A Multi-agent Approach for Online Twitter Bot Detection

Conference Paper · June 2021

DOI: 10.18239/jornadas_2021.34.03

CITATIONS
0

READS
168

3 authors:

Jefferson Viana Fonseca Abreu
Virginia Polytechnic Institute and State University
6 PUBLICATIONS **6** CITATIONS
SEE PROFILE

Célia Ghedini Ralha
University of Brasília
145 PUBLICATIONS **582** CITATIONS
SEE PROFILE

Joao Gondim
University of Brasília
31 PUBLICATIONS **150** CITATIONS
SEE PROFILE

http://doi.org/10.18239/jornadas_2021.34.03

A Multi-agent Approach for Online Twitter Bot Detection

Jefferson Viana Fonseca Abreu
Computer Science Departament
University of Brasília
Brasília, DF, Brazil
Email: jeffvfa@hotmail.com
ORCID: 0000-0001-6609-5338

Célia Ghedini Ralha
Computer Science Departament
University of Brasília
Brasília, DF, Brazil
Email: ghedini@unb.br
ORCID: 0000-0002-2983-2180

João José Costa Gondim
Computer Science Departament
University of Brasília
Brasília, DF, Brazil
Email: gondim@unb.br
ORCID: 0000-0002-5873-7502

***Abstract*—Online social networks are tools that allow interaction between human beings with a large number of users. Platforms like Twitter present the problem of social bots which are controlled by automated agents potentially used for malicious activities. Thus, social bot detection is important to keep people safe from harmful effects. In this work, we approach the Twitter bot online detection problem with a multi-agent system (MAS). It is based on supervised classification with three machine learning algorithms and a reduced set of features. The MAS performance compared to the three algorithms applied separately - Random Forest, Support Vector Machine, and Naïve Bayes - presented similar results. Besides, interesting results for online bot detection with the MAS prototype suggested that 88.19% of bots detected were correctly labeled. The results indicate that the approach used is feasible and promising for the real-time bot detection problem.**

***Keywords*—Bot detection, Social bots, Twitter, Agents, Multi-Agent System, MAS.**

1. Introduction

Online social networks (OSN) are tools designed to facilitate interaction between human beings. Due to a large number of OSN users, it is natural that some individuals might be willing to carry out malicious actions seeking to gain undue advantage, the weakest link in information security [1], [2]. Social bots (hereinafter referred to only as bots), OSN profiles that impersonate human beings through automated interactions [3], are one of the methods used to carry out these abuses. These bots can be grouped into botnets acting in a coordinated manner, enabling more powerful and harmful attacks [4].

Bots can be used for malicious tasks such as: sharing spam [5], [6], [7], [8], vectors for phishing [3], [9] or spread of news [10]. This issue is very current in the Brazilian scenario after the 2018 elections, where there was a great prominence in electoral campaigns linked to the internet, where OSN provided a favorable scenario for the use of bots. The suspicion of the use of this device during the electoral period, and after it, culminated in the establishment of a Joint Parliamentary Commission of Inquiry (CPMI) at the Brazilian national congress to investigate, among other issues, the use of bots to influence the results of the 2018 elections [11].

Among the most well-known OSN, it is possible to highlight Twitter. The relevance of this social media is so great, that several notable people and heads of state adopt this platform as one of their main media. In the literature, several works have as one of their objectives the classification of bots on Twitter. In [12], bot classification approaches permeating three different classes are identified: using machine learning (ML) algorithms; based on graphs and emergent approaches; and one which encompasses techniques that do not fit into the other two categories. All the works listed in [12] were studied along with more recent publications. In [13] there is the conception of a reduced set of features that allows the detection of bots with quality comparable to one of the works that are state of the art Twitter bot detection [14].

Despite the thorough review of the literature, no work was identified involving the detection of bots using an approach based on intelligent agents (hereinafter treated only by agents). This fact represents an interesting research opportunity because the use of this approach can be very useful in this field of application. Agents are entities capable of performing activities in an automated, persistent manner, and having the ability to adapt to different contexts [15]. These are desirable attributes when it is envisaged to enhance the detection of bots for Twitter.

Thus, the main objective of this work is to develop a multi-agent system (MAS) capable of autonomously detect bots on Twitter. The MAS design phase uses Tropos methodology. The MAS applies a supervised classification approach with three ML algorithms using and a reduced set of features (presented previously in [13]). The contributions of this work includes:

- a bot classifier using a MAS approach that combines three ML algorithms presenting good performance with the average of AUC equal to 0.9856 and standard deviation of 0.0199; and
- proof of concept with the MAS capturing and clas-

sifying users provided by an online stream of tweets where 88.19% of bots detected were correctly labeled.

This paper is organized as follows: Section 2 presents the preliminary concepts that are needed to fully understand our work, Section 3 contains the works that were found in the literature and are most related to this research, Section 4 explains the methodology followed, Section 5 shows the results achieved and discusses them, Section 6 are presented the final considerations and the future work.

2. Preliminaries

In this section, some concepts applied in the work are presented, including intelligent agents and MAS design.

