COPPE
Instituto Alberto Luiz Coimbra de
Pós-Graduação e Pesquisa de Engenharia
UFRJ

A FRAMEWORK FOR DETECTING COORDINATED POSTS OF BOTS IN ONLINE SOCIAL NETWORKS

Victor Garritano Noronha

Dissertação de Mestrado apresentada ao Programa de Pós-graduação em Engenharia de Sistemas e Computação, COPPE, da Universidade Federal do Rio de Janeiro, como parte dos requisitos necessários à obtenção do título de Mestre em Engenharia de Sistemas e Computação.

Orientadores: Daniel Ratton Figueiredo
Gerson Zaverucha

Rio de Janeiro
Julho de 2021

A FRAMEWORK FOR DETECTING COORDINATED POSTS OF BOTS IN ONLINE SOCIAL NETWORKS

Victor Garritano Noronha

DISSERTAÇÃO SUBMETIDA AO CORPO DOCENTE DO INSTITUTO ALBERTO LUIZ COIMBRA DE PÓS-GRADUAÇÃO E PESQUISA DE ENGENHARIA DA UNIVERSIDADE FEDERAL DO RIO DE JANEIRO COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE MESTRE EM CIÊNCIAS EM ENGENHARIA DE SISTEMAS E COMPUTAÇÃO.

Orientadores: Daniel Ratton Figueiredo
Gerson Zaverucha

Aprovada por: Prof. Daniel Ratton Figueiredo
Prof. Gerson Zaverucha
Prof. Pedro Olmo Stancioli Vaz De Melo
Prof. André Carlos Ponce de Leon Ferreira de Carvalho

RIO DE JANEIRO, RJ – BRASIL
JULHO DE 2021

Noronha, Victor Garritano

A framework for detecting coordinated posts of bots in online social networks/Victor Garritano Noronha. – Rio de Janeiro: UFRJ/COPPE, 2021.

XIII, 75 p.: il.; 29, 7cm.

Orientadores: Daniel Ratton Figueiredo
Gerson Zaverucha

Dissertação (mestrado) – UFRJ/COPPE/Programa de Engenharia de Sistemas e Computação, 2021.

Referências Bibliográficas: p. 65 – 75.

1. bot detection. 2. botnet. 3. online social network. 4. community detection. 5. twitter. I. Figueiredo, Daniel Ratton *et al.* II. Universidade Federal do Rio de Janeiro, COPPE, Programa de Engenharia de Sistemas e Computação. III. Título.

Agradecimentos

Agradeço a Jesus, meu Senhor e Salvador, por me dar forças, ânimo e sabedoria durante todos os momentos, e por me fazer chegar até a conclusão de mais essa etapa. A Ele seja toda honra e glória.

Agradeço a minha família, em especial aos meus pais, Nadia e Adilson e a minha irmã Vivian, por me apoiarem e me incentivarem durante o mestrado.

Agradeço aos meus amigos, em especial aos que estiveram comigo durante a graduação e agora no mestrado, por todos os momentos vividos juntos e aprendizados compartilhados, na vida dentro e fora da sala de aula.

Agradeço também aos meus orientadores do mestrado, Daniel e Gerson, por todos os ensinamentos e dedicação durante este 1 ano e meio que trabalhamos juntos.

Registro também um agradecimento final à Twist, empresa da qual faço parte desde 2018. Obrigado a todos os colaboradores pelos momentos juntos, toda a ajuda, todo o incentivo, todos os conselhos e desafios superados.

Resumo da Dissertação apresentada à COPPE/UFRJ como parte dos requisitos necessários para a obtenção do grau de Mestre em Ciências (M.Sc.)

UM FRAMEWORK PARA DETECÇÃO DE POSTAGENS COORDENADAS DE BOTS EM REDES SOCIAIS ONLINE

Victor Garritano Noronha

Julho/2021

Orientadores: Daniel Ratton Figueiredo
Gerson Zaverucha

Programa: Engenharia de Sistemas e Computação

A crescente popularidade das redes sociais online mudou a maneira pela qual as pessoas geram, consomem e interagem com a informação. Entretanto, essa transição levanta diversos questionamentos sobre a credibilidade do conteúdo compartilhado nessas redes, uma vez que elas podem ser utilizadas por diferentes segmentos da sociedade para disseminar desinformação e manipular a opinião pública. Uma parte crucial das estratégias utilizadas por essas organizações são os usuários automatizados (*bots*), que são mobilizados para coordenar suas ações, com o intuito de promover conteúdo de maneira direcionada e popularizar artificialmente assuntos ou personalidades. Os métodos tradicionais para identificação de *bots* falham em identificar esse comportamento sincronizado e coordenação contextual, uma vez que eles analisam as contas de usuários individualmente. Para suprir essa defiência, este trabalho propõe uma metodologia para identificação de grupos de *bots* que coordenam suas postagens nas redes sociais online, utilizando as similaridades de conteúdo entre as contas para criar redes de usuários que revelam esses grupos e seus tópicos. Utilizando duas bases de dados reais do Twitter, as diferenças estruturais e linguísticas entre os grupos compostos por usuários autênticos e automatizados são investigadas. Além disso, um método para classificar automaticamente os grupos como *bots* ou humanos é desenvolvido. A metodologia proposta é aplicada em três bases de dados de acontecimentos recentes no Brasil para identificar os grupos de *bots* e o foco de suas ações.

Abstract of Dissertation presented to COPPE/UFRJ as a partial fulfillment of the requirements for the degree of Master of Science (M.Sc.)

A FRAMEWORK FOR DETECTING COORDINATED POSTS OF BOTS IN ONLINE SOCIAL NETWORKS

Victor Garritano Noronha

July/2021

Advisors: Daniel Ratton Figueiredo
Gerson Zaverucha

Department: Systems Engineering and Computer Science

The growing popularity of online social networks have changed the way people generate, consume and interact with information. However, this transition brings several concerns on the credibility of the content shared in such networks, as they can be used by different segments of society to spread misinformation and manipulate public opinion. A crucial part of the strategies used by such organizations are the automated accounts (bots), that are mobilized to coordinate their actions in order to promote targeted content and artificially increase the popularity of a topic or a person. Traditional approaches for bot detection fail to identify this synchronized behavior and contextual coordination, as they analyze the user accounts individually. To bridge this gap, this work proposes a framework for detecting communities of bots that coordinate their posts in online social networks, leveraging content similarities between users to build user networks that reveal such communities and their topics. Using two real datasets of Twitter accounts, we investigate the structural and linguistic differences between communities composed of automated and authentic user accounts. Moreover, we design a procedure to automatically classify the communities as bots or humans based on their linguistic and structural characteristics. The framework is also applied to three datasets comprising recent events in Brazil to uncover communities of bots and the focus of their actions.

Contents

List of Figures

List of Tables

List of Abbreviations

BERT	Bidirectional Encoder Representations from Transformers, p. 7
ECCDF	Empirical Complementary Cumulative Distribution Function, p. 27
ELMo	Embeddings from Language Models, p. 7
ER	edge reciprocity, p. 36
GCC	global clustering coefficient, p. 36
GN	Girvan-Newman, p. 9
JS	Jensen-Shannon, p. 22
KL	Kullback-Leibner, p. 22
LM	Language Model, p. 5
LSTM	Long short-term memory, p. 6
OSN	online social network, p. 64
POS	Part-of-Speech, p. 16
PR	PageRank, p. 24
RNNs	Recurrent Neural Networks, p. 6
SAW	self-avoiding random walks, p. 11
UMLFiT	Universal Language Model Finetuning, p. 7
biLSTM	bidirectional LSTM, p. 7

Chapter 1

Introduction

Over the past decade, the number of online social network users has increased exponentially. In the beginning of 2021, the barrier of 4 billion active users has been broken, corresponding to 53.6% of the global population and an increase of 490 million compared to the number of users in the previous year [4]. Besides connecting people from different places around the world, these social networks have been replacing traditional media outlets and are becoming a the major source of information for a considerable portion of their users [5].

However, online social networks are also extensively used to spread disinformation, due to the ease with which information can be propagated as users actively post and share content, contributing to the dissemination of rumors, spam and fake news [6]. This kind of content has started to be used by government agencies, political parties and private firms to manipulate public opinion around the world [7].

These organizations use actors known as “ cyber troops ” to deploy their agendas in online social networks, which includes visible government representatives, social media influencers and even private companies that offer computational propaganda as a service, focused on spreading pro-government content, attacking and propagating smear campaigns against opposition parties, suppressing attendance at democratic events and encouraging division and polarization among citizens [7].

These troops are often composed of several types of accounts that support their actions, like authentic accounts, operated by humans and responsible for engaging in specific discussions towards promoting a certain narrative, and automated accounts, known as bots, that act to artificially increase the popularity of a specific topic or point of view and undermine the credibility and reach of others. The “2020 Global Inventory of Organized Social Media Manipulation” from the Oxford Internet Institute [7] has found evidence that automated accounts have been used to spread disinformation in 57 (70%) out of 81 monitored countries around the world.

Bot accounts actively influence the discussions and consequently the viral topics in online social networks, especially information originating from low-credibility sources, targeting susceptible users with tailored information that are reshared by them afterwards, providing a false sense of authenticity for these potentially fraudulent contents [8]. Thus, it is crucial to disclose these bot accounts, as they play a fundamental role in the spread of disinformation and the opinion formed by individuals.

Different approaches have been introduced to detect bots in online social networks: several methods extract features from the metadata and shared content of accounts to classify them in a supervised way [9], [10], [11], [12]. Methods that analyze the structure of the interaction networks of bot and human accounts have also been proposed [13], [14], [15], [16]. At the same time, unsupervised techniques have been introduced as an way to alleviate the need for annotated datasets of bot accounts [17], [18], [19].

However, identifying automated accounts is a challenging task, since bot developers are constantly evolving the behavior of their agents in order to evade the most recent detection systems [20]. This makes existing methodologies quickly obsolete, as they are unable to generalize and recognize the characteristics of unseen types of bots [21]. Moreover, bot developers constantly create new accounts, as the account are identified and shutdown by online social network administrators [22]. Therefore, the bot detection systems also have to continuously evolve. The Botometer system, for example, have deployed several versions of its framework over the recent years, in order to map the newer bot behaviors and enhance the generalization capabilities of the system [9], [23], [12].

Besides this evolution, bot developers are also changing their *modus operandi*, putting aside individual and isolated actions to create large groups of coordinated accounts, that act together to reshape political discussions and increase the popularity of target users [24], [2]. This coordination among multiple automated accounts is much more effective in disseminating misinformation and forming opinion.

This new behavior imposes an additional challenge, as the standard machine learning approaches to identify automated accounts traditionally analyze each profile individually, thus being ultimately unable to detect groups of accounts that coordinate their actions. Moreover, the existing literature of detection systems for identification of coordinated groups of bots is still very scarce [2].

