. The score is computed by a given *fitness function* that is designed according to the task at hand. During each generation, the current population is modified via either a single input function called *mutation* or a two input function called *crossover*. The outputs of these functions are called *offsprings* and a new generation is formed by merging some of the individuals from the previous generation with some of the offsprings. In our experimental scenario, when we refer to an individual, we refer to a group of users. Thus, our population is composed of different groups of users that evolve passing from a generation to the next one.

Notations. We call P_0 the initial population and P_i the population at the i-th generation. Given a population P_i, $G_j = \{u_1, u_2, \ldots, u_M\}$ is the j-th individual (i.e., a group of users) of P_i. Then, $G_j[k] = u_k$ is the k-th user of the group G_j, characterized by its digital DNA sequence. In other words, u_k is a digital DNA sequence encoding the behavior of the k-th Twitter user. We define v_j as the fitness score of the j-th individual. The population $P_i = \{(G_1, v_1), (G_2, v_2), \ldots, (G_J, v_J)\}$ is a set of pairs containing the individuals and their associated fitness scores.

3 AN ALGORITHM FOR SIMULATING BOT EVOLUTIONS

The design of a custom genetic algorithm for solving a given task involves the definition of the parameters and functions used by the algorithm in its iterative execution. In this section, we first describe the design choices and building blocks of the GenBot algorithm and we conclude by defining the algorithm itself.

3.1 Building blocks: fitness, mutation and crossover

Fitness. In our scenario, individuals of best quality are groups of bots that best emulate the behavior of a group of legitimate (human-operated) accounts. To formalize this intuition, we rely on the notion of behavioral similarity expressed by digital DNA and LCS curves. More specifically, our goal for this task is to *minimize* the distance between the LCS curve (i.e., the behavioral representation) of a group of legitimate accounts and the LCS curve of a population of synthetic evolved bots.

We rely on the Kullback-Liebler distance (D_{KL}) to compute the distance between two LCS curves, since it has already been fruitfully employed in recent similar work [16, 46]. D_{KL} is an information theoretic metric that measures how much information is lost when a target probability distribution $P_X(x)$ is approximated by $\hat{P}_X(x)$. In detail, D_{KL} is the symmetric version of the Kullback-Liebler divergence d_{KL} (defined as, $d_{KL}(\hat{P}_X, P_X) = \sum_x \ln\left(\frac{\hat{P}_X(x)}{P_X(x)}\right)\hat{P}_X(x)$), where,

$$D_{KL}(\hat{P}_X, P_X) = \frac{d_{KL}(P_X, \hat{P}_X) + d_{KL}(\hat{P}_X, P_X)}{2} \quad (2)$$

```
Fit(G_j, b)
1 g ← LCS(G_j)
2 v_j ← D_KL(g, b)
3 return v_j
```

```
GCO(G_x, G_y, r)
1 for i ∈ {1, ..., r} do
2     G_xy[i] ← G_x[i]
3     G_yx[i] ← G_y[i]
4 for i ∈ {r + 1, ..., |G_x|} do
5     G_xy[i] ← G_y[i]
6     G_yx[i] ← G_x[i]
7 return G_xy, G_yx
```

In the GenBot algorithm, the target distribution $P_X(x)$ is obtained from the LCS curve of legitimate accounts, while the approximating distribution $\hat{P}_X(x)$ is obtained from the LCS curve of a group of bots. The D_{KL} distance is computed by the fitness function Fit$(\cdot)$ that accepts as input an individual G_j and the target LCS curve b of legitimate accounts. The output is a scalar v_j that represents the fitness score of the individual G_j.

Mutation. The mutation is an operator commonly used in many genetic algorithms [36]. It typically consists in making some bases of the DNA sequences of an individual to mutate into a different base (e.g., A $\xrightarrow{\text{mutation}}$ C).

Previous studies showed that the distribution of bases within the digital DNA sequences of legitimate users is not uniform [13, 16]. Quite intuitively, some actions tend to occur more often than others, such as the tweeting action. For this reason, in GenBot we design a mutation operator that favors mutations from the C (replies) and T (retweets) bases to the A (tweets) base. Nonetheless, also mutations from A to C and T are possible, although with a lower probability. Our Mutation$(\cdot)$ function accepts two parameters: P_i, which is the full population at the i-th generation of the genetic algorithm; and L, which is the length of the digital DNA sequences. The output returned by the function is the mutated P_i population.

```
Mutation(P_i, L)
1  for (G_j, v_j) ∈ P_i do
2      for u_k ∈ G_j do
3          for l ∈ {1, ..., L} do
4              r ← rand()
5              if r < MUT-PROB then
6                  if u_k[l] = C or u_k[l] = T then
7                      u_k[l] = A
8                  else
9                      if r < 0.5 then
10                         u_k[l] = C
11                     else
12                         u_k[l] = T
13 return P_i
```

Crossover. Traditionally, a crossover operator has two parent individuals as input, and generates two offsprings as the output [36]. The offsprings are obtained via a recombination of the DNA sequences of the two parents. Remarkably, in this study an individual G_j is a group of users $G_j = \{u_1, u_2, \ldots, u_M\}$ rather than a single user with its DNA sequence. Thus, by following the traditional crossover approach, we have two groups of users as input (the parents) and we obtain two groups of users as output (the offsprings). Recombinations at this level (i.e., at the group-level) can occur by mixing users between the two parent groups, rather than by mixing DNA sequences. Within GenBot, this crossover strategy is implemented with the function GCO$(\cdot)$ (***G**roup **C**ross**O**ver*). GCO$(\cdot)$ uses a one-point crossover technique. In detail, it randomly selects two groups $G_x, G_y \in P_i$ as the parents and a random crossover point r. Then, it generates the offspring G_{xy} as the combination of the first parent up to point r with the second parent from point r onwards, and the offspring G_{yx} viceversa. Figure 3 gives an intuitive idea of how our group-level crossover operator works.