2.1. Intelligent Agents

According to [16] computational intelligence is the area of study that deals with the development of intelligent agents. [15] defines an agent as an entity that acts, but a computational agent is expected to do more like: operate under autonomous control, perceive its environment, persist for an extended period, adapt to changes, and be able to create and achieve goals. As all computer programs are developed to process something, we can differentiate an agent by the ability to perceive its environment with its sensors and act on it using its actuators.

According to [17], [18], a MAS is composed of a set of agents capable of interacting with the environment and with each other, cooperatively or competitively, in pursuit of achieving one or more individual or collective objectives. The agents of a MAS can play several roles to achieve their objectives, either through the independent execution of actions, adapting to changes in the environment, and interacting with other agents. A MAS project needs the modeling phase to define the behavior and reasoning of each agent, the definition of a communication and interaction protocol for the group of agents, as well as which tools will be used in the implementation of the system.

According to [15], a rational agent needs to perform actions seeking to achieve the best possible result using the set of available information. Acting rationally is an action that may involve the treatment of uncertainties, logical inferences, and reflexes. Agents can be classified according to their performance in the environment through their behavior at various levels of complexity, from the simplest to the most complex including simple reactive agent, model-based reactive agent, objective based agent, utility based agent, and learning agent. All types of agents can be extended using ML techniques.

2.2. MAS Design

An essential element in the design of MAS is the correct characterization of the environment in which agents are inserted. According to [15], [17], [18] the environment must be characterized according to six essential aspects: observable, deterministic, episodic, static, discrete, and multi-agent.

For an adequate definition of the task environment of the rational agent, [15] defined PEAS (Performance measure, Environment, Actuators, and Sensors). In the MAS design pre-project phase, the aspects of performance measurement, environment, perceptions, and actions for each agent should be specified as detailed as possible. The performance measure is not fixed for all agent tasks, it is defined by the designer and must be able to assess the agent's behavior in a specific environment. For each sequence of perceptions of the rational agent, the designer must select an action that will maximize the defined performance measure, given the evidence provided by the sequence of perceptions and any internal knowledge of the agent or received from other agents.

In this work, we applied the modeling methodology called Tropos to design agent oriented software systems. According to [19], Tropos is based on two key ideas. First, the notion of agent and all related mentalistic notions (for instance goals and plans) are used in all phases of software development, from early analysis down to the actual implementation. Second, Tropos covers also the very early phases of requirements analysis, thus allowing for a deeper understanding of the environment where the software must operate, and of the kind of interactions that should occur between software and human agents. The development of a project using Tropos is divided into five phases: initial requirements, late requirements, architectural design, detailed design, and implementation. For the initial modeling phases, the language i* (iStar) is adopted, which has a focus on the intentional (why?), social (who?) and strategic (how?) dimensions of the software [20].

3. Related Work

Considering the Twiter bot detection works available in the literature, we present some recent ones that achieved interesting results comparable to our proposal. In Table 1 the related work comparison is presented.

In [14] the idea of classifiers with only one class trained with data from legitimate users is explored obtaining a tool capable of detecting any type of bot. To demonstrate this thesis several ML algorithms were investigated: the Bayesian networks, J48, RF, Adaboost, bagging, k-nearest neighbors. The authors identified the most used multiclass algorithms in the literature review performed by them include the logistic regression, multilayer perceptron, Naïve Bayes (NB), and Support Vector Machine (SVM). The tested class algorithms were bagging-TPMiner, bagging-random miner, one-class k-means with randomly-projected features algorithm, one-class SVM, and NB. The classifiers were trained and tested offline with the public datasets of [23], [24]. As a result, there is a consistent classification of bots of different types without needing any prior information, with an AUC of 0.89.

In [4] a classifier is developed to detect retweeting bots. The method consisted of taking all the retweets during a time

TABLE 1. RELATED WORK COMPARISON

Reference	ML algorithm	Application Domain	Method
Rodríguez-Ruiz et al. (2020) [14]	BN, J48, RF, Adaboost, Bagging, KNN, LR, MLP, NB, SVM, BTPM, BRM, OCKRA, ocSVM	general	offline supervised classification
Mazza et al. (2019) [4]	LSTM	retweet botnets	online unsupervised classification
Begenilmis & Uskudarli (2018) [21]	RF, LR, SVM	election campaign	offline supervised classification
Varol et al. (2017) [22]	RF	general	offline supervised classification, with online data

window to analyze the retweets pattern for each account during the period. A data visualization technique, called ReTweet-Tweet, was developed, which found four patterns of retweets, one for human users and three for bots. When conceiving the classifier Retweet-Buster (RTBust) the unsupervised algorithm Long Short-Term Memory (LSTM) was applied. With 12 features, a F1-score = 0.87 was obtained better than those in [22], [25], [26].

In [21] an organized behavior classifier is built in Twitter. The period chosen here is the 2016 American elections, as it is believed that there was a high volume of propaganda spread and fake news using OSN. Three different classifiers were trained and tested offline: RF, LR and SVM. The algorithms classified the accounts as organized or organic, political or non-political, pro Trump or pro Hillary. The RF algorithm performed better than the others with an average accuracy and F1-score greater than 95% in each category. The source code and the datasets used are available in a public repository at GitHub. The main contribution of this work is the design of a basic model for the classification of organized behavior on Twitter.