In order to help bridge this gap, this work introduces a novel framework for detecting groups of coordinated bot accounts in online social networks. The key idea is to link bot accounts based on the content generated by them (e.g. posts). The similarities among their content is used to uncover groups of users that act together to promote a topic of interest. While the key idea leverages content, the proposed methodology is language-agnostic, as it relies only on statistical features of the vocabulary to characterize and establish similarities among accounts. The framework is also system-agnostic, as it can be applied to any online social network where accounts post information (e.g. Twitter).

1.1 Contributions

This work makes the following contributions:

- Propose a novel and unsupervised framework for detecting groups of users that post and share similar content in online social networks, as an indication of coordinated action. Our framework builds representations of users based on their content, and use these representations to establish a similarity among them. This information is used to construct a network of users where edges represent very similar accounts. The network is then used to reveal groups (communities) of accounts that possibly coordinate their actions (posts).

- Conduct an analysis using two datasets of bot accounts in Twitter to assess the existence of structural differences between different types of users with respect to the connectivity among their peers, and verify whether these differences could indicate coordinated actions. The impact of different parameters of the framework in the quality of the resulting coordinated groups is also evaluated.

- Investigate the main topic that circulated inside the resulting communities returned by the framework in the two datasets. We also compare the observed results and communities with the ones reported in the original works of each of the datasets.

- Formulate a machine learning classifier to evaluate the resulting communities according to their features as either automated (`bot`) or organic (`human`) coordination, using features derived from both structural and linguistic characteristics of communities. Using the two datasets jointly, the classifier had an accuracy of 0.85 and F1-score of 0.83, indicating its effectiveness in revealing the kind of community (`bot` or `human`).

- Apply the proposed framework to three datasets collected in Twitter during recent events in Brazil, uncovering the coordinated groups in each of them, as well as a comparison with Botometer v4 [12], a state-of-the-art system for bot detection in Twitter.

1.2 Thesis structure

The remainder of the text is organized as follows:

- Chapter 2 reviews the recent literature regarding the main aspects of the proposed framework. We cover previous works related to language models, community detection in networks and bots classification in online social networks.

- Chapter 3 presents the proposed framework, detailing its main components and methods.

- Chapter 4 presents the results of the framework when applied to two publicly available dataset of Twitter accounts, investigating structural distinctions between the two classes of users (`bots` and `humans`) as well as how different parameters settings affect the quality of the identified communities of users.

- Chapter 5 we presents the structural and linguistic features used to train a classifier to determine the kind of a community identified by the unsupervised framework (`bot` or `human`), as well as the training methodology and generalization performance.

- Finally Chapter 6 draws some conclusions and point out directions for future work.

Chapter 2

Related Work

This chapter briefly reviews the recent literature regarding the main components of the proposed framework. We cover works related to the different methods such as language models, community detection in networks and detection of bot accounts in social networks.

2.1 Language Models

The first step of the proposed framework is responsible for characterizing users using a Language Model (LM). Different variations of the simplest LM have been proposed across the years, and this section briefly presents some of those techniques.

2.1.1 N-gram Language Models

A Language Model is a model that assigns probabilities over sequences of words, computing $P(W) = P(w_1, w_2, \ldots, w_k)$, where $w_1, w_2, \ldots, w_k$ is a sequence of words. Using the chain rule of probability, we can decompose $P(W)$ as:

$$\begin{aligned} P(W) &= P(w_1, w_2, \ldots, w_k) \\ &= \prod_i P(w_i | w_1, w_2, \ldots, w_{i-1}) \end{aligned} \tag{2.1}$$

Thus, we can rewrite $P(W)$ as a product of conditional probabilities, computing the probability of a word w_i given the previous words $w_1, w_2, \ldots, w_{i-1}$ in the sequence. Furthermore, assuming the Markov property, we can approximate $P(w_i|w_1, w_2, \ldots, w_{i-1})$ by only considering a window of n previous words, instead of the complete sequence of $i - 1$ words observed so far, as shown in Eq. 2.2.

$$P(w_i|w_1, w_2, \ldots, w_{i-1}) \approx P(w_i|w_{i-n}, \ldots, w_{i-1}) \tag{2.2}$$

This approximation leads to one of the simplest language models, the *n-gram model*, which considers the $n-1$ previous words during the computation of $P(W)$.

2.1.2 Language Models Based on Shallow Neural Networks

Due to its simplicity, there are some limitations regarding the n-gram language model: as we increase the number of possible word sequences, and thus parameters, grows exponentially, which can be prohibitive when dealing with a large corpus. Moreover, the n-gram models will assign a zero probability to unseen words or sequence of words, which degrades their generalization capabilities.

In order to overcome these shortcomings, several improvements have been proposed in the literature. In [25], a neural network methodology was proposed to compute $P(w_i|w_{i-n},\ldots,w_{i-1})$, decomposing in two parts: in the first step, each word is associated with a real-valued vector of fixed dimension, known as an embedding; in the second step, a neural network learns a probability function over words, taking the embeddings associated with the previous words $w_{i-n},\ldots,w_{i-1}$ as the input and the word w_i as the expected output.

During the optimization phase, both embeddings and neural networks parameters are learned. The proposed methodology scales linearly with the vocabulary size and can generalize better because semantically and syntactically similar words are expected to have similar embeddings.

One major drawback of this approach is that the context size is still limited to a fixed window of $n-1$ previous words, preventing the model from capturing longer contextual information.

In [26], a Language Model based on Recurrent Neural Networks (RNNs) was proposed. The RNNs encode context information of arbitrary length, and were used to enhance the contextual capabilities of prior Language Models, outperforming n-gram models in a variety of tasks [27]. However, the RNNs have difficulties to handle long-term contexts that may appear in long sequences, due to the its *vanishing and exploding gradient* problems, which slows down the training process and leads to overfitting [28].

As an alternative to RNNs, the Long short-term memory (LSTM) unit was introduced [29]. The LSTM is explicitly designed to avoid the aforementioned gradient problems and learn long-term dependencies, using a more complex and expensive structure to perform computations. A Language Model using these units has been proposed in [30].

2.1.3 Language Models Based on Deep Learning

In recent years, several works have been introduced for Language Models using architectures based on Deep Learning. The Embeddings from Language Models (ELMo) model [31] uses a bidirectional LSTM (biLSTM) to compute contextual embeddings considering words appearing before and after w_i in the sequence. The Universal Language Model Finetuning (UMLFiT) introduces a procedure for finetuning a pretrained LM for different NLP tasks, like text classification and sentiment analysis, exhibiting meaningful performance improvements in these tasks [32].

Another promising approach is the Transformer model [33], that replaced the recurrent units in favor of attention mechanisms, as an alternative for capturing contextual dependencies. This modification makes the Transformer model suitable for parallel computations, unlike LSTM and RNN units, that fundamentally relies on sequential computations. Several Transformer variations have been proposed, like Bidirectional Encoder Representations from Transformers (BERT) [34], Transformer-XL [35] and GPT-2 [36], exhibiting an exponentially increasing number of parameters and establishing new state-of-the-art results in several natural language inference and language modeling benchmarks.

In the opposite direction, several works introduce strategies to reduce the computational complexity associated with the Transformer model, while retaining their linguistic capabilities, like the DistilBert model, [37] which has been able to preserve 97% of the performance of the pre-trained BERT model with 40% fewer parameters on the GLUE benchmark for Natural Language Understanding tasks [38]; the Reformer model [39], which have reduced to computational and memory complexity of the attention mechanism from $\mathcal{O}(L^2)$ to $\mathcal{O}(L \log L)$ (where L is the sequence length); and the Linformer model [40], which has achieved a linear complexity in the attention computation using low-rank matrix approximations.

2.2 Network Community Detection

The main goal of the proposed framework is to identify coordinated groups of users in social networks. This objective can be interpreted as the task of Community Detection in Network Science, one of the most studied topics in the area [1]. The task aims to partition the network vertices into different groups (communities) according to the network structure and possibly node attributes. A community can be defined as a set of vertices where the density of edges among nodes inside the cluster is relatively higher than the density of edges between nodes inside and outside the cluster, as exemplified in Fig. 2.1.

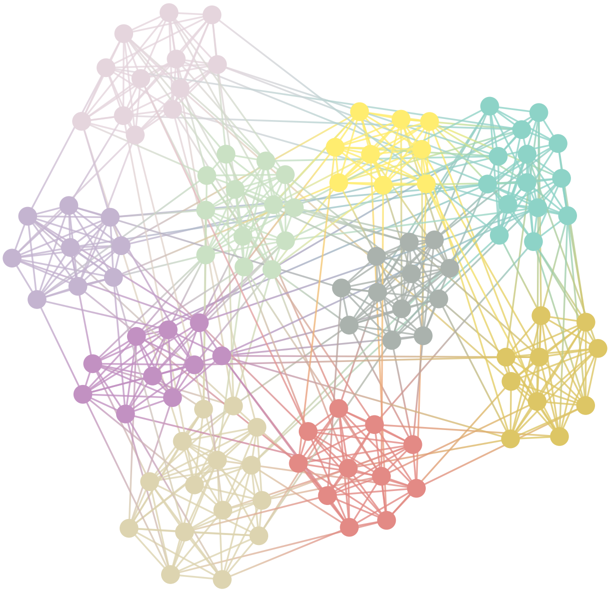

Figure 2.1: Network with 10 communities. The groups are composed by more densely connected subgraphs.

Given a specific partition of vertices, one should be able to assess the quality of this division. A commonly used metric is the *modularity* [41], which compares the density of edges inside a cluster with the density in the same group of vertices in a random network model, indicating whether the vertices inside the clustering do exhibit a community structure or the connections among them have emerged by chance. The maximum *modularity* value is 1 and the higher the value, the better the partition quality.

Considering a network with n vertices, m edges, A_{ij} as an element of adjacency matrix associated with the network, p_{ij} as the probability that an edge exists between vertices i and j, as well as a network partition into K communities, where V_k represents the set of vertices in cluster k, the *modularity* (M) is computed as:

$$M = \frac{1}{2m} \sum_{k \in K} \sum_{i,j \in V_k} (A_{ij} - p_{ij}) \tag{2.3}$$

If one assumes a random network model known as *configuration model* [42], p_{ij} is simply defined as:

$$p_{ij} = \frac{d_i d_j}{2m} \tag{2.4}$$

where d_i and d_j denotes the degrees of vertices i and j, respectively. Thus, Eq. 2.3 can be written as:

$$M = \frac{1}{2m} \sum_{k \in K} \sum_{i,j \in V_k} \left(A_{ij} - \frac{d_i d_j}{2m} \right) \tag{2.5}$$

The task of finding the partition that maximizes the *modularity* is however a *strongly NP-complete* problem [43], for which several heuristic methods have been proposed.