Since our individuals are composed of many users, we can complement the previous crossover strategy with additional fine-grained crossovers working at the user-level. Specifically, we define two user-level crossover operators, referred to as the ***U**ser **C**ross**O**ver* operator, defined in function UCO$(\cdot)$, and the ***U**ser **R**everse **C**ross**O**ver* operator, defined in function URCO$(\cdot)$. Function UCO$(\cdot)$ is a one-point crossover operator that acts at user-level. Similarly to the GCO$(\cdot)$ operator, two parents $u_x, u_y \in G_j$ and one crossover point r are randomly picked. Next, two offsprings – respectively u_{xy} and u_{yx} – are generated as the combination of the first parent up to point r with the second parent from point r onwards, and viceversa. Figure 4 gives an intuitive idea of how our UCO$(\cdot)$ user-level crossover operator works, in comparison with the group-level crossover.

Function URCO$(\cdot)$ differs from UCO$(\cdot)$ in the way the second parent u_y is exploited. In fact, in URCO$(\cdot)$, the digital DNA sequence of u_y is reversed before being recombined with that of the first parent u_x. This simple operation allows to create much more variability in the DNA sequences of the offsprings and it demonstrates very effective in practice.

Contrarily to traditional genetic algorithms, we exploit the richness of the DNA-based behavioral representations, by designing a multi-level crossover strategy. At the user-level, the fine-grained UCO$(\cdot)$ and URCO$(\cdot)$ operators apply recombinations to the DNA sequences of single users. Furthermore, at the group-level, the coarse-grained GCO$(\cdot)$ operator shuffles users between different groups.

```
UCO(u_x, u_y, r)
1 for i ∈ {1, ..., r} do
2     u_xy[i] ← u_x[i]
3     u_yx[i] ← u_y[i]
4 for i ∈ {r + 1, ..., |u_x|} do
5     u_xy[i] ← u_y[i]
6     u_yx[i] ← u_x[i]
7 return u_xy, u_yx
```

```
URCO(u_x, u_y, r)
1 for i ∈ {1, ..., r} do
2     u_xy[i] ← u_x[i]
3     u_yx[i] ← u_y[|u_y| − i]
4 for i ∈ {r + 1, ..., |u_x|} do
5     u_xy[i] ← u_y[|u_y| − i]
6     u_yx[i] ← u_x[i]
7 return u_xy, u_yx
```

3.2 The GenBot algorithm

We implemented GenBot by following the steps of the (1+1)-evolutionary algorithm scheme [18]. In its simplest definition, a (1+1)-EA is a *randomized hill climbing technique* [26] that relies on mutations only. In the GenBot algorithm, the simple (1+1)-EA scheme is extended by the adoption of a multi-level crossover strategy (i.e., 1 group-level and 2 user-level crossovers), as previously defined.

The core of the GenBot algorithm is represented in Algorithm 1 by the *for loop* at lines 7–37. Each iteration of the loop applies the mutation and crossover operators to obtain a new generation of spambots. In detail, the evolutionary steps in GenBot begin by mutating the current population and continue by substituting

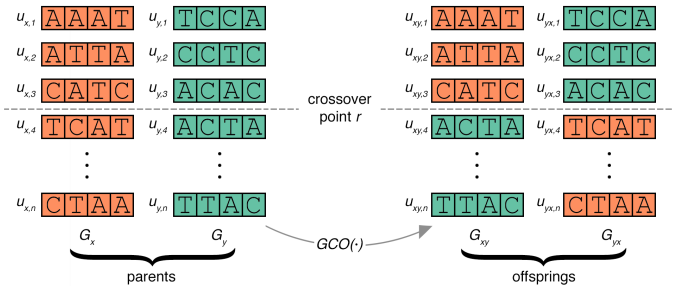

Figure 3: Group-level crossover: GCO(·) operator.

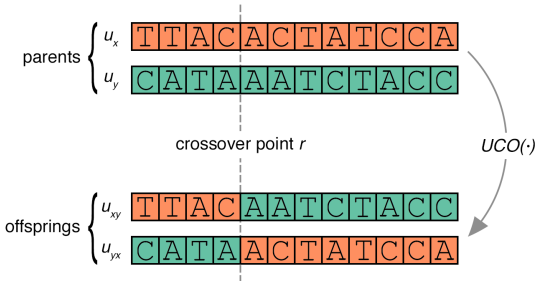

Figure 4: User-level crossover: UCO(·) operator.

Algorithm 1 GenBot

input : target legitimate group $G_{\text{legitimate}}$, initial population P_0
output: last generation of evolved spambots P_{best}

$b \leftarrow \texttt{LCS}(G_{\text{legitimate}})$
$U \leftarrow \texttt{numOfUsers}(G_{\text{legitimate}})$
$T \leftarrow \texttt{numOfTweets}(G_{\text{legitimate}})$
$P_{\text{best}} \leftarrow$ **null**
for $G_i \in P_0$ **do**
  $v_i \leftarrow \texttt{Fit}(G_i, b)$ // initial fitness score
for $i \in \{1, \ldots, \text{MAX-GEN}\}$ **do**
  mut = $\texttt{Mutation}(P_{i-1}, T)$ // apply mutations
  for $m_j \in$ mut **do**
    $mv_j \leftarrow \texttt{Fit}(m_j, b)$
    if $mv_j < v_j$ **then**
      $(G_j, v_j) \leftarrow (m_j, mv_j)$
  for $(G_k, v_k) \in P_i$ **do**
    $x \leftarrow \texttt{rand}(1, \text{POP-SIZE})$
    $y \leftarrow \texttt{rand}(1, \text{POP-SIZE})$
    $G_{xy} \leftarrow \texttt{GCO}(G_x, G_y, U)$ // apply group crossovers
    for $j \in \{1, \ldots, \text{NUM-URCO}\}$ **do** // apply user reverse crossovers
      $u_x \leftarrow \texttt{rand}(1, U)$
      $u_y \leftarrow \texttt{rand}(1, U)$
      $(u_{xy}, u_{yx}) \leftarrow \texttt{URCO}(G_j[u_x], G_j[u_y], T)$
      $G_{xy}[u_x] \leftarrow u_{xy}$
      $G_{xy}[u_y] \leftarrow u_{yx}$
    $mv_k \leftarrow \texttt{Fit}(G_{xy}, b)$
    if $mv_k < v_k$ **then**
      $(G_k, v_k) \leftarrow (G_{xy}, mv_k)$
  for $(G_k, v_k) \in P_i$ **do**
    for $j \in \{1, \ldots, \text{NUM-UCO}\}$ **do** // apply user crossovers
      $u_x \leftarrow \texttt{rand}(1, U)$
      $u_y \leftarrow \texttt{rand}(1, U)$
      $(u_{xy}, u_{yx}) \leftarrow \texttt{UCO}(G_j[u_x], G_j[u_y], T)$
      $G_{xy}[u_x] \leftarrow u_{xy}$
      $G_{xy}[u_y] \leftarrow u_{yx}$
    $mv_k \leftarrow \texttt{Fit}(G_{xy}, b)$
    if $mv_k < v_k$ **then**
      $(G_k, v_k) \leftarrow (G_{xy}, mv_k)$
  $P_{i+1} \leftarrow P_i$
  $P_{\text{best}} \leftarrow P_i$ // update evolved spambots
return P_{best}