In [22] a report is made about the classifier *bot or not* (called today as botometer). The bots classification is done using Random Forest (RF), where more than a thousand different Twiter account attributes are used to perform the classification. A dataset containing only bots obtained through a honeypot [27] and a dataset manually annotated by the researchers are used. Both datasets are then mixed to build the dataset used in the article that is available to the community. The authors also maintain a web tool that can classify any profile on Twitter in real-time. The measure of the accuracy of implementation is given by the Area under the ROC Curve (AUC) with a score of 0.95. Authors realized that user metadata and tweet content are the most important attributes to find out if the account is a bot or not. Also, it is shown that accounts controlled by software can be grouped according to their intentions or *modus operandi*.

In a previous work [13], a set of five features is defined and four classifiers - RF, SVM, NB, and one class SVM (ocSVM) - were implemented to use this reduced set of features. The performance of the algorithms is compared to the state-of-the-art bot detection work presented in [14]. The classifiers were trained and tested offline with the public datasets of [23], [24]. The accuracy of the classifier was considered homogeneous with an average of 0.8549 and 0.1889 of standard deviation. Also, all multiclass classifiers achieved AUC greater than 0.9, indicating a practical benefit for bot detection on Twitter.

4. Methodology

As presented in Section 1, the objective of this work is to develop a MAS that performs the detection of bots for Twitter automatically. In the MAS design, we used the Tropos methodology. The choice of this methodology is motivated mainly by the fact that it covers several different phases of the MAS project, which allows a greater understanding of the system and a more refined set of requirements. In this section, the development stages performed during the work will be reported.

4.1. Design Model

In the pre-project phase of the MAS design, we define the PEAS (see Section 2.2). In this work, the environment that the MAS will interact is Twitter. It consists of a microblog where users post short messages (hereinafter referred to as tweets) in their respective profiles [28]. According to [29], the tweets are up to 280 text characters long and may contain references to other profiles by typing @ and the username to be mentioned. Other multimedia elements are also possible, such as videos, images, surveys, geolocation, among others [30]. Thus, the environment shared by agents in the MAS can be defined as:

- partially observable: impossible to view the complete set of all tweets;
- non-deterministic: it is not possible to determine the contents of the tweet in advance;
- episodic: tweets can be treated loosely connected in episodes;
- dynamic: new tweets may appear at any time;
- continuous: tweets may appear in a continuous spectrum of values; and
- multi-agent: more than one agent can perceive and act in the environment.

Three types of agents were defined in the MAS. The first type is the collector which is responsible for identifying the profiles that are making tweets according to a pre-established domain, saving the respective data in a monitoring database. As soon as data is identified, the collector forwards it to the classifying agents. In turn, the classifiers perform the classification of the profiles and send the results to the referee. The training of the classifiers is carried

TABLE 2. PEAS FOR THE THREE AGENTS: COLLECTOR, CLASSIFIER AND REFEREE.

Agent Type	Performance Measure	Actuators	Sensors
Colletor	Number of collected tweets	Monitor tweets according to key-words, Collect information about tweets, Record information in database, Send data to classifiers.	Verifies the existence of new tweets in the tweet flow chosen
Classifier	Number of executed classifications	Receive tweet data from collectors, Extract the necessary features, Classify the user who posted the tweet.	Verifies the existence of new data from tweets collected by collectors
Referee	Number of accounts classified as bots	Count *votes* and decide about the classification.	Verifies the existence of new classification from classifier agents

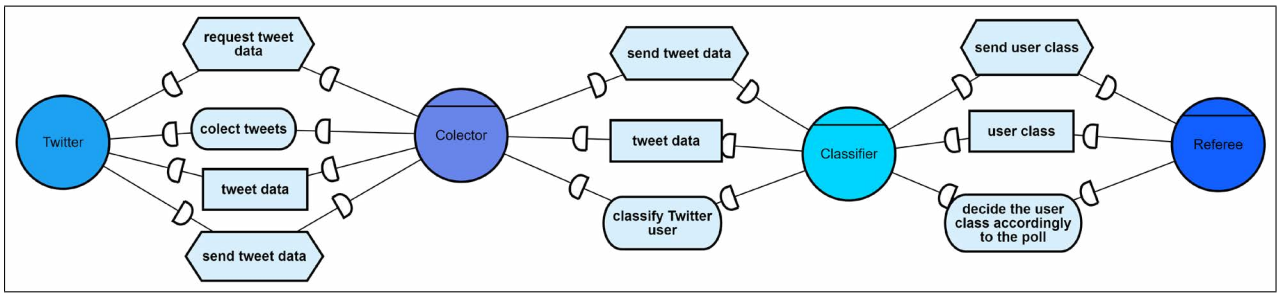

Figure 1. Late requirements diagram.

out offline, making its use faster during the execution of the system. Each classifying agent is implemented with a different ML algorithm, so they can classify the same profile into different classes. The choice of the approach using supervised ML algorithms was based on [12], where it was possible to glimpse the most widely used algorithms for the Twiter bot application domain.