2.2.1 Girvan-Newman Algorithm

The Girvan-Newman (GN) algorithm is a greedy method for finding good community structure in terms of *modularity* [44]. The procedure is defined as follows:

1. Initially, each vertex belongs to its own community;
2. For each pair of communities, one computes the increase in the *modularity* value by joining this pair into a single community;
3. Merge the pair of communities that provides the highest increase.
4. Return to step 2, until reaching a single partition.

The best partition scheme and *modularity* values in each iteration are recorded. Finally, the partition with the highest overall *modularity* across the iterations is returned.

The entire algorithm runs in time $\mathcal{O}((m+n)n)$, which can be prohibitive for large graphs in real-world applications.

2.2.2 Louvain Algorithm

The Louvain algorithm [45] proposes an alternative greedy algorithm, with linear complexity in the number of edges in the network, $\mathcal{O}(m)$. Instead of computing the *modularity* gain for each pair of clusters, the proposed method computes the increase when moving a vertex into the partition of one of its neighbors only. The complete procedure is defined as follows:

1. Initially, each vertex belongs to its own community;
2. For each vertex, compute the increase in the *modularity* value by merging this vertex into one of its neighbors' communities;
3. Retain the new community that gives the highest *modularity* increase.
4. Build a new network, where each community is represented by a single vertex.

5. Return to step 2, until no further improvement can be achieved

This method has been widely used and is available in many network science libraries, since it is computationally efficient and often yields good partitions [46], [47], [48].

2.2.3 Walktrap

Alternative approaches have been explored to identify communities in networks. One such approach is based on random walks. The intuition is that a walker tends to visit vertices in the same community for a significant number of steps, as the density among nodes inside the community is expected to be considerably higher than the density with vertices outside the community, which could be used to exploit the underlying community structure.

The *Walktrap* method, introduced in [49], explores this idea, proposing metrics of structural similarity between vertices and communities, and using these metrics to iteratively merge vertices into groups.

The distance metric between vertices is:

$$r_{i,j} = \sqrt{\sum_{k=1}^{n} \frac{(P_{ik}^t - P_{jk}^t)^2}{d(k)}} \tag{2.6}$$

where P_{ik}^t and P_{jk}^t encodes the probability of reaching vertex k, departing from vertex i and j in t steps respectively, and $d(k)$ is the degree of vertex k. Note that t is a parameter of the method.

The distance between vertices is generalized to represent the distance between communities in the following way:

$$r_{C_1,C_2} = \sqrt{\sum_{k=1}^{n} \frac{(P_{C_1,k}^t - P_{C_2,k}^t)^2}{d(k)}} \tag{2.7}$$

$$P_{C.,k}^t = \frac{1}{|C.|} \sum_{i \in C.} P_{ik}^t \tag{2.8}$$

where $P_{C.,k}^t$ represents the probability of reaching a vertex k departing from a community $C.$, which is defined in terms of the probability of reaching vertex k departing from any vertex i belonging to group $C.$, as shown in Eq. 2.8.

Using r_{C_1,C_2} as the distance metric, a hierarchical clustering algorithm is applied for finding communities: initially, each vertex belongs to its own community; then the pair of communities with smallest distance is selected and merged into a single group. The distance values between communities are updated and the process is repeated, until obtaining a single community. The final partition is chosen by selecting the set of communities with the highest *modularity* value. The distance computations and updates are only performed between adjacent communities (i.e. communities with at least one edge connecting them), in order to reduce complexity and ensuring connected groups. The algorithm runs in time $\mathcal{O}(mn^2)$.

2.2.4 Self-avoiding random walks for community detection

A method based on self-avoiding random walks (SAW) has been proposed in [50]. In this special random walk, the walker is not allowed to return to an already visited vertex, being forced to continue its walk only over unseen nodes. The proposed method computes the following for each pair of vertices i and j.

$$p_{i,j} = \frac{m_{i,j}}{M_i} \tag{2.9}$$

$$l_{i,j} = \frac{1}{m_{i,j}} \sum_{w_i^k} l_{i,j}^{w_i^k} \tag{2.10}$$

$$f_{i,j} = \frac{p_{i,j}}{l_{i,j}} \tag{2.11}$$

In Eq. 2.9, $m_{i,j}$ is the number of walkers that reach j starting from i and M_i is that number of walkers that started their walks in vertex i, indicating how easy it is to reach j from i. Eq. 2.10 resembles a mean distance between vertices, where $l_{i,j}^{w_i^k}$ is the number of steps that w_i^k, the k-th walker that started in vertex i, took to reach j. Finally, the equations are combined into a single metric $f_{i,j}$, as shown in Eq. 2.11. The method computes $f_{i,j}$ for each pair of vertices, storing them in a matrix $\mathcal{F}$. The work points out that high values of $f_{i,j}$ are expected between vertices of densely connected subgraphs, which are likely to correspond to communities. The method works as follows:

1. Run a sufficient number of SAWs starting from each vertex;

2. Compute $\mathcal{F}$;

3. Apply the PCA method, a dimensionality reduction technique, upon $\mathcal{F}$;

4. Extract the two principal components and project $\mathcal{F}$ into the new basis defined by the principal components, obtaining $\mathcal{F}'$;

5. Run an hierarchical clustering algorithm upon $\mathcal{F}'$, computing the *modularity* value in each iteration;

6. Record the partition that gives the highest *modularity* value;

7. Increment the number of principal components considered, obtaining a new $\mathcal{F}'$;

8. Return to step 5 and repeat the process, until using all components in the computation of $\mathcal{F}'$;

9. Return the partition with overall highest *modularity*;

The algorithm runs in time $\mathcal{O}(n^3 \log n)$, producing communities with higher *modularity* scores than prior approaches in real-world network benchmarks [50].

2.2.5 Consensus Clustering

Another approach for finding communities in networks is combining the information of different partitions, that may arise either from different methods or several runs of the same algorithm, which generates different outputs due to different choices of initial conditions or random seeds.

The Consensus Clustering algorithm [51] is based on this idea. In order to consider several partitions when dividing the network into communities, the approach introduces a Consensus Matrix $\mathcal{C}$, which stores the relative frequency of co-occurrence of vertices in clusters from different partitions, using $\mathcal{C}$ to define a new network, with a set of weighted edges that represents the co-occurrences, as exemplified in Fig. 2.2:

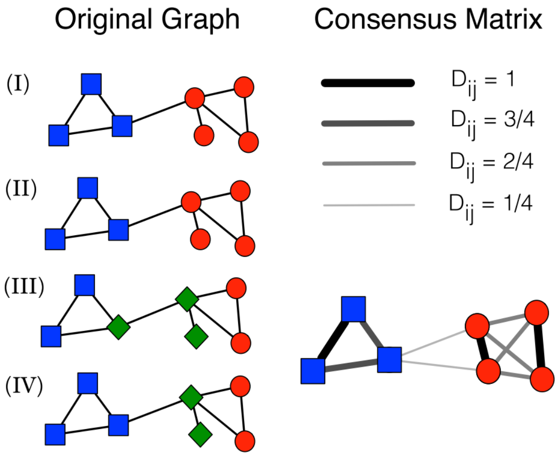

Figure 2.2: An example of the Consensus Clustering algorithm. The four partitions in the left part are used to compute the Consensus Matrix $\mathcal{C}$ that defines the network in the right part. The thickness of each edge is proportional to its value in $\mathcal{C}$. Extracted from [1]

The new network increases the cohesion between vertices in the same community and the separation between vertices in different communities, making the underlying community structure more evident.

Given a community detection algorithm $\mathcal{A}$, the following procedure is applied:

1. Apply $\mathcal{A}$ on the network p times, producing p partitions;
2. Compute the Consensus Matrix $\mathcal{C}$;
   (a) $\mathcal{C}_{ij}$ is defined as the number of partitions where vertices i and j are in the same group, divided by p;
   (b) $\mathcal{C}$ is expected to be denser than the original adjacency matrix;
   (c) Heavier edges are expected to occur between vertices in the same community while lighter ones may occur between nodes in different communities;
   (d) Optionally, one can define a threshold τ to remove edges that are too light;
3. Apply $\mathcal{A}$ p times, upon the network defined by $\mathcal{C}$ and go to Step 2;

This process are repeated, until every element of $\mathcal{C}$ assumes a value 0, indicating that i and j belongs to different communities, or 1, when i and j are in the same community.

The communities extracted from the final $\mathcal{C}$ correspond to the partition of the initial network. As the algorithm considers the co-occurrence between every pair of vertices, it scales quadratically in the number of nodes, which can be prohibitive in large networks. A variation of the algorithm that computes the co-occurrence only between adjacent vertices in the original network has been proposed [52], reducing the complexity of computing $\mathcal{C}$ to $\mathcal{O}(m)$.

2.2.6 Infomap

The Infomap [53] uses information about the dynamics in a network to highlight its community structure. The dynamics are encoded using the *map equation*, which describe the trajectory of a random walk in the network.

Each vertex will be associated with a codeword, following the Huffman coding scheme [54], that provides an optimal way to encode the vertices by assigning shorter codewords to nodes that are visited more often and longer ones to less visited nodes.

The method also attributes codes to network modules, that corresponds to communities, enabling the sharing of codewords between vertices, given that they belong to different communities, producing an additional reduction in the overall code length.

The codewords for nodes and modules are stored in different codebooks: the *index codebook* identifies the module where the walker is, and are composed by a collection of *m module codebooks*. The codes are derived from the frequency the walker visits each module. In a similar way, the *module codebooks* encodes the vertices that belong to each *module*, that are derived from the frequency the walker visits a vertex. Each *module codebook* also contains a code to indicate when the walker leaves the module.

Considering both codebooks, the average code length that describes a step of the random walk is defined by the *map equation* L($\mathcal{M}$), where $\mathcal{M}$ is a partition of n nodes in m modules:

$$L(\mathcal{M}) = q_{\curvearrowright} H(\mathcal{Q}) + \sum_{i=1}^{\mathbf{m}} p^{i}_{\circlearrowright} H(\mathcal{P}^{i}) \tag{2.12}$$

The first term encodes the average code length in the *index codebook*, weighted by the probability that the random walker switches modules. The second term represents the average code length of *module codebook* i, weighted by the fraction of time that the random walker spends on module i. Both terms are expressed in terms of the entropy because it provides a lower bound for the average code length needed to describe a single step of the random walk [53].