the individuals that improve as a consequence of the mutations. Then, the group-level (coarse-grained) crossover GCO(·) is applied. Finally, also the 2 user-level (fine-grained) crossovers are applied. Specifically, at first URCO(·) is applied and the obtained offsprings are evaluated. Only those offsprings that improved the fitness score are retained. Subsequently, UCO(·) is applied and the offsprings are evaluated one last time, thus obtaining a new population that becomes the starting point of the next iteration of the algorithm.

Notably, traditional evolutionary heuristics based on genetic algorithms perform only one update of the population at each iteration of the algorithm. In GenBot however, each generation is the combination of three intermediate generations, respectively obtained via (i) mutations, (ii) a combination of group-level and reverse user-level crossovers, and (iii) user-level crossovers.

4 EXPERIMENTS, SETUP AND EVALUATION

4.1 Dataset

The dataset for this study is composed of the timelines of 3,474 legitimate Twitter accounts.

In order to build this dataset of certified human-operated accounts, random Twitter users were contacted by mentioning them in tweets. Then, contacted users were asked simple questions in natural language. Possible answers to such questions were collected by means of a Twitter crawler[3]. Upon manually verifying the answers, all 3,474 accounts that answered were certified as legitimate ones. Notably, this dataset has already been used in recent works [11–13, 16] and it is considered an important resource in the field of spambot and automation detection.

4.2 Experimental setup

The GenBot algorithm is implemented in C++ and the code is publicly available for scientific purposes (link in the introduction). For an efficient, linear-time computation of the LCS curves, we rely on an adapted version of the GLCR toolkit[4] implementing the algorithms in [3]. All the experiments ran on a machine with an Intel Xeon E7-4830v4, with a 64-bits architecture at 2 GHz, 112 cores and 500 GB of RAM. As the reference with which to compare our results, we consider the last (most recent) 2,000 actions performed by the legitimate accounts in our dataset.

We run GenBot with a population of 30 individuals per run (POP-SIZE) and a generation limit of 20,000 epochs as a stopping criterion (MAX-GEN). The initial population P_0 is composed of 30 identical individuals. Each individual represents a group of accounts and the starting point for all the individuals is a DNA sequence of length 2,000 (same DNA length of the legitimate accounts), whose first 1,000 positions are filled with the DNA base A, followed by 500 positions filled with the base C and the last 500 positions filled with the base T. Regarding the mutation operator, the probability

[3] https://developer.twitter.com/en/docs
[4] https://www.uni-ulm.de/in/theo/research/seqana.html

to mutate each action is set equal to 0.0002 (MUT-PROB). For each generation simulated by GenBot, a total of 30 offsprings are generated via group-level crossovers. Concerning the user-level crossovers, for each individual 2 offsprings are generated via the URCO(·) operator (NUM-URCO), while 12 offsprings are generated via the UCO(·) operator (NUM-UCO). Each experiment is repeated 5 times and each run of the algorithm starts with a different random seed. Results are averaged across the 5 runs.

4.3 Experimental evaluation

In the next section, we provide results for our experiments. In each experiment, the quality of the solutions generated by the the GenBot algorithm is evaluated by comparing the LCS curve of the last generation of spambots with that of the reference group. The LCS curve of the last generation of spambots is computed by applying the point-to-point average of all the LCS curves of the spambots groups constituting the last generation. We give both a qualitative and a quantitative evaluation of the results, as follows: (i) we provide a graphical comparison of LCS curves, giving a direct and intuitive insight into the quality of our results; (ii) we compute and compare the AUC of the LCS curves with Equation (1); (iii) we measure the distance between the LCS curves by means of the D_{KL} defined in Equation (2).

5 RESULTS

The last generation of spambots generated by GenBot (i.e., the one under evaluation) is referred to as *evolved spambots*.

5.1 Behavioral analysis of evolved spambots

Here, we evaluate the extent to which the evolved spambots generated by GenBot are capable of emulating the behavior of legitimate users. The first – and to the best of our knowledge, unique – scientific attempt to solve this task was documented in [16], where the authors employed a set of resampling techniques to generate new behavioral fingerprints, as similar as possible to those of legitimate users. In order to provide a comparison between GenBot and [16], we applied the best performing techniques described in [16] to our dataset. Specifically, 3 types of DNA resampling techniques were used: (i) a statistical resampling of the digital DNA sequences of legitimate users based on the average characteristics of those users (labeled *average*); (ii) a block permutation with block size = 5 (labeled *5-permutation*); and (iii) a block bootstrap with block size = 5 (labeled *5-bootstrap*). For the sake of clarity, Figure 8 pictorially shows the way to perform a block resampling on a digital DNA sequence [16].

Qualitative results of this experiment are shown in Figure 5. The LCS curve of the group of legitimate accounts is labeled *benchmark*. As shown, the LCS curve of the spambots generated with GenBot is almost completely overlapping the benchmark. The only exception is in the tail of the LCS curve and it is visible in the log-log inset of Figure 5. The comparison between previous techniques and GenBot is clearly in favor of the latter. In fact, all previous techniques greatly struggle to fit the head of the LCS curve, while instead they perform better in the tail. To this regard, our approach and those described in [16] seem to be complementary, with our GenBot algorithm capable of fitting all the LCS curve except for the tail, which instead is well fit by the resampling approaches of [16]. One further improvement could be the adoption of block resampling strategies in GenBot, adding them to the mutations and crossovers that we already employ.