The referee agent is responsible for making the final classification of the profile, according to the results of the classifying agents through a simple majority vote. In other words, the referee joins the decisions resulting in a group learning approach and records these decisions in a comma-separated values (csv) file. The experiment is only interrupted by the researchers' decision. The PEAS of the three agent types is available in Table 2. Since the environment is common to all agents, it is not included in the PEAS table.

During the MAS design project, four different models were built referring to development phases. The models designed include the initial requirements, late requirements, architectural design, and detailed design. Due to the adoption of the Tropos [19] methodology, the models were built using the language i* (iStar) [20]. To build the models, the online tool piStar implemented by [31] was used. Figure 1 illustrates the late requirements diagram for the MAS project, showing the relationships between actors, agents, tasks, resources, and objectives.

4.2. Agent Reasoning Model

Three types of intelligent agents were defined for the MAS. Each type has different properties and reasoning models. The reasoning model of each agent was chosen according to the actions that they would perform as presented in the PEAS definition of Table 2.

The first type of agent is the collector with the reactive reasoning model. This is the simplest type of reasoning without a sophisticated strategy but the choice of action that will be performed is based on the perception-action mapping. This reasoning model was chosen because the activity to be performed is trivial since the agent is responsible for accessing Twitter to collect tweets based on a set of pre-defined keywords and passing the tweet data to the classifying agents.

The classifier agent is defined as a learning agent. This agent is responsible for receiving data from the collector agent, applying a feature selection routine, and performing a data classification using a ML algorithm. This classification will be the agent's *vote* to assist the referee's decision. In the first MAS implementation, three classifying agents were built using one of the three most common algorithms in this application domain: RF, SVM, and NB. The same classifiers previously used in [13] intending to allow comparison with the MAS results.

Finally, the referee is an objective-oriented agent. Such agents consider, among the universe of actions to be taken, which is the best one to achieve their objectives. The referee is responsible for counting the classifier agents' *votes* referring to a Twitter profile. The referee's decision will allow reporting the profile to Twitter and monitor whether the profile has been banned.

4.3. Architecture

The MAS architecture is based on the reactive agent model that is horizontally layered as presented in [32]. The architecture layers connect directly the sensors (input) and the actuators (output). There are three layers and m possible actions suggested by each layer resulting in m^3 possible

interactions. The three layers are divided by well-defined activities and each has homogeneous agents. The activities include tweets capture, the tweets' authors classification, and arbitration. In other words, the architecture is horizontalized distributed into competency modules. Note that the architecture is naturally prepared to accommodate the online detection, since it establishes a direct pipeline for processing and classifying tweets as they are captured by the agents.

One of the facts that support the architecture choice is the high fault tolerance inherent in the model, failures that can occur at any of the levels. As the MAS aims to deal with a dynamic environment this property is interesting. Besides, the layered scheme is a good choice when we have several agents with different capabilities interacting in the same environment. Adversity faced in this architectural model is the bottleneck introduced by the communication between MAS and the environment since there are unique points of communication for both input and output. However, the positive points mentioned compensating for this negative characteristic for the time being. Also, the proposed architecture does not have many layers, which helps to soften the bottleneck. In Figure 2 we present a diagram of the proposed MAS architecture. It is possible to visualize the agent types and how they interact with each other.

4.4. Implementation

After the artifacts described, the implementation of the MAS was carried out. We used the language Python, version 3 and the framework for agent-oriented development called Python Agent DEvelopment framework (PADE) developed by [33]. The source code is open source and is available in a public repository of the research group at GitLab (https://gitlab.com/InfoKnow/SocialNetwork/jeffersonabreu-twitterbotdetection). The three proposed agents were developed and the interactions between them occur using communication protocols standardized by the Foundation for Intelligent Physical Agents (FIPA) [34].

The communication between the collector and the classifier agents occurs through the FIPA Agent Communication Language (ACL) with an interaction protocol called subscribe [35]. In this protocol, the agent's subscribers perform a subscription to an agent publisher, and after that, all subscribers receive notifications sent by the publisher. The role of subscribers while the collecting agent acts as a publisher. We believe this interaction protocol implements the desired behavior for communication between classifiers and the collector, through a simple message exchange using the FIPA inform performative.

For training and testing the MAS, the dataset used is provided by [23]. It is composed of several files in the csv format containing information related to user profiles and tweets. Each file is flagged with the type of profile produced by the present data. The types of profiles used in this work are the same as those of [13], including:

- Social spambots#1 (Social1) which includes accounts that retweeted the candidate for mayor of Rome in the 2014 Italian elections, where one of the candidates hired a marketing company that used 991 bots to run the campaign.
- Genuine accounts which the $3,474$ accounts were classified as legitimate using the following method: a simple question was asked in natural language for profiles chosen at random in the OSN, after this step the responses were analyzed by human beings to define whether the account is legitimate.
- Traditional spambots#1 (Traditional) is the same dataset provided by [24], which contains information on $1,000$ bots that have posted with malicious links.