Each term in the *map equation* is computed using the values of PageRank [55] for each node in the network. The algorithm used to find the optimal partition that minimizes the Eq. 2.12 is similar to the Louvain algorithm described in subsection 2.2.2 and is defined as follows:

1. Initially, each vertex belongs to its own community;
2. For a randomly chosen vertex, move it into the neighbor' module that results in the highest decrease in the *map equation* value;
    (a) If no movement reduces the Eq. 2.12, the vertex remains in its original module
3. The process is repeated until no movement decreases the *map equation*;
4. A new network is built, where each module is represented by a single vertex, and the process of combination of modules starts again, until the *map equation* reaches its minimum;

Similar to the Louvain algorithm, the method scales as $\mathcal{O}(m)$, where m is the number of edges in the network. The infomap was used to identify bioregions of animals from species location data [56].

2.3 Bot Detection in Online Social Networks

This section briefly reviews recent works and methods for the task of identifying automated and inauthentic users in online social networks, investigating their actions in different contexts and how their operational strategy evolved over time to avoid the most recent detection systems.

2.3.1 Hateful People or Hateful Bots?

The usage of automated accounts for spreading hate speech against religious minorities in Arabic tweets has been recently investigated [57]. The goal was to quantify the impact of bots regarding their number of posts and active accounts, as well as to verify the effectiveness of bot detection frameworks trained on English data in detecting automated accounts in the Arabic language.

The work introduces a dataset of 450 accounts, extracted from a sample of 6000 tweets in Arabic Language, manually annotated using a scale from 0 to 5, indicating the likelihood that an account is a bot. Using this dataset, a evaluation of Botometer's performance on classifying Arabic accounts is conducted, and the obtained results suggest that there is need for more specific methodology focused on the Arabic Language.

Thus, the work proposes a method for Arabic bots classification, using features derived from content, like average length of tweets and mentions; proportion of replies, retweets and original posts; average sentiment score of posts and user metadata, like number of followers, favorites and account age. The 200 most recent posts of each user were used to build the features for each of them.

Using these derived features, several Random Forest Regressors were trained, using different combinations of features groups, considering only characteristics extracted from content; combining content features with user metadata, and so on.

The best model reported exhibits a Spearman and Kendall correlation coefficients of 0.74 and 0.60 respectively, when its predictions are compared with the manually annotated scores of the dataset. The Botometer obtained coefficients of 0.43 and 0.33, which indicates that the development of a language-specific methodology was advantageous to detect Arabic bots.

The work also investigates the most informative features during the decision process, and found that the average number of mentions, alongside the proportion of replies, retweets and original tweets, were the features exhibiting the highest impact in classifying an account as bot or human. It is worth noticing that the least informative features were the ones related to user metadata, like the presence or absence of a profile description, location and URL link, suggesting that there is no significant distinction between organic and automated accounts in these characteristics.

The study founds that 11% of hateful tweets were authored by bot accounts, and the Arabic bot exhibits some different traits from their English counterparts, with a higher rate of original tweets and a much higher number of followers.

2.3.2 Botometer

The Botometer [9] is a framework used to detect bots in Twitter, leveraging 1150 features from different classes in its detection process. For a given user u, the system extracts information from the following six categories, as described in [58]:

- **User-based features**: user's metadata, such as number of digits in screen name, default profile and account age, as well as the number of original tweets, retweets, mentions and replies per hour, considering the 200 most recent posts of u, which is the limit of return of the corresponding endpoint in the Standard Twitter API.

- **Friends features**: These features are computed considering other accounts that interact with u in some way. These accounts are divided in 4 groups: profiles that u retweets; profiles that u mentions; profiles that retweet u and profiles that mention u. For each group, one extracts the number of distinct languages in the group's tweets, the fraction of users with a default profile, as well as the descriptive statistics regarding the number of followers, friends and tweets distribution, such as mean, median, standard deviation and skewness.

- **Network features**: The framework considers 3 types of networks, based on the users' interactions: retweet, mention, where the nodes correspond to users and an edge (u, v) exists if u retweets v, and hashtag co-occurrence networks, where the different hashtags as treated as nodes, and an edge (h_1, h_2) exists if the hashtags h_1 and h_2 occur in the same tweet, weighted by the frequency of the occurrence. For each network, the framework computes the edge density and clustering coefficient, as well as descriptive statistics for in and out-degree distributions.

- **Temporal features**: Given the 200 most recent posts of the users, the system groups them into 3 categories: original tweets, retweets and mentions. For each group, the associated posts are sorted by their creation date, the time intervals between two consecutive activities are computed, and the descriptive statistics are extracted from these intervals.

- **Content and Language features**: The system extracts statistics regarding the number of words and word entropy distribution from users' most recent posts. Additionally, a Part-of-Speech (POS) Tagging algorithm is applied to each of them, and the descriptive statistics are computed for each tag distribution.

- **Sentiment Features**: The framework applies several sentiment analysis techniques, to quantify English-specific sentiment features, such as *arousal, valence, happiness and polarization* scores.

The original framework was trained using a dataset of 15K manually annotated bot accounts and 16K verified human profiles. The classifier performance was evaluated by computing the ROC AUC in 5-fold cross-validation with different classifiers. The best reported model achieved a mean AUC of 0.95 using the Random Forest algorithm with 100 estimators.

In recent years, some modifications and improvements have been proposed upon the original Botometer model [23], [12]. A reformulation of the original methodology was also introduced in [59]. This alternative framework, known as *BotometerLite*, aims to resolve two major limitations of currently prevailing models: scalability and generalization.

The issues regarding scalability involve the Twitter API rate limits and the huge volume of data continuously generated in social networks, which prohibits real-time evaluation by current models, given their intrinsic high computational complexity.

In turn, the limitations regarding generalization are associated with the constant evolution of automated accounts, whose behavior is constantly upgraded to evade the most recent detection systems. A model exhibiting a great performance in validation sets could easily be deceived by recent unauthentic users.

In order to overcome these limitations, the *BotometerLite* framework employs a methodology that considers only user metadata, giving up most of the other contextual information for improved computational performance. This choice enables an analysis of 8.6M accounts per day using a single credentials set, which is over 200 times the rate limit of the original Botometer framework [9]. Besides, given that each tweet object from the Twitter API has an embedded user object on it, there is no need for additional requests before applying the *BotometerLite* framework, and it also reflects the user behavior at the moment the tweet was retrieved from the API.

The proposed framework was trained using a compilation of 11 publicly available datasets of human and bot accounts, in addition to three new datasets. All datasets are available at Botometer's Bot Repository. Altogether, the compiled database is composed by 94124 bots accounts and 43396 human profiles. The compilation includes datasets ranging from 2011 to 2019, extracted from different contexts, such as politics and financial market, in a effort to capture the evolution of social bots over time.

The work also proposes a more strict evaluation strategy, setting some datasets aside for the training process, and using these holdout datasets for generalization tests. Thus, 8 datasets were considered during the training phase and 4 during the generalization test phase (some datasets were merged because they contain only a single class). Considering the 8 training databases, the work applied a data selection scheme, treating each dataset as a single unit and training a Random Forest model for each combination of these units (247 combinations, yielding 247 candidate models) and evaluating the performance for each of them via ROC AUC score in a 5-fold cross-validation scheme.

The performance of each model for each test dataset was computed, and the work also evaluated the trained models' performance on a sample of 100K random users, measuring their correlation with the predictions of the original Botometer model.

In order to select the final model among the 247 candidates, the work ranks each candidate model by their performance, with the best model in the first place, for each of the 6 tests (AUC score during cross-validation, AUC score on the 4 holdout datasets and correlation with the predictions of the original Botometer model). The selected model was the one that presented the lowest product of the 6 ranks, being considered as the one with the best overall generalization capabilities. The best candidate was trained with only 5 out of 8 available databases.

As a final test to verify whether the proposed methodology would be able to evaluate tweets and users in real time, an offline experiment was designed to assess the classification speed, and the reported results show that the *BotometerLite* framework would be able to evaluate almost 900M tweets per day, which is beyond the average volume of 500M daily tweets, reported in [60].

2.3.3 RTBust

A methodology for detecting automated users based on their retweeting temporal patterns was proposed in [2]. The work introduces a novel visualization for uncover possibly coordinated behaviors of users, as well as an unsupervised technique for detecting groups of automated users retweeting messages in a coordinated fashion.

Given a set of users and its retweets, the proposed visualization consists of a scatter plot where the timestamp of a retweet and the corresponding original tweet are shown in the x-axis and y-axis, respectively. A point near diagonal represent a retweet performed right after the publication of the original tweet, while points away from the diagonal indicates a longer time interval between the two posts. The work reports three suspicious retweeting behaviors, probably associated with automated users: in the first one, the user always retweets a message almost immediately after the original tweet was posted, stopping by a short inactivity interval and starting its retweeting session again; in the second one, the user retweets past messages up to a fixed period of time and stop its current activity session in a very regular behavior, as shown in Fig. 2.3, where a triangular pattern is observed below diagonal.

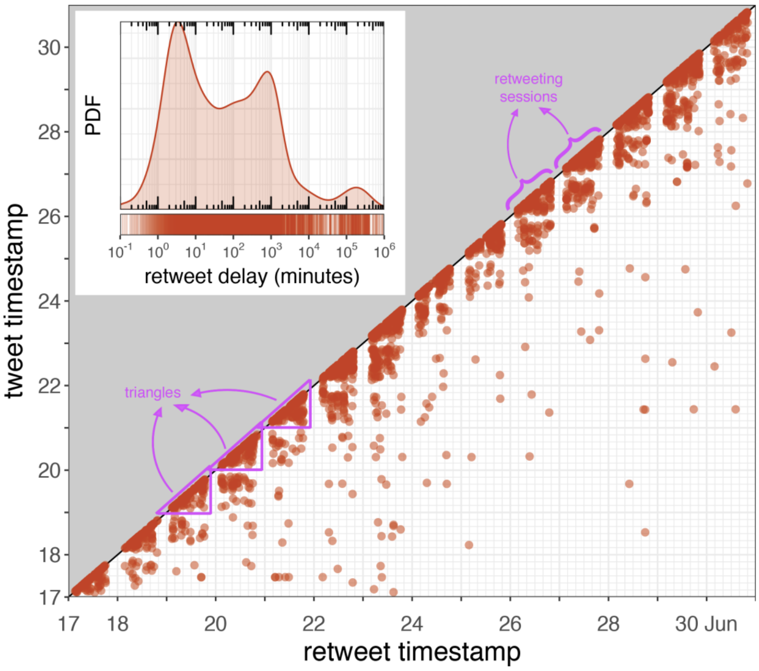

Figure 2.3: Triangular pattern of retweeting activity. Extracted from [2]

The last one is associated with users whose activity goes way back on time, systematically retweeting messages from several days ago, and also exhibiting sessions of activity. Motivated by the observation of these patterns, the work proposes a methodology for detecting retweeting bots, based on their activity time series.