A quantitative evaluation of the quality of evolved spambots is obtained by comparing the AUC of the LCS curves of the spambots with that of the legitimate accounts. As shown in Table 1, the spambots obtained with GenBot were able to reproduce the LCS of legitimate users with a percentage error of only 2.98% in excess. Instead, all other techniques managed to reproduce only about half of the behavioral similarities expected within a group of legitimate accounts.

5.2 Detection of evolved spambots

Going further, we are now interested in evaluating whether the evolved spambots are able to avoid detection by state-of-the-art techniques [13]. We design this experiment by replicating the working conditions of most spambot detection systems – that is, we focus on the analysis of an unknown group of users that contains both spambots and legitimate users.

In detail, we start by mixing together part of our evolved spambots with part of the legitimate users. Then, we compare the LCS curve of the mixed group with that of the legitimate users only. Figure 6 shows a qualitative result of this comparison. As shown, the LCS curve of the mixed group lays very close to that of the legitimate users. In turn, this means that also the mixed group of evolved spambots and legitimate users still behaves like a group solely composed of legitimate users. Table 2 presents the comparison in terms of AUC values, which quantitatively confirm the result, although showing a larger error than that reported in the previous experiment of Table 1.

Finally, we apply 2 state-of-the-art spam and bot detection techniques [13, 35] to the mixed group, and we assess their performance in detecting the evolved spambots. We compare these results with those measured while applying the techniques in [13, 35] to a group of non-evolved spambots. Results are reported in Table 4 and show that the evolved spambots generated by GenBot largely evade detection (mean $F1 \simeq 0.260$). In addition to the techniques tested in Table 4, we also applied the system in [1] to our evolved spambots. Similarly to [13, 35], also [1] proves incapable of accurately detecting the evolved bots with $Accuracy = 0.495$ and $MCC = -0.071$. This result is in contrast with previous work [11, 13] where the detection rate for non-evolved spambots was $F1 = 0.923$.

5.3 Generalizability

In this section, we evaluate the generalizability of the GenBot algorithm and of the previously shown results. We change the group of legitimate accounts and we assess whether the evolved spambots generated by GenBot are still similar to the legitimate accounts. We first randomly split the original group of legitimate accounts into 2 disjunct subgroups (labeled *Group A* and *Group B*), each subgroup counting about 50% of the accounts of the original group. These subgroups are the 2 new references for GenBot to generate evolved spambots. Then, we compare the behavioral similarity between the evolved spambots and the subgroup (either *Group A* or *Group B*) used to generate them. Figure 7 shows the results of

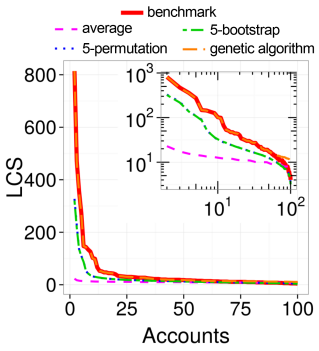

Figure 5: Qualitative analysis of evolved spambots and comparison with previous techniques.

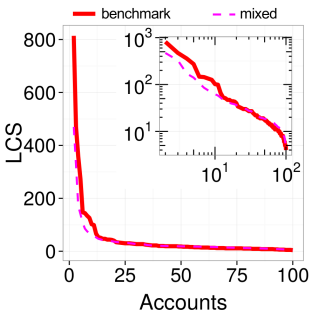

Figure 6: Comparison of the LCS curve of legitimate users with that of a mixed group composed of evolved spambots and legitimate users.

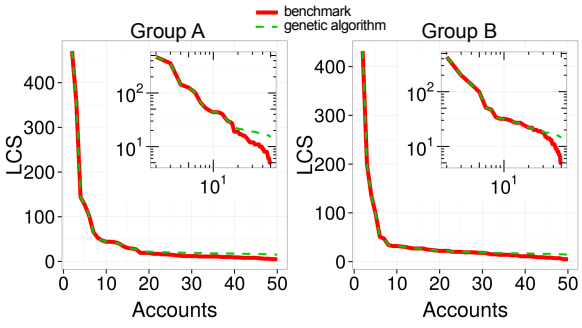

Figure 7: Qualitative comparison of evolved spambots against 2 different groups of legitimate accounts. Despite the different shape of the LCS curves of *Group A* and *Group B*, the corresponding evolved spambots closely match their behavior.

technique	AUC	% error
benchmark	3961.0	–
average	971.9	−75.46%
5-bootstrap	2001.7	−49.46%
5-permutation	2019.8	−49.01%
genetic algorithm	4079.0	+2.98%

Table 1: Quantitative analysis: comparison with previous techniques.

group	AUC	% error
benchmark (legitimate)	3961.0	–
mixed (bot + legitimate)	3090.2	−21.98%

Table 2: AUC comparison (group of legitimate users vs mixed groups: evolved spambots and legitimate users).

	Group A		*Group B*	
technique	AUC	% error	AUC	% error
benchmark	1797.0	–	1521.50	–
genetic algorithm	2025.6	+12.72%	1632.20	+07.28%

Table 3: Quantitative comparison of evolved spambots against 2 different groups of legitimate accounts.

		evaluation metrics					
technique	accounts	*Precision*	*Recall*	*Specificity*	*Accuracy*	*F1*	*MCC*
Cresci *et al.* [13]	non-evolved spambots	1.000	0.858	1.000	0.929	0.923	0.867
Miller *et al.* [35]	non-evolved spambots	0.555	0.358	0.698	0.526	0.435	0.059
Cresci *et al.* [13]	evolved spambots (GenBot)	0.512	0.210	0.800	0.505	0.298	0.012
Miller *et al.* [35]	evolved spambots (GenBot)	0.720	0.360	0.860	0.610	0.480	0.254

Table 4: Performances of 2 state-of-the-art spam and bot detection techniques towards the detection of non-evolved spambots and evolved spambots generated with GenBot. The evolved spambots largely go undetected.