The classification algorithms used are the same as those of [13], trained with 90% of the dataset Genuine

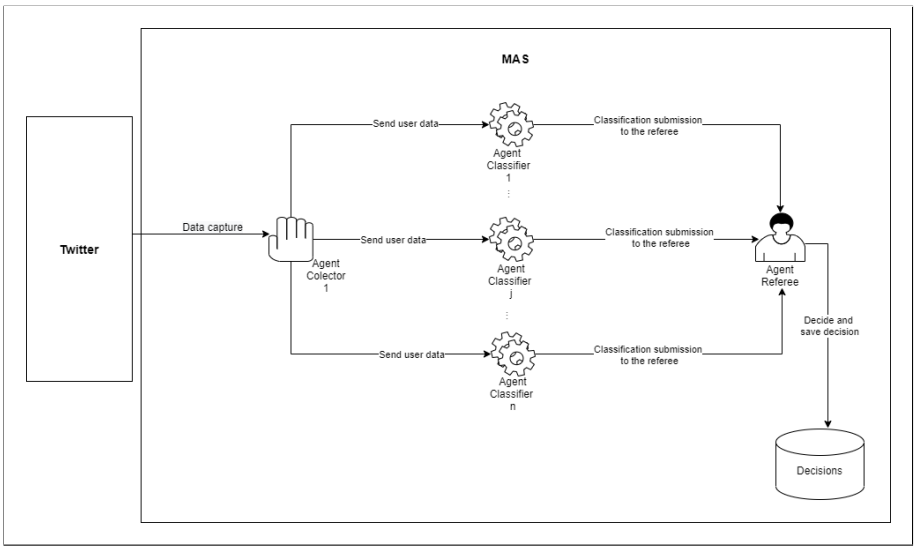

Figure 2. The MAS horizontally layered architecture.

accounts combined with 90% of social spambots#1 xor Traditional spambots #1, serialized with the help of the pickle [36], [37] library. The serialization is important to avoid training the three classifiers every time the MAS is executed. The features set adopted here is the same as [13], it includes the amount of tweets ($statuses\_count$), number of followers ($followers\_count$), number of friends ($friends\_count$), number of likes given by the account ($favorites\_count$), and number of lists that the account is included ($listed\_count$).

5. Experiments and Results

Two different tests were carried out during the development of this research. The first one intended to measure the performance of the MAS developed, using the same methodology as [13] and comparing the two works. The second experiment is a proof of concept involving the use of the MAS to perform the Twitter bot detection on an online stream of tweets. Both tests are better discussed in the following subsections.

5.1. MAS Performance

To validate the proposed solution, the MAS performance was compared with the three algorithms executed separately as previously presented in [13]. In Table 3 we present the comparison of the scores for the three algorithms - RF, SVM, and NB - using the Social1 and Traditional datasets. As in [13] the implementation of the ML algorithms are provided by [38]. Note that, the MAS presented the second-best performance among the other algorithms with an accuracy of 0.9795 in the dataset Social1 and 0.9840 in the Traditional dataset, only behind the RF results with 0.9821 and 0.9933 with Social1 and Traditional datasets, respectively.

According to [39], [40], the recall (also known as sensitivity) is the ratio between the correctly classified examples (true-positive - TP) and the incorrectly classified examples (false-negative - FN). In other words, is the proportion of true-positive cases that are correctly predicted. As we can see the recall score from the MAS is considerably high (it is only lower than the RF) in both datasets. This measure indicates that the MAS maintains the high sensitivity achieved by our best ML classifier.

According to [41], F1-score represents the harmonic average between precision and recall. The value of the F1-score is a number between zero and one, where values closer to one indicate high classification performance. Over again, our MAS keeps the good score achieved by our experiment performed at [13], which is a good indicator that the solution proposed applies to the task of classifying Twitter bots. The results related to the AUC score are better explained below.

In Table 4 there is a comparison between the AUC of the classifiers executed separately and combined in the MAS. For the calculus of the AUC, the probability of the MAS was measured considering the average of the probabilities of the algorithms. This procedure is the same used in [38] for the calculus of the probability for the Adaboost ensemble method. Since we have fewer examples for each type of bot than examples of legitimate accounts in the dataset, we faced a class imbalance problem. Therefore, choosing AUC as a performance measure is appropriate [14], [42]. The MAS achieved the second-best performance among the other algorithms with an average AUC of 0.9856 (standard deviation of 0.0199) only behind the RF results with 0.9878 (standard deviation of 0.0170). Figure 3 contains the results presented in Table 4, where for each classifier the first two bars represent the AUC values obtained using the two datasets (i.e., Social1 and Traditional) and the third bar contains the average value. The standard deviation is also shown in the first two bars.

TABLE 3. ACCURACY, AUC, RECALL AND E F1-SCORE FOR CLASSIFIERS (ALONE AND COMBINED IN MAS), ACCORDING TO TRAIN/TEST DATASET.