In the first step of the proposed framework, the user u is represented by a time series of retweet intervals $[r_{u,0}, r_{u,1}, r_{u,2}, \dots, r_{u,n}]$, with granularity of 1 second, starting from timestamp t_{REF} = `2018-06-17T00:00:00` associated with $r_{u,0}$. Each component of the time series is computed in the following way: for a given timestamp t, $r_{u,t}$ is equals to $t(x) - t_{REF}$ seconds, if the user u retweeted the message x in time t, or 0, if the user did not retweet any message in time t.

In general, the time series associated with each user will be very sparse. Thus, in the next step of the proposed framework, the work employs a technique to compress sequences of consecutive 0's, substituting them by negative value of their sequence lengths (i. e., a series `[3, 0, 0, 0, 4, 0, 7]` will become `[3, -3, 4, -1, 7]`), generating a compact time series that encodes both inactivity periods and retweeting activities.

The time series of each user is then mapped to a vector of fixed length, using a unsupervised technique known as variational autoencoders. The technique is implemented using two LSTM networks: one will act as the encoder, that compress the input data into a representation in the latent space of features and the other one will be the decoder, which aims to reconstruct the original input, using the latent representation. After training, the encoder will serve as an unsupervised feature extractor for the users' retweeting behaviors.

Using the representations obtained from the encoder, a density-based clustering algorithm known as HDBSCAN [61] is applied to find groups exhibiting common retweeting patterns. All accounts that were clustered by HDBSCAN labeled as bots, and the ones that are not clustered as legitimate accounts.

The work evaluates the performance of the proposed framework using a manually annotated dataset of $\approx$ 1000 accounts classified as bots or humans in a binary classification setting, reporting the results in terms of precision, recall and F1-score, among others. The reported F1-score was 0.87, outperforming Botometer [9] by a large margin (F1 = 0.42) in all evaluated metrics.

Finally, the work reports 2 previously unknown botnets: a group of 44 automated accounts focused on retweeting posts from European car manufactures; and a group of almost 300 bot profiles, that seems to be focused on artificially increasing the popularity and reach of an Italian pop singer.

Chapter 3

Proposed Framework

This chapter presents the proposed framework to detect groups of users that act in a coordinated fashion in online social networks. The framework is organized in two major components, an unsupervised component and supervised component. The main units that compose them are summarized in Fig. 3.1, where the unsupervised component is highlighted in red, while the supervised component is pointed out in blue.

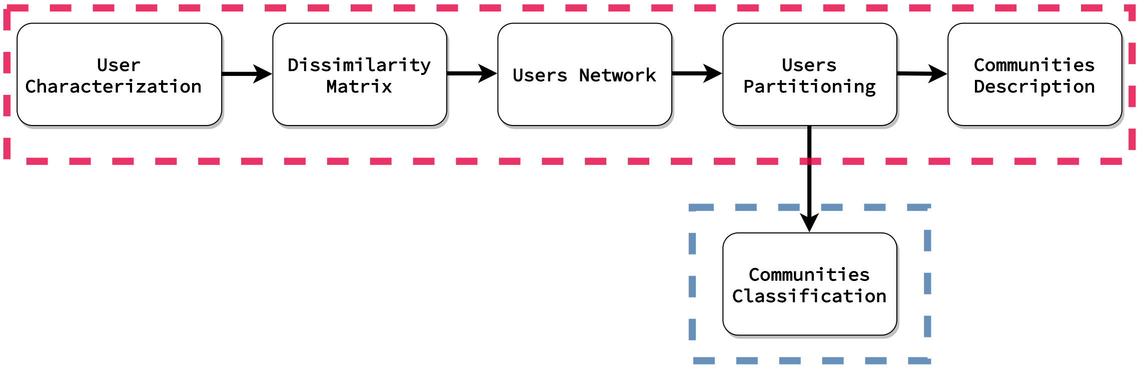

Figure 3.1: Main components of the proposed framework

We start by generating representations of the social network users, using their posts to build vector representations that will feed a language model. Thus, each user will be assigned a language model. In the following step, we establish the degree of similarity among different users, using the representations from the previous stage.

Then, we construct a network that encodes the similarities between the most similar users, using the values computed in the previous step. Thereon, we divide the network into communities, using only the network structure to uncover the set of users that coordinate their posts. We proceed by automatically extracting a description of the main language topic circulating inside each community, combining the users' language models and their structural importance in the group.

Finally, we categorize the kind of coordination of the community, distinguishing between automated (`bot`) and organic (`human`) synchronized actions, using information from the community language model as well as the structure of the relationship of users inside the community. Each component of the framework is presented in the following sections.

3.1 User Characterization

The starting point of our framework generates representations of users, using the language models discussed in the previous Chapter. In our framework, we employ the unigram model to characterize each user, using its posts to parametrize the language model, which corresponds to the simplest *n-gram model* with $n = 1$. In this model, $P(W)$ will be defined as the product of relative frequency of each word $w_i \in W$, as shown in Eq. 3.1:

$$P(W) \approx \prod_{w_i \in W} P(w_i) \tag{3.1}$$

Thus, the language model of user u (LM_u) will be represented by a vector $P_u = [P(w_1), P(w_2), \ldots, P(w_T)]$, where T is the vocabulary size of the whole corpus of posts, considering all users in the dataset. Moreover, $P_u(w_i)$ for user u is given by the relative frequency of the word w_i across all words posted by user u in the corpus.

3.2 Dissimilarity Matrix

The language model for each user can be interpreted as a probability distribution over the vocabulary defined by users' posts. Thus, the LMs can be used to compute a distance between users. In our framework, we employ the Jensen-Shannon (JS) Divergence [62] as our dissimilarity measure. The JS Divergence is a symmetric version of the Kullback-Leibner (KL) Divergence [63], limited in the interval $[0, 1]$, as defined in Eq. 3.2.

$$\begin{aligned} D_{KL}(p \parallel q) =& \sum_x p(x) \log \frac{p(x)}{q(x)} \\ D_{JS}(p \parallel q) =& \frac{1}{2} D_{KL}\left(p \,\middle\|\, \frac{p+q}{2}\right) + \frac{1}{2} D_{KL}\left(q \,\middle\|\, \frac{p+q}{2}\right) \end{aligned} \tag{3.2}$$

where p and q are probability distributions over the same sample space (in our case, two language models over the same vocabulary). We use the JS Divergence to compute the dissimilarities matrix between pairs of users, in order to establish a degree of similarity among them. In particular, the similarity between users u and

v is given by $D_{JS}(P_u \parallel P_v)$, where P_u an P_v are the probability distribution for the language models of users u and v, respectively.

3.3 Users Network

Given the similarities among users, we proceed to the next step of our framework, generating a multilayer network that encodes similarities among users' language models. In this kind of network, each layer represents a different type of relationships between nodes, and all layers shares the same set of vertices.

The nodes will correspond to the set of users in the dataset, and an edge (u, v) will exists between users u and v if and only if the user v is among the T most similar users from the point of view of user u, according to the JS Divergence computed in the previous step. Note that the out-degree of every node is exactly T. Moreover, an edge (u, v) does not imply an edge (v, u), as v may not have u among its T most similar users.

Lower values of parameter T leads to a stronger signal of similarity and coordination among accounts, while higher values generates more connected networks, with a possibly clearer community structure. In our case, each layer will represent the edges associated with a different value of T, ranging from 1 to L. Note that all edges in the network of layer T are present in the network of layer $T + 1$. Thus, as T increases the network becomes denser.

3.4 Users Partitioning

The next step of the framework aims to detect the coordinated groups present in the dataset. We formulate the task as a community detection problem in multilayer networks. As previously mentioned, different values of parameter T leads to networks with different and desirable characteristics, like stronger signal of coordination and clearer community structure.

The goal is to leverage these traits during the process of community detection. Thus, we employ the multilayer version of the Louvain algorithm for network community detection using a network with L layers [64]. Similar to the version present in Chapter 2, the multilayer Louvain algorithm aims to find the partition of vertices that maximizes the *modularity* metric. In the multilayer setting, the quality of a particular partition of users will be defined as a weighted sum of the *modularity* values in each layer (since each of them is a network), as shown in Eq. 3.3:

$$\mathcal{Q}(C) = \sum_k w_k q_k(C) \tag{3.3}$$

where w_k and $q_k(C)$ corresponds to the weight and the *modularity* value of the user partition C in the k-th layer, respectively. Note that the partition C is common to all layers, and that $C = (C_1, \ldots, C_k)$ divides the users into k communities.

The algorithm in the multilayer variation is essentially the same: the vertices are moved to the community providing the highest *modularity* gain, considering the individual contribution of each layer. A new network is built, where each community will be represented as a single vertex, and the process is repeated, until no further improvement can be achieved (see details in Section 2.2.2).

We explore the effect of different values of L (number of layers) as well as different weight schemes (values for w_k) to ponder the contribution of each layer in a later chapter. Intuitively, these parameters will play a role in the effectiveness of the framework.

3.5 Communities Description from User's Language Models

The next step of the framework defines a method to identify the main topic discussed in each community. It works as follows:

- The induced subgraph with the subset of nodes belonging to a community is extracted from the last layer of the network (layer L);

- The centrality of each node in the induced subgraph is computed, using the PageRank (PR) algorithm [55];

- The language model of the community is defined as a linear combination of the LM of each user in the community, weighted by user's centrality. In particular, the language model of community C_k will be defined as:

$$LM_{C_k} = \sum_{u \in C_k} PR(u) \cdot LM_u \tag{3.4}$$

  where $PR(u)$ and LM_u represent the centrality and the language model of user u in the community, respectively. After that, we normalize LM_{C_k} in order to obtain a well-defined probability distribution over the corpus vocabulary.

- The five highest probability words in the language model of the community LM_{C_k} are extracted. These words will be taken as the community description.

3.6 Communities Classification

The final step of the framework is to classify the communities as `bot` or `human` acting as a coordinated group. The communities classified as `bot` correspond to synchronized groups of automated accounts, coordinating their posts to often promote and artificially increase the popularity of a specific content or topic. On the other hand, the communities labeled as `human` correspond to organic groups of users engaged with a common cause, whose actions (posts) are similar due to their common interests.

Features extracted from the community language model, alongside features extracted from the network structure, are used as the input to a Random Forest model [65], using the majority class of each community as its label. Thus, a supervised classification procedure is designed using labeled data for each user in the dataset (bots and humans). Note that the goal here is to classify a community and not individual users. The training details and evaluation of the classifier will be presented in a later chapter.

Chapter 4

Empirical Analysis of the Unsupervised Component

This Chapter presents an analysis of the unsupervised component of the proposed framework using real data. We briefly introduce the two bot datasets used throughout the experiments and submit them to the first steps of the methodology, highlighting aspects like the similarities between user's language models and the structural differences between the two classes of users. We also analyze the impact of several weight schemes and number of layers in the final quality of users partitions, using a metric of homogeneity to assess the quality of a community. Finally, we investigate the main topic that circulated in the communities identified by the framework in both datasets.