	D_{KL}	
target group	mean	std
legitimate full	32.64	2.97
legitimate *Group A*	73.83	11.44
legitimate *Group B*	34.16	2.33

Table 5: Variability of our results across 5 runs of our algorithm for different experiments.

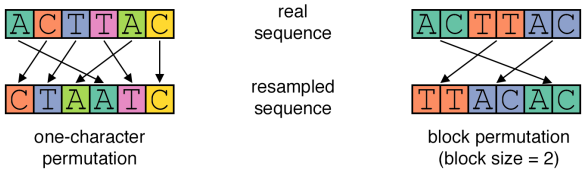

Figure 8: Application of block resampling to a digital DNA sequence, and comparison with one-character resampling.

a qualitative comparison. Despite the different shape of the LCS curves of *Group A* and *Group B* of legitimate accounts (solid red line), the corresponding evolved spambots closely match their behavior. This testifies that even by changing the characteristics of the accounts to mimic, GenBot is capable of generating spambots that behave in a similar way with respect to the legitimate ones.

Moreover, this also implies that GenBot is capable of generating spambots featuring different characteristics. Interestingly, Figure 7 also shows a rather poor performance in fitting the tail of the LCS curve, similarly to what we already saw in Figure 5. Finally, Table 3 reports quantitative results for the 2 subgroups of legitimate accounts, showing a low percentage error for both groups.

The experiments were executed 5 times, with random seeds to assess variability. In particular, we measure the mean and the standard deviation (std) of the distance between the LCS curve of the evolved spambots generated by GenBot and that of legitimate accounts, across 5 runs of the algorithm. The distance between LCS curves is measured by means of the D_{KL} distance defined in Equation (2). As shown in Table 5, the standard deviation of D_{KL} is rather low in every experiment, and significantly lower than the mean, which suggests low variability in the results.

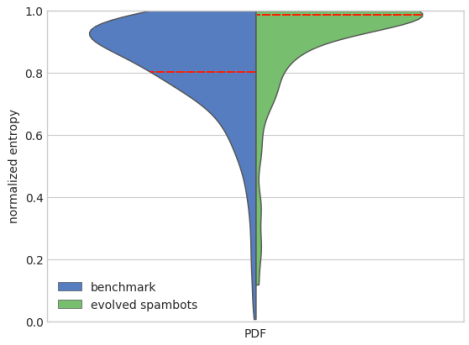

Figure 9: Beanplot showing the PDF of the normalized Shannon entropy of DNA sequences related to evolved spambots and legitimate accounts. DNA sequences of the evolved spambots feature a suspiciously high entropy.

5.4 Improving current detection techniques

As demonstrated above, the GenBot-generated spambots are able to avoid actual detection techniques. We can see that collectively, the evolved spambots do not leave traces of their automated nature, since they accurately reproduce the LCS curve of legitimate accounts. However, interesting insights can be gained by studying the single digital DNA sequences of every spambot. Indeed, a close inspection reveals that the digital DNA sequences of the spambots generated by GenBot have very few repetitions and almost no regularities at all[5]. In turn, such DNA sequences represent very erratic and heterogeneous behaviors. This finding is in contrast with the known characteristics of legitimate accounts [13, 16] that tend to favor certain actions in their behaviors. As a consequence, digital DNA sequences of legitimate accounts have a prevalence for certain DNA bases (e.g., the A base for tweets). We formalize this intuition by relying on the notion of normalized Shannon entropy (H_{norm}). In particular, we compute the entropy of each digital DNA sequence, for all the evolved spambots and all the legitimate accounts. Figure 9 shows a beanplot of the empirical probability density function (PDF) of the normalized entropy, comparing measurements for evolved spambots with those of legitimate accounts. As expected, the spambots generated by GenBot have mean $H_{norm} \simeq 1$, whereas for legitimate accounts mean $H_{norm} = 0.8$ (dashed red lines in Figure 9). A straightforward consequence of this observation is the possibility to extend current detection techniques based on the accounts' behavior, such as [11, 13], by also considering information related to the repetitions and regularities within sequences, thus making current systems more robust against possible future spambot evolutions. Although limited in scope, this experiment nonetheless testifies the usefulness – and one among many possible applications – of the proposed analytical framework for simulating spambot evolutions.

6 DISCUSSION

Our results demonstrated the possibility to combine the behavioral representation of digital DNA with the computational framework of genetic algorithms, in order to create evolved spambots capable of escaping current state-of-the-art detection techniques, based on the accounts' behavior [13] and on the content of posts [1, 35].

In particular, we designed an analytical framework for simulating spambot evolutions, thus answering to the first of our research questions (**RQ1**). In such framework, social spambots behavior is modeled via digital DNA. Then, DNA sequences are fed to the novel genetic algorithm (GenBot), designed to simulate spambot evolutions. After thousands of subsequent iterations, the output of GenBot is a novel generation of *evolved spambots*, described by their digital DNA. Notably, in this work we grounded the framework for simulating possible spambot evolutions on (i) genetic algorithms and (ii) the recent advances in digital DNA behavioral modeling, since they currently represent its key enabling factors. However, it is likely that in the near future the same methodological approach for studying spambot evolution could leverage different techniques and methodologies, thus widening its applicability.

Aiming to answer **RQ2**, we then evaluated the extent to which the evolved spambots are capable of going undetected by state-of-the-art techniques. Since our experiments grounded on modeling the actions in the timelines of the accounts under investigation, it was natural to consider detection techniques based on accounts behavior and posted contents. Thus, with regards to the technique in [13], we showed that (i) the behavioral fingerprint of the spambots generated by GenBot is similar to that of legitimate users, (ii) a group containing both our evolved spambots and legitimate users is almost indistinguishable from a group solely composed of legitimate users, and (iii) the *social fingerprinting* detection technique largely fails in detecting the evolved spambots. Moreover, 2 other recent detection techniques, based on the content of posts [1, 35], also fail in detecting the spambots generated by GenBot. These results raise concerns towards the vulnerabilities of current state-of-the-art techniques.