ML algorithm/ Dataset	Accuracy	AUC	Recall	F1-score
RF/Social1	**0,9821**	0,9758	0,9636	0,9737
RF/Traditional	**0,9933**	0,9999	0,9850	0,9902
SVM/Social1	0,8281	0,9382	0,6186	0,6421
SVM/Traditional	0,9821	0,9978	0,9778	0,9744
NB/Social1	0,5000	0,9011	0,6710	0,4980
NB/Traditional	0,8438	0,9937	0,8994	0,8145
MAS/Social1	**0,9795**	0,9716	0,9587	0,9682
MAS/Traditional	**0,9840**	0,9996	0,9776	0,9756

TABLE 4. COMPARISON BETWEEN AUC OBTAINED FOR MAS AND THE CLASSIFIERS ALONE AS IN [13].

Técnica de AM	Social1	Traditional	Average	Standard Deviation
RF	0,9758	0,9999	**0,9878**	**0,0170**
SVM	0,9382	0,9978	0,9680	0,0421
NB	0,9011	0,9937	0,9474	0,0654
MAS	0,9716	0,9996	**0,9856**	**0,0199**

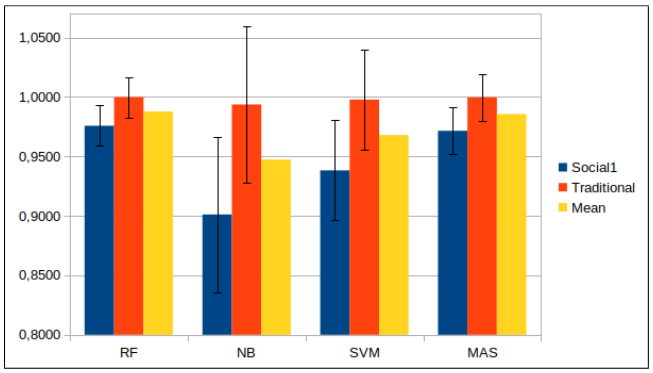

Figure 3. Parallel of the AUCs obteined for MAS and [13]

Considering the presented experiments, the MAS results are very close to the best algorithm that is RF meaning the approach can be used for classification with satisfactory performance. We might consider that the method on which the vote is taken is not complementary. Also, the RF algorithm presents better results considering the datasets

used, but we cannot guarantee it will perform better with any dataset. The ensemble methods in ML produce optimal predictive models, as pointed out in the literature and the MAS approach, can provide an ideal framework for the implementation of such methods. In this work, we provided the MAS proof of concept with three different ML algorithms and a simple decision process based on voting. However, other solutions can enhance the results of ensemble methods especially considering online tweet bot detection.

5.2. Online Twitter Bot Detection

An important attribute for a bot classifier is execution in near real time, thus becoming a detector. In [9] there is a demonstration that the detection of bots from Twitter allows very simple bots to continue working on the platform. Therefore, it is necessary to have tools that help the discovery of bots as close as possible to real time. A very slow tool can detect bots when they have already been excluded or have already caused damage. Thus, harmful bots can be discovered earlier by minimizing the damage caused by them. During the literature review it was noticeable that most articles do not perform the detection of bots in real time. In this way, a proof of concept (PoC) of the execution of the MAS was also carried out using real-time data collected from the Twitter API. The main objective of this PoC was to assess the MAS approach feaseability for real time detection.

The methodology was simple and used the MAS set up with the three ML classifiers (RF, SVM, and NB) trained with the dataset Social1 combined with the dataset genuine accounts (both provided by [23]), the same used in part of the first test. This dataset choice was done intending to discover the most sophisticated bots, as argued in [13], [23]. The stream of tweets was retrieved looking for the keyword "vacina" which is the Portuguese word that means vaccine. This keyword was chosen by the fact that the test was carried out during the COVID19 pandemic, and this theme was a hot topic in Brazil. It is notorious that it is more likely to capture bots in a larger sample of accounts, and the keyword is very helpful to allow for it. The test was executed using one notebook with the following specs: 7,8 GiB of RAM, CPU Intel® Core™ i7-3610QM, GPU NVIDIA® GeForce® GT 630M with 2GB DDR3 VRAM, Solid State Drive 250 GB, Operational System Xubuntu 20.04.2 LTS (64 bits).

The execution took place on February $12^{th}, 2021$, for approximately six hours (from 11:56 am to 6:00 pm). About 44,156 tweets were captured, corresponding to 31,271 different users, which were classified (representing a throughput of approximately 123 tweets per minute, roughly two per second). Among these, 183 tweets came from profiles classified as bots by MAS. These 183 suspicious tweets were published by 144 different Twitter accounts.

On March $25^{th}, 2021$, a request for data about these 144 profiles classified as bots was carried out. The Twitter API was able to return data about only 17 profiles, which means that 127 suspicious profiles were suspended by Twitter or excluded by their owners. It suggests that $88,19\%$ of the users classified as bots by our method were correctly labeled, using the same methodology as [27] to validate their results. This preliminary test, although quite simple, showed that the proposed methodology is on the right track and applies to real-time data.