4.1 Datasets

Two datasets of labeled bot and humans accounts in Twitter as well as tweets of such accounts are considered in the following evaluation, both of which are publicly available in Botometer's Bot Repository [1].

4.1.1 cresci-stock

The first dataset, that we refer as `cresci-stock` throughout the following sections, consists of bot accounts performing financial market coordinated campaigns, named by the authors as *cashtag piggybacking*, in order to artificially promote low-value stocks alongside high-value ones on Twitter [66]. The work reports that there's a positive correlation between the market value of a stock and its presence in social media discussions, so interest groups could use automated accounts to increase the stock price of target companies.

[1] https://botometer.osome.iu.edu/bot-repository/datasets.html

In the monitored period, large coordinated groups of bots were mobilized to perform massive publications of cashtags (a tag associated to a company or stock, like `$APPL` for Apple, Inc. and `$AMZN` for Amazon.com, Inc.) where both high and low-value stock tags were present together. The cashtags are visually highlighted on Twitter's interface, capturing the attention of users and potential investors.

The authors collected $\sim 9M$ tweets containing cashtags from a list of 6689 stocks traded on US Markets between May and September 2017, posted by $\sim 2.5M$ distinct user accounts. After that, a subsample of 25957 accounts was labeled as either bot or human account using a bot spam detection system [67].

Due to limited computational resources, we perform an additional subsample, randomly selecting a subset of 2000 accounts labeled as bots and 2000 labeled as humans (all accounts of each class had the same probability of being chosen), responsible for 105062 tweets.

4.1.2 rtbust

The second dataset, introduced in [2] and referenced here as `rtbust`, consists of all Italian retweets posted between 17 and 30 June of 2018, containing 9989819 messages from 1446250 distinct user accounts. The original work proposed a framework for detecting coordinated behavior of bot accounts based on temporal patterns of retweets, as discussed in Chapter 2.

The original dataset was filtered to only include users whose mean number of retweets per days was between 2 and 50. After this preprocessing, the original dataset was reduced to 63762 distinct accounts. Finally, a subsample of almost 1000 users was taken, and then the selected accounts were manually annotated by the authors [2].

The available version of dataset, obtained from Botometer's Bot Repository, contains 759 labeled users, alongside their 95777 retweet ids. Using the Standard Version of Twitter API [2], we were able to retrieve 74014 posts, associated with 643 of the 759 annotated users in the dataset.

Table 4.1 reports the descriptive statistics of the number of tweets per user in each dataset, and Figure 4.1 shows their respective Empirical Complementary Cumulative Distribution Function (ECCDF) plots, where it is evident that both datasets exhibit a heavy-tail characteristic in the distribution of the number of tweets per user account.

[2] https://developer.twitter.com/en/docs/twitter-api/v1

Table 4.1: Descriptive statistics for tweets per user in each dataset

	max	min	mean	median
`cresci-stock`	7341	1	26.3	6
`rtbust`	598	1	115.1	66

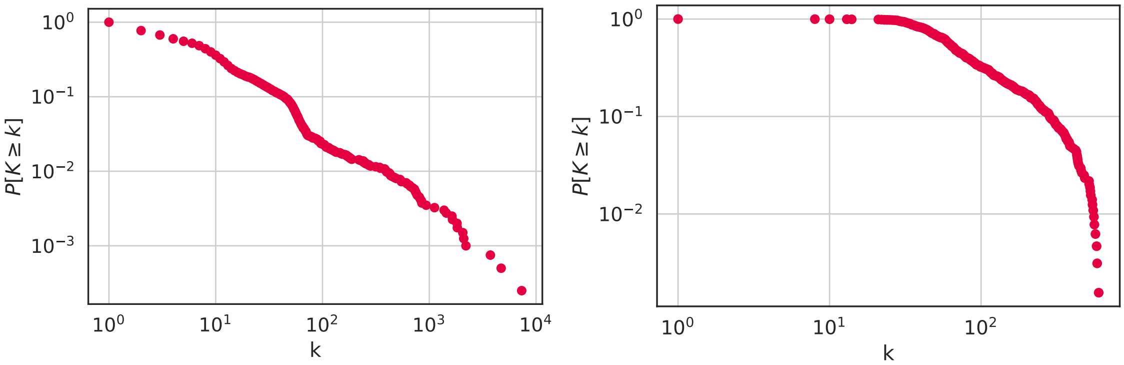

Figure 4.1: ECCDF of the number of tweets per user for `cresci-stock` (left) and `rtbust` (right) datasets.

4.2 Dissimilarity Matrices and User Networks

We applied the proposed framework upon the two datasets. Figure 4.2 shows a fragment (subset of user accounts) of the computed dissimilarity matrix for each dataset, based on the language models for each user account. Due to the symmetry property of the JS Divergence, we report only the lower diagonal of each matrix.

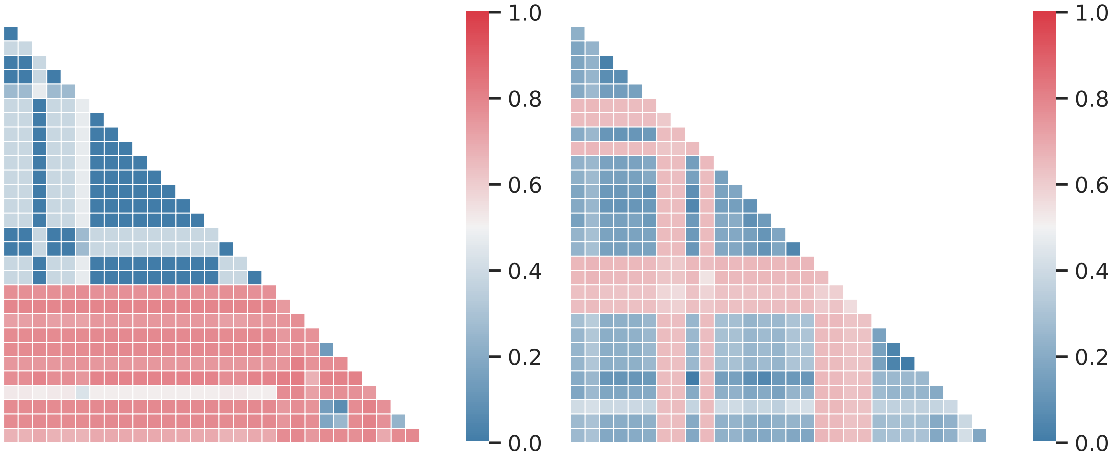

Figure 4.2: Fragments of dissimilarity matrices for `cresci-stock` (left) and `rtbust` (right) datasets.

Each matrix element (denoted as a single square) represents a pair of users, and its color encodes the dissimilarity value between them. A hue towards red indicates a relatively high dissimilarity (near 1.0) between users' language models (constructed using all tweets posted by the user), while a hue towards blue indicates a low dissimilarity (near 0.0), representing users with aligned content in social media.

Considering the matrix associated with the `cresci-stock` dataset, we observe some pairs of similar users in the upper part of the fragment. The same can be observed for `rtbust`, where many pairs of users have a similar (and in some cases, almost identical) language model. This trait indicates that some accounts could be posting and/or sharing the same content during the period that comprises each dataset, possibly in an orchestrated way.

Using the computed dissimilarities among users, we built the directed user network. Figures 4.3, 4.4 and 4.5 show the resulting networks for `cresci-stock` when each user connects to its 1, 2 and 3 most similar users, respectively. Blue nodes represent users labeled as human in the dataset, while red nodes corresponds to bot accounts. Figures 4.6, 4.7, 4.8 show the corresponding networks for `rtbust`.

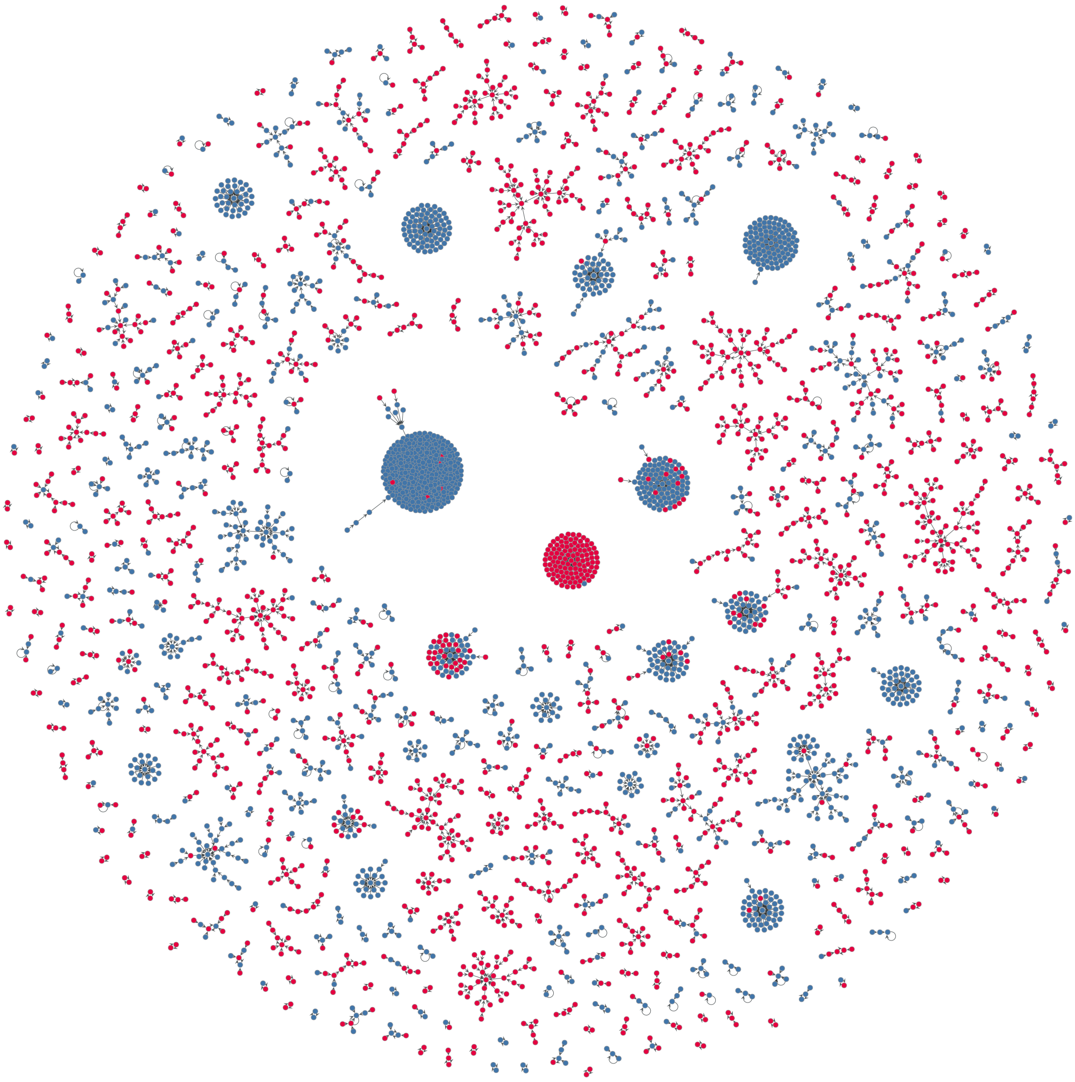

Figure 4.3: cresci-stock users network ($L = 1$)