Finally, we studied the characteristics of the evolved spambots, with the goal of answering to **RQ3** – that is, looking for ways to improve (at least a subset of) current detection techniques. Specifically, we investigated whether the evolved spambots still had some peculiar characteristics that would make them detectable. We noticed that, although the group of evolved spambots behaves like a group of legitimate users, the digital DNA of the spambots is more *entropic* than that of legitimate users. Thus, we argue that it would be possible and fruitful to extend current detection techniques by also considering the amount of entropy within digital DNA sequences. This last finding thus represents a useful suggestion for improving current spambot detection techniques.

This study – the very first of its kind – moves in the direction of a *proactive* spambot detection. For the first time since the advent of OSNs, we have the chance to proactively study spambot evolutions and to design more robust detection techniques, possibly capable of withstanding the next evolutions of social spambots. Although unlikely to completely defeat spambots and other malicious accounts, the application of the proposed proactive approach would nonetheless bring groundbreaking benefits. The capability to foresee possible spambot evolutions would not only allow to test the detection rate of state-of-the-art techniques (including techniques based, e.g., on the exploration of the social graph of the accounts, or on their profiles and posting aptitudes), but also and above all,

[5]Here we are interested in repetitions *within* sequences rather than *across* sequences, as it was for the case of studying the LCS.

to *a priori* adapt them and even to design new detection techniques. As a consequence of the additional design and experimentation allowed by the proactive approach, many spambot evolutions will be detected from *day 0*. Spambots will see their chances to harm severely restricted, with clear and immediate benefits for our online environments, and ultimately, for our societies (e.g., less fake news and biased propaganda). Notably, for those few spambot evolutions still not foreseen by this proactive approach, we will still be able to fall back to the traditional reactive approach, at no additional cost.

7 ETHICAL CONSIDERATIONS

With the rise of AI, our daily lives are increasingly influenced by decisions taken on our behalf by automated systems. Algorithmic filtering (which leads to filter bubbles, echo chambers, and eventually polarization), algorithmic bias and current limits in explainability of predictive models already raise serious ethical concerns on the development and adoption of AI solutions. Within this context, algorithmic approaches to the characterization, development, and detection of social bots make no exception [17, 42]. For instance, one might naively think that all endeavors devoted to the development of social bots are to be blamed. However, not all social bots are nefarious by nature [20]. Indeed, bots can be programmed to automatically post information about news and academic papers [24, 32], and even to provide help during emergencies [4, 39]. Undeniably, the provision of useful services by benign bots will make them become an established presence on social platforms [37]. Meanwhile, other researchers investigated malicious bots [2, 15, 29, 33], in an effort to better understand their evolution, impact and interactions with benign users. The so-recognized existence of benign versus malicious bots sparked heated debates about the rights of automated accounts. For instance, both researchers and everyday social media users wondered the extent to which social bots should be considered equal to humans, as far as censorship[6,7] and suppression of free speech [34] are concerned.

While advancing the state of the art in the fascinating field of AI, we are aware that the successful implementation of new technologies will pose greater challenges to discriminate between human and automated behaviors. Now more than ever, spambot evolution and their subsequent detection meet ethical considerations. The framework proposed in this paper should not be seen merely as a technical exercise. In fact, our evolved spambots have not been conceived to support botnet developers (a criticism that could be very well posed to all the above-cited research). Instead, we remark that one of the main goals of this paper is to proactively sharpen detection techniques to cope with future evolutions of spambots, as typically done in the well-recognized field of adversarial learning.

Lastly, [50] clearly explains that "a supervised machine learning tool is only as good as the data used for its training". Since spambot detection is a rapidly-evolving field, we are all involved in the quest for up-to-date datasets. By playing with the parameters of the evolutionary algorithm here proposed, we advocate the capability to create a huge variety of fresh data, to re-train and fine-tune existing detection mechanisms.

[6]https://www.nytimes.com/2018/09/05/technology/lawmakers-facebook-twitter-foreign-influence-hearing.html
[7]https://www.theguardian.com/technology/2018/oct/16/facebook-political-activism-pages-inauthentic-behavior-censorship

8 RELATED WORK

Although representing an effective heuristic to solve complex optimization problems [36], genetic algorithms usually require a string-based genetic encoding of information to be applied. This requirement severely limited their applicability. Remarkably, in recent years, we assisted to the proliferation of many studies on modeling and analyzing online behaviors. A stream of research focused on specific behavioral analytics tasks, such as detecting specific behavioral patterns [6, 27, 38], predicting future behaviors [30, 55], and detecting anomalous ones [7, 11, 13, 45, 53, 54]. Others instead achieved more general results. In [25] authors showed that individuals have persistent and distinct online inter-event time distributions, while [16] focused on modeling human tweeting behaviors, showing that such behaviors are very diverse and heterogeneous, although far from being random. One result achieved in behavioral analytics is the possibility to encode the behavioral information of an account in a DNA-like string of characters [11, 16]. The characterization of the account behavior through this *digital DNA* modeling technique, coupled with the capability to carry out evolutionary simulations by means of genetic algorithms, opens up the unprecedented opportunity to quantitatively experiment with spambot evolutions and motivates our research.

Meanwhile, progress has been made towards the detection of malicious accounts (e.g., fakes, bots, spammers). As such accounts put in place complex mechanisms to evade existing detection systems, scholars tried to keep pace by proposing powerful techniques based on profile- [5, 10, 54], posting- [5, 8, 11, 13, 22, 48], and network-characteristics [5, 31, 47, 48, 51–53] of the accounts. However, until now, new detection systems have been developed only as a consequence of spambot evolutions [12]. In fact, no work has ever been done towards studying, and possibly anticipating, such evolutions. In other words, malicious accounts detection has always been tackled with a *reactive* approach, which is in contrast with the novel *proactive* approach envisaged in this research.

9 CONCLUSIONS

We presented the first exploratory study to carry out a quantitative analysis of spambot evolutions. Specifically, riding the wave of the adversarial learning line of research, we first designed a novel genetic algorithm for simulating spambot behavioral evolutions. Then, we evaluated the extent to which the evolved spambots are capable of evading 3 state-of-the-art detection systems, based on evaluating the accounts' behavior and the content of posts. Testing the first system, results showed that as much as 79% of the evolved spambots evade detection. Additionally, we also investigated the characteristics of the evolved spambots, highlighting distinctive features (e.g., entropy within digital DNA sequences) that would allow to distinguish them from legitimate accounts. Considering these features in current detection systems based on behavioral characteristics of the accounts would make them more robust against possible future spambot evolutions.