6. Conclusion

The main objective of extending the work in [13] by developing a MAS capable of autonomously detecting Twitter bots was successfully achieved. The average AUC of 0.9856 and standard deviation equal to 0.0199 shows that MAS is useful for labeling bots on Twitter. The approach applyed in the MAS developement allows exploring ML classifiers by replacing, removing or adding algorithms.

The MAS architecture naturally accommodated online detection as it uses a direct pipeline for processing and classifying tweets captured by agents. When tested on a PoC with an online tweet stream it behaved well with an indication of good throughput and accuracy of detection. Nevertheless, more robust tests should be executed to evaluate the classification of data using online streams.

In future work, we intend to improve agents' reasoning capacity, especially the referee agent. This can be enhanced by making more rational decisions about the use of classification votes from the agents. Another idea would be to transform the referee into a coordinator agent that could perform a pre-classification by sending profiles with certain characteristics to specialized classifiers saving computational resources. Furthermore, we intend to carry out a more robust validation of online bot detection, since this is a very important feature for the adoption of the MAS as a real-time Twitter bot detector.

References

[1] B. Schneier, "Secrets & lies: Digital security in a networked world," *International Hydrographic Review*, vol. 2, no. 1, pp. 103–104, 2001.

[2] K. D. Mitnick and W. L. Simon, *The art of deception: Controlling the human element of security*. John Wiley & Sons, 2003.

[3] M. Shafahi, L. Kempers, and H. Afsarmanesh, "Phishing through social bots on twitter," in *Big Data (Big Data), 2016 IEEE International Conference on*. IEEE, 2016, pp. 3703–3712.

[4] M. Mazza, S. Cresci, M. Avvenuti, W. Quattrociocchi, and M. Tesconi, "RTbust: Exploiting Temporal Patterns for Botnet Detection on Twitter," in *Proceedings of the 10th ACM Conference on Web Science*, ser. WebSci '19. Boston, Massachusetts, USA: Association for Computing Machinery, 2019, pp. 183–192. [Online]. Available: https://doi.org/10.1145/3292522.3326015

[5] A. H. Wang, "Detecting spam bots in online social networking sites: A machine learning approach," in *Data and Applications Security and Privacy XXIV*, S. Foresti and S. Jajodia, Eds. Berlin, Heidelberg: Springer, 2010, pp. 335–342.

[6] G. Stringhini, C. Kruegel, and G. Vigna, "Detecting spammers on social networks," in *Proceedings of the 26th Annual Computer Security Applications Conference*, ser. ACSAC '10. New York, NY, USA: Association for Computing Machinery, 2010, p. 1–9. [Online]. Available: https://doi.org/10.1145/1920261.1920263

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

[8] G. Tavares and A. Faisal, "Scaling-laws of human broadcast communication enable distinction between human, corporate and robot twitter users," *PloS one*, vol. 8, no. 7, 2013.

[9] J. V. F. Abreu, J. H. C. Fernandes, J. J. C. Gondim, and C. G. Ralha, "Bot development for social engineering attacks on twitter," 2020.

[10] C. Freitas, F. Benevenuto, S. Ghosh, and A. Veloso, "Reverse engineering socialbot infiltration strategies in twitter," in *2015 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*. IEEE, 2015, pp. 25–32.

[11] L. da Mata, "Plano de trabalho CPMI da Fake News," Congresso Nacional. Disponível em: http://legis.senado.leg.br/sdleg-getter/documento/download/d78bab7d-515f-45df-b785-03e364a7e138 Acessado em: 19/04/2020., 2019.

[12] M. Latah, "Detection of malicious social bots: A survey and a refined taxonomy," *Expert Systems with Applications*, vol. 151, p. 113383, 2020. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0957417420302074

[13] J. V. Fonseca Abreu, C. Ghedini Ralha, and J. J. Costa Gondim, "Twitter bot detection with reduced feature set," in *2020 IEEE International Conference on Intelligence and Security Informatics (ISI)*, 2020, pp. 1–6.

[14] J. Rodríguez-Ruiz, J. I. Mata-Sánchez, R. Monroy, O. Loyola-González, and A. López-Cuevas, "A one-class classification approach for bot detection on twitter," *Computers & Security*, vol. 91, p. 101715, 2020. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0167404820300031

[15] S. Russell and P. Norvig, *Artificial Intelligence: A Modern Approach*, 3rd ed. USA: Prentice Hall Press, 2009.

[16] D. Poole, A. Mackworth, and R. Goebel, *Computational Intelligence*. UK: Oxford university press, 1998.

[17] M. Wooldridge, *An introduction to multiagent systems*, 2nd ed. UK: John Wiley & Sons Ltd, 2009.

[18] G. Weiss, Ed., *Multiagent systems*, 2nd ed. MIT press, 2013.

[19] P. Bresciani, A. Perini, P. Giorgini, F. Giunchiglia, and J. Mylopoulos, "Tropos: An agent-oriented software development methodology," *Autonomous Agents and Multi-Agent Systems*, vol. 8, no. 3, pp. 203–236, 2004.

[20] F. Dalpiaz, X. Franch, and J. Horkoff, "istar 2.0 language guide," 2016.