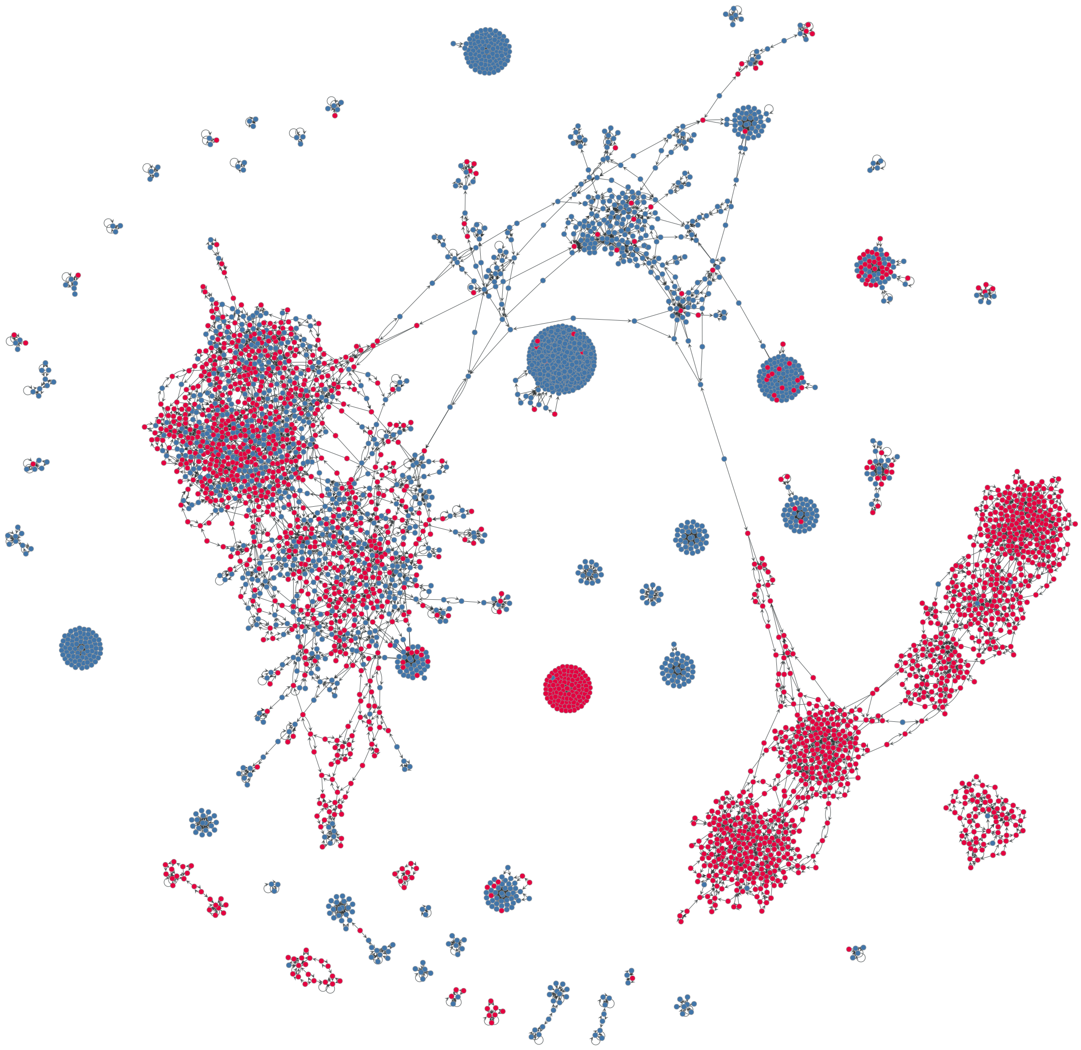

Figure 4.4: cresci-stock users network ($L = 2$)

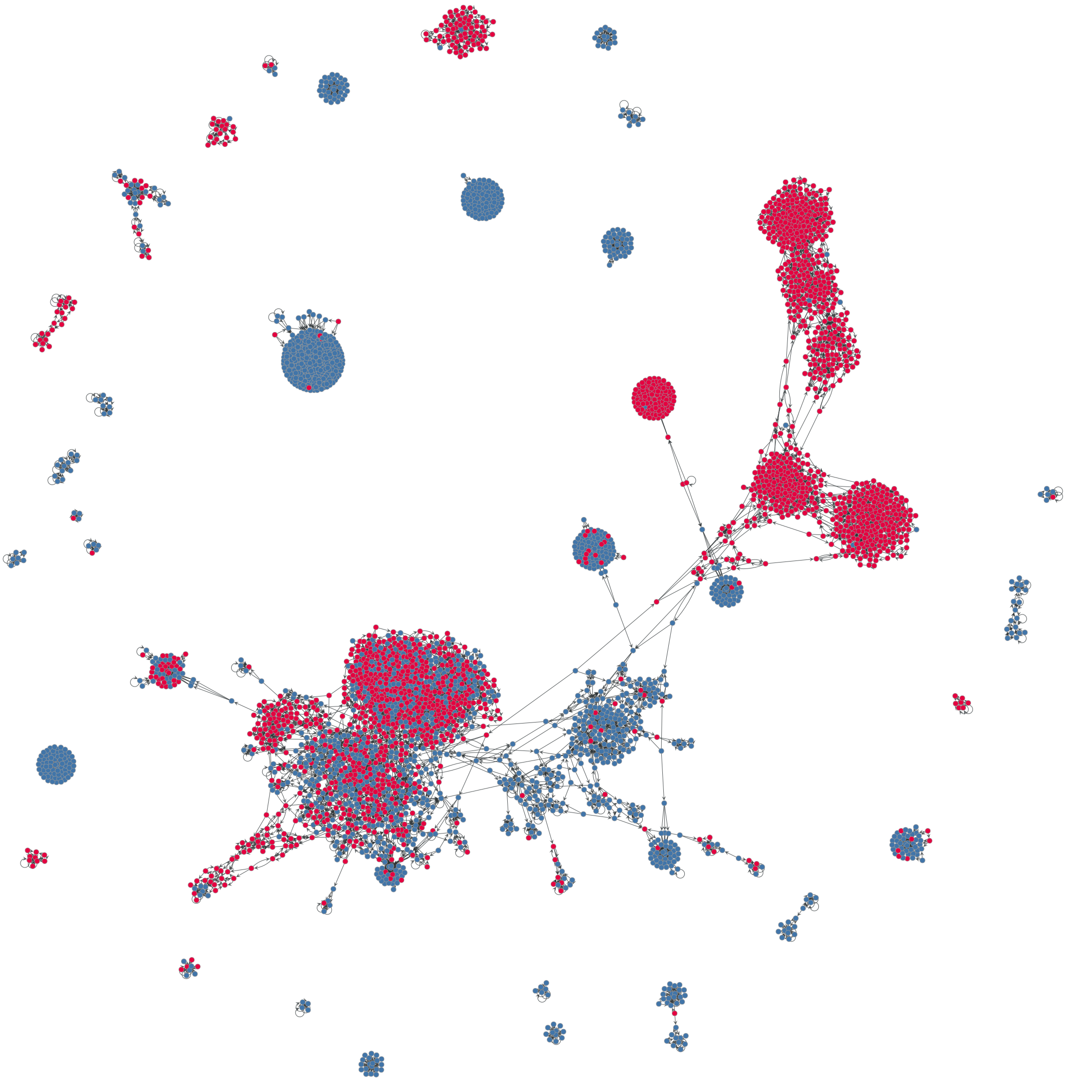

Figure 4.5: cresci-stock users network ($L = 3$)

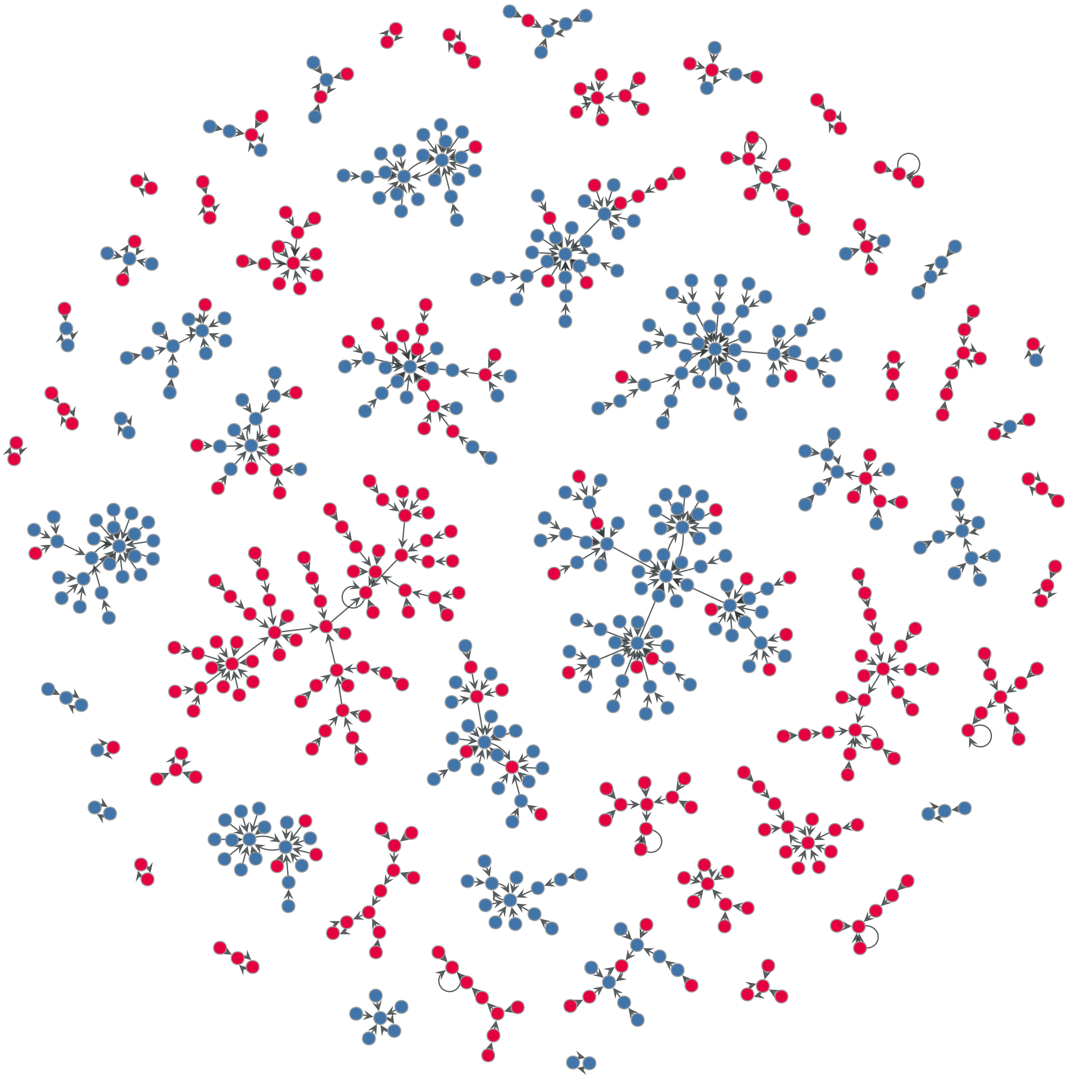

Figure 4.6: RTBust users network ($L = 1$)