Further experimentation could lead to new results not only in bot design (e.g., chatbots), but also and foremost in the development of spambot detection systems. Indeed, until now, researchers had to *react* to bot evolutions. However, for the first time since the advent of OSNs, there is the concrete chance to *proactively* tackle the

challenging task of spambot detection. Although here instantiated for a specific modeling and detection technique, we thus argue that the proactive approach will provide the scientific community with the possibility to experiment with and simulate future spambot evolutions, substantially raising the bar for spambot developers.

ACKNOWLEDGEMENTS

Research supported in part by TOFFEe (TOol for Fighting FakEs), a PAI ('Progetto di Attività Integrate') project of IMT School For Advanced Studies Lucca and by MIUR under grant 'Dipartimenti di eccellenza 2018-2022', Computer Science Dept., Sapienza University.

REFERENCES

[1] Faraz Ahmed and Muhammad Abulaish. 2013. A generic statistical approach for spam detection in Online Social Networks. *Computer Communications* 36, 10 (2013), 1120–1129.
[2] Luca Maria Aiello, Martina Deplano, Rossano Schifanella, and Giancarlo Ruffo. 2012. People are Strange when you're a Stranger: Impact and Influence of Bots on Social Networks. In *ICWSM*. AAAI.
[3] Michael Arnold and Enno Ohlebusch. 2011. Linear time algorithms for generalizations of the longest common substring problem. *Algorithmica* 60, 4 (2011).
[4] Marco Avvenuti, Salvatore Bellomo, Stefano Cresci, Mariantonietta Noemi La Polla, and Maurizio Tesconi. 2017. Hybrid crowdsensing: A novel paradigm to combine the strengths of opportunistic and participatory crowdsensing. In *WWW Companion*. ACM.
[5] Prudhvi Ratna Badri Satya, Kyumin Lee, Dongwon Lee, Thanh Tran, and Jason Jiasheng Zhang. 2016. Uncovering fake likers in online social networks. In *CIKM*. ACM.
[6] Hakan Bagci and Pinar Karagoz. 2015. Random walk based context-aware activity recommendation for location based social networks. In *DSAA*. IEEE.
[7] Chiyu Cai, Linjing Li, and Daniel Zeng. 2017. Detecting Social Bots by Jointly Modeling Deep Behavior and Content Information. In *CIKM*. ACM.
[8] Nikan Chavoshi, Hossein Hamooni, and Abdullah Mueen. 2016. DeBot: Twitter Bot Detection via Warped Correlation.. In *ICDM*. IEEE.
[9] Lucas Chi and Kwong Hui. 1992. Color set size problem with applications to string matching. In *Combinatorial Pattern Matching*. Springer.
[10] Stefano Cresci, Roberto Di Pietro, Marinella Petrocchi, Angelo Spognardi, and Maurizio Tesconi. 2015. Fame for sale: Efficient detection of fake Twitter followers. *Decision Support Systems* 80 (2015).
[11] Stefano Cresci, Roberto Di Pietro, Marinella Petrocchi, Angelo Spognardi, and Maurizio Tesconi. 2016. DNA-inspired online behavioral modeling and its application to spambot detection. *IEEE Intelligent Systems* 31, 5 (2016).
[12] Stefano Cresci, Roberto Di Pietro, Marinella Petrocchi, Angelo Spognardi, and Maurizio Tesconi. 2017. The Paradigm-Shift of Social Spambots: Evidence, Theories, and Tools for the Arms Race. In *WWW Companion*. ACM.
[13] Stefano Cresci, Roberto Di Pietro, Marinella Petrocchi, Angelo Spognardi, and Maurizio Tesconi. 2017. Social fingerprinting: detection of spambot groups through DNA-inspired behavioral modeling. *IEEE TDSC* (2017).
[14] Stefano Cresci, Fabrizio Lillo, Daniele Regoli, Serena Tardelli, and Maurizio Tesconi. 2018. $FAKE: Evidence of spam and bot activity in stock microblogs on Twitter. In *ICWSM*. AAAI.
[15] Stefano Cresci, Marinella Petrocchi, Angelo Spognardi, and Stefano Tognazzi. 2019. On the capability of evolved spambots to evade detection via genetic engineering. *Online Social Networks and Media* 9 (2019).
[16] Stefano Cresci, Roberto Di Pietro, Marinella Petrocchi, Angelo Spognardi, and Maurizio Tesconi. 2017. Exploiting digital DNA for the analysis of similarities in Twitter behaviours. In *DSAA*. IEEE.
[17] Carolina Alves de Lima Salge and Nicholas Berente. 2017. Is that social bot behaving unethically? *Commun. ACM* 60, 9 (2017).
[18] Stefan Droste, Thomas Jansen, and Ingo Wegener. 2002. On the Analysis of the (1+1) Evolutionary Algorithm. *Theoretical Computer Science* 276 (2002).
[19] Tom Fawcett. 2006. An introduction to ROC analysis. *Pattern recognition letters* 27, 8 (2006).
[20] Emilio Ferrara, Onur Varol, Clayton Davis, Filippo Menczer, and Alessandro Flammini. 2016. The rise of social bots. *Commun. ACM* 59, 7 (2016).
[21] Jerome Friedman, Trevor Hastie, and Robert Tibshirani. 2001. *The elements of statistical learning*. Vol. 1. Springer Series in Statistics.
[22] Maria Giatsoglou, Despoina Chatzakou, Neil Shah, Alex Beutel, Christos Faloutsos, and Athena Vakali. 2015. ND-Sync: Detecting Synchronized Fraud Activities. In *PAKDD*.
[23] Dan Gusfield. 1997. *Algorithms on strings, trees and sequences: computer science and computational biology*. Cambridge University Press.