[21] E. Beğenilmiş and S. Uskudarli, "Organized behavior classification of tweet sets using supervised learning methods," in *Proceedings of the 8th International Conference on Web Intelligence, Mining and Semantics*, ser. WIMS '18. New York, NY, USA: Association for Computing Machinery, 2018. [Online]. Available: https://doi.org/10.1145/3227609.3227665

[22] O. Varol, E. Ferrara, C. A. Davis, F. Menczer, and A. Flammini, "Online Human-Bot Interactions: Detection, Estimation, and Characterization," in *11th International AAAI Conference on Web and Social Media (ICWSM)*, May 2017. [Online]. Available: https://www.aaai.org/ocs/index.php/ICWSM/ICWSM17/paper/view/15587

[23] S. Cresci, R. Di Pietro, M. Petrocchi, A. Spognardi, and M. Tesconi, "The paradigm-shift of social spambots: Evidence, theories, and tools for the arms race," in *Proceedings of the 26th International Conference on World Wide Web Companion*, ser. WWW '17 Companion. Republic and Canton of Geneva, CHE: International World Wide Web Conferences Steering Committee, 2017, p. 963–972. [Online]. Available: https://doi.org/10.1145/3041021.3055135

[24] C. Yang, R. Harkreader, and G. Gu, "Empirical evaluation and new design for fighting evolving twitter spammers," *IEEE Transactions on Information Forensics and Security*, vol. 8, no. 8, pp. 1280–1293, 2013.

[25] S. Cresci, R. Di Pietro, M. Petrocchi, A. Spognardi, and M. Tesconi, "Fame for sale: Efficient detection of fake twitter followers," *Decision Support Systems*, vol. 80, pp. 56–71, 2015.

[26] S. Liu, B. Hooi, and C. Faloutsos, "A contrast metric for fraud detection in rich graphs," *IEEE Transactions on Knowledge and Data Engineering*, vol. 31, no. 12, pp. 2235–2248, 2019.

[27] K. Lee, B. Eoff, and J. Caverlee, "Seven Months with the Devils: A Long-Term Study of Content Polluters on Twitter," in *International Conference on Weblogs and Social Media (ICWSM), AAAI Publications*, 2011.

[28] M. Fazil and M. Abulaish, "A hybrid approach for detecting automated spammers in twitter," *IEEE Transactions on Information Forensics and Security*, vol. 13, no. 11, pp. 2707–2719, 2018.

[29] Twitter Inc., "Help center," https://help.twitter.com/en, 2020, accessed 09/17/2020.

[30] T. Gui, P. Liu, Q. Zhang, L. Zhu, M. Peng, Y. Zhou, and X. Huang, "Mention recommendation in twitter with cooperative multi-agent reinforcement learning," in *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*, ser. SIGIR'19. New York, NY, USA: Association for Computing Machinery, 2019, p. 535–544.

[31] J. Pimentel and J. Castro, "pistar tool – a pluggable online tool for goal modeling," in *2018 IEEE 26th International Requirements Engineering Conference (RE)*, 2018, pp. 498–499.

[32] F. J. Mora Lizán and C. Rizo-Maestre, "Intelligent buildings: Foundation for intelligent physical agents," 2017-05.

[33] L. S. Melo, R. F. Sampaio, R. P. S. Leão, G. C. Barroso, and J. R. Bezerra, "Python-based multi-agent platform for application on power grids," *International Transactions on Electrical Energy Systems*, vol. 29, no. 6, p. e12012, 2019.

[34] P. D. O'Brien and R. C. Nicol, "Fipa—towards a standard for software agents," *BT Technology Journal*, vol. 16, no. 3, pp. 51–59, 1998.

[35] M. A. Karzan and N. Erdogan, "Topic based agent migration scheme via publish/subscribe paradigm," *International Journal of Information and Education Technology*, vol. 3, no. 3, p. 290, 2013.

[36] M. Lutz, *Learning python: Powerful object-oriented programming*. " O'Reilly Media, Inc.", 2013.

[37] P. S. Foundation, "pickle — python object serialization — python 3.9.2 documentation," https://docs.python.org/3/library/pickle.html, 03 2021, (Accessed on 03/18/2021).

[38] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[39] M. Sokolova, N. Japkowicz, and S. Szpakowicz, "Beyond accuracy, f-score and roc: A family of discriminant measures for performance evaluation," in *AI 2006: Advances in Artificial Intelligence*, A. Sattar and B.-h. Kang, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 1015–1021.

[40] D. M. Powers, "Evaluation: From precision, recall and f-measure to roc., informedness, markedness & correlation," *Journal of Machine Learning Technologies*, vol. 2, no. 1, pp. 37–63, 2011.

[41] A. Tharwat, "Classification assessment methods," *Applied Computing and Informatics*, 2018. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S2210832718301546

[42] Provost, F and Fawcett, T, "Analysis and visualization of classifier performance: Comparison under imprecise class and cost distributions," in *Proceedings of the Third International Conference on Knowledge Discovery and Data Mining*, 1997, pp. 43–48.