[24] Stefanie Haustein, Timothy D. Bowman, Kim Holmberg, Andrew Tsou, Cassidy R. Sugimoto, and Vincent Larivière. 2016. Tweets As Impact Indicators: Examining the Implications of Automated "Bot" Accounts on Twitter. *J. Assoc. Inf. Sci. Technol.* 67, 1 (2016).
[25] Jiwan Jeong and Sue Moon. 2017. Interval Signature: Persistence and Distinctiveness of Inter-event Time Distributions in Online Human Behavior. In *WWW Companion*. ACM.
[26] Ari Juels and Martin Wattenberg. 1996. Stochastic hillclimbing as a baseline method for evaluating genetic algorithms. In *NIPS*.
[27] Srijan Kumar, Justin Cheng, and Jure Leskovec. 2017. Antisocial Behavior on the Web: Characterization and Detection. In *WWW Companion*. ACM.
[28] Alexey Kurakin, Ian Goodfellow, and Samy Bengio. 2017. Adversarial machine learning at scale. In *ICLR*.
[29] Kyumin Lee, Brian Eoff, and James Caverlee. 2011. Seven Months with the Devils: A Long-Term Study of Content Polluters on Twitter. In *ICWSM*. AAAI.
[30] Kang Li and Yun Fu. 2014. Prediction of human activity by discovering temporal sequence patterns. *IEEE TPAMI* 36, 8 (2014).
[31] Shenghua Liu, Bryan Hooi, and Christos Faloutsos. 2017. HoloScope: Topology-and-Spike Aware Fraud Detection. In *CIKM*. ACM.
[32] Tetyana Lokot and Nicholas Diakopoulos. 2016. News Bots: Automating news and information dissemination on Twitter. *Digital Journalism* 4, 6 (2016).
[33] Gregory Maus. 2017. A Typology of Socialbots (Abbrev.). In *WebSci*. ACM.
[34] Jeffrey Mervis. 2014. An Internet research project draws conservative ire. *Science* 346, 6210 (2014).
[35] Zachary Miller, Brian Dickinson, William Deitrick, Wei Hu, and Alex Hai Wang. 2014. Twitter spammer detection using data stream clustering. *Information Sciences* 260 (2014).
[36] Melanie Mitchell. 1998. *An Introduction to Genetic Algorithms*. MIT Press.
[37] Bjarke Monsted, Piotr Sapiezynski, Emilio Ferrara, and Sune Lehmann. 2017. Evidence of complex contagion of information in social media: An experiment using Twitter bots. *PloS one* 12, 9 (2017).
[38] Marcel Salathé, Duy Q Vu, Shashank Khandelwal, and David R Hunter. 2013. The dynamics of health behavior sentiments on a large online social network. *EPJ Data Science* 2, 1 (2013).
[39] Saiph Savage, Andres Monroy-Hernandez, and Tobias Höllerer. 2016. Botivist: Calling volunteers to action using online bots. In *CSCW*. ACM.
[40] Indira Sen, Anupama Aggarwal, Shiven Mian, Siddharth Singh, Ponnurangam Kumaraguru, and Anwitaman Datta. 2018. Worth its Weight in Likes: Towards Detecting Fake Likes on Instagram. In *WebSci*. ACM.
[41] L Steward, Ahmer Arif, and Kate Starbird. 2018. Examining Trolls and Polarization with a Retweet Network. In *WSDM Workshops*. ACM.
[42] Andree Thieltges, Florian Schmidt, and Simon Hegelich. 2016. The devil's triangle: Ethical considerations on developing bot detection methods. In *Spring Symposium Series*. AAAI.
[43] Stefano Tognazzi, Stefano Cresci, Marinella Petrocchi, and Angelo Spognardi. 2018. From Reaction to Proaction: Unexplored Ways to the Detection of Evolving Spambots. In *WWW Companion*. ACM.
[44] Liang Tong, Bo Li, Chen Hajaj, and Yevgeniy Vorobeychik. 2017. Hardening Classifiers against Evasion: the Good, the Bad, and the Ugly. *CoRR* abs/1708.08327 (2017). arXiv:1708.08327 http://arxiv.org/abs/1708.08327
[45] Onur Varol, Emilio Ferrara, Clayton Davis, Filippo Menczer, and Alessandro Flammini. 2017. Online Human-Bot Interactions: Detection, Estimation, and Characterization. In *ICWSM*. AAAI.
[46] Bimal Viswanath, Muhammad Ahmad Bashir, Muhammad Bilal Zafar, Simon Bouget, Saikat Guha, Krishna P Gummadi, Aniket Kate, and Alan Mislove. 2015. Strength in Numbers: Robust Tamper Detection in Crowd Computations. In *COSN*. ACM.
[47] Binghui Wang, Neil Zhenqiang Gong, and Hao Fu. 2017. GANG: Detecting Fraudulent Users in Online Social Networks via Guilt-by-Association on Directed Graphs. In *ICDM*. IEEE.
[48] Fangzhao Wu, Jinyun Shu, Yongfeng Huang, and Zhigang Yuan. 2015. Social spammer and spam message co-detection in microblogging with social context regularization. In *CIKM*. ACM.
[49] Chao Yang, Robert Harkreader, and Guofei Gu. 2013. Empirical evaluation and new design for fighting evolving Twitter spammers. *IEEE TIFS* 8, 8 (2013).
[50] K. C Yang, O. Varol, C. A. Davis, E. Ferrara, A. Flammini, and F. Menczer. 2019. Arming the public with AI to counter social bots. *Human Behavior and Emerging Technologies* (2019).
[51] Zhi Yang, Christo Wilson, Xiao Wang, Tingting Gao, Ben Y Zhao, and Yafei Dai. 2014. Uncovering social network sybils in the wild. *ACM TKDD* 8, 1 (2014).
[52] Rose Yu, Xinran He, and Yan Liu. 2014. GLAD: group anomaly detection in social media analysis. In *KDD*. ACM.
[53] Shuhan Yuan, Xintao Wu, Jun Li, and Aidong Lu. 2017. Spectrum-based deep neural networks for fraud detection. In *CIKM*. ACM.
[54] Reza Zafarani and Huan Liu. 2015. 10 Bits of Surprise: Detecting malicious users with minimum information. In *CIKM*. ACM.
[55] Aoying Zhou, Weining Qian, and Haixin Ma. 2012. Social media data analysis for revealing collective behaviors. In *KDD*. ACM.