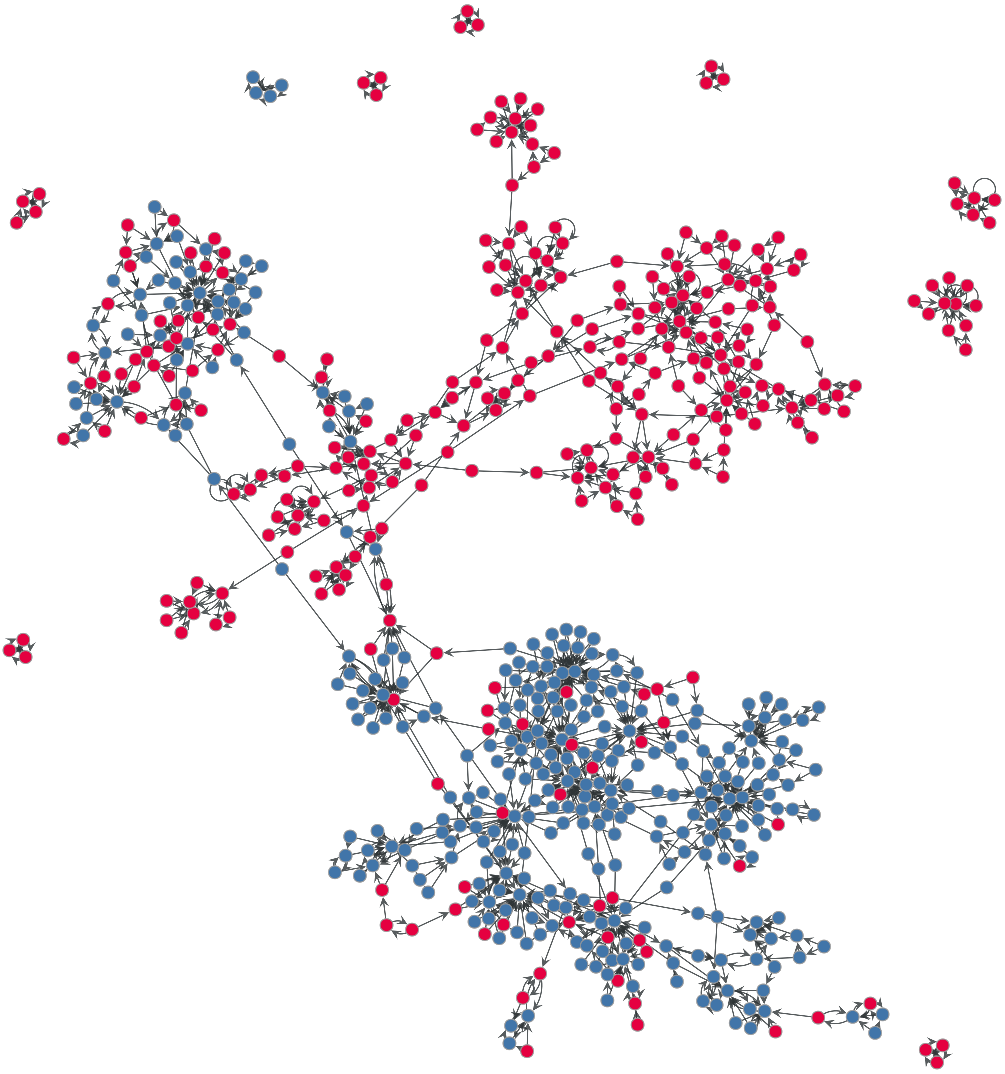

Figure 4.7: RTBust users network ($L = 2$)

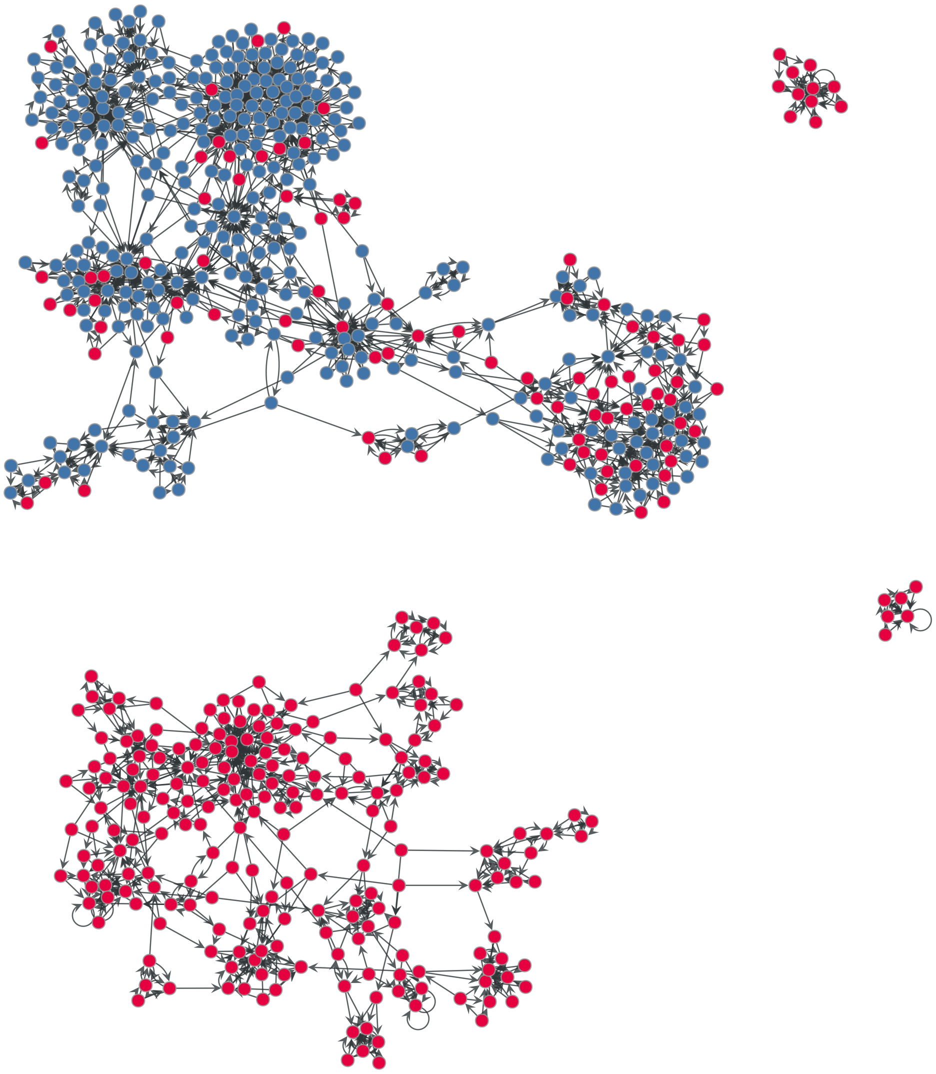

Figure 4.8: RTBust users network ($L = 3$)

Lower values of parameter L represent networks with a stronger signal of similarity and a larger number of connected components, while higher values generate denser networks and fewer connected components. Note that some connected components exhibit a clear majority user label, which emerges directly from the process of building of the networks. For example, the `rtbust` network with $L = 3$ yields 4 connected components, three of which are composed of only bot accounts, and the other is mostly composed of human accounts (see Figure 4.8). Moreover, some connected components are clearly formed by distinct communities, such as the `cresci-stock` network with $L = 2$ (see Figure 4.4), where a larger number of bot accounts (in red) is connected to the other nodes in the component through a single node. This characteristic suggests that representing a user by its linguistic features (using a Language Model, in our case) can be useful to differentiate among the different types of coordinated behaviors.

4.2.1 Structural analysis of induced subgraphs of bots and humans

Before proceeding to the next step of the framework, we investigate structural network characteristics of the different types of user. Given the networks using 1, 2, and 3 nearest-neighbors for each dataset, we extract two induced subgraphs: the first one preserves the edges between users labeled bots, while the second one preserves the edges between users labeled humans. The remaining types of edges (human → bot and bot → human) were removed from each subgraph. In our analyses, we consider two structural metrics: the edge reciprocity and the global clustering coefficient.

The edge reciprocity metric (ER) [68] computes the ratio between the number of edges in both directions (i.e. (u_1, u_2) and (u_2, u_1), where u_1 and u_2 represent different users) and total number of edges in the network. Its value can be interpreted as the probability that the edge (u_1, u_2) exists in the network, given the existence of the edge in the opposite direction, which could indicate some sort of coordination between pairs of accounts in our case, as two users whose LMs are the most similar to each other will be connected in both directions.

The second metric, the global clustering coefficient (GCC) [69] is defined as:

$$GCC = 3 * \frac{\text{number of triangles in the network}}{\text{number of connected triples of vertices}} \tag{4.1}$$

This metric can be interpreted as the probability that two vertices that share a neighbor are also neighbors, which could also indicate a degree of coordination among triplets of vertices. For this metric, the direction of the edge has been ignored.

We have also computed descriptive statistics of the in-degree distribution of each subgraph (recall that all nodes have the same out-degree, given by L). Tables 4.2, 4.3, 4.4 show the results for the subgraphs of each class extracted from `cresci-stock` networks with $L = 1$, 2 and 3, respectively. Figures 4.9, 4.11, 4.13 show the ECCDF for in-degree distribution of each subgraph, and Figures 4.10, 4.12, 4.14 provide visualizations for each induced subgraph.

Table 4.2: Structural analysis of induced subgraphs - `cresci-stock` ($L = 1$)

	ER	**GCC**	**in-degree distribution**			
			max	**min**	**mean**	**median**
humans	0.124	0	394	0	0.97	0
bots	0.252	0	109	0	0.95	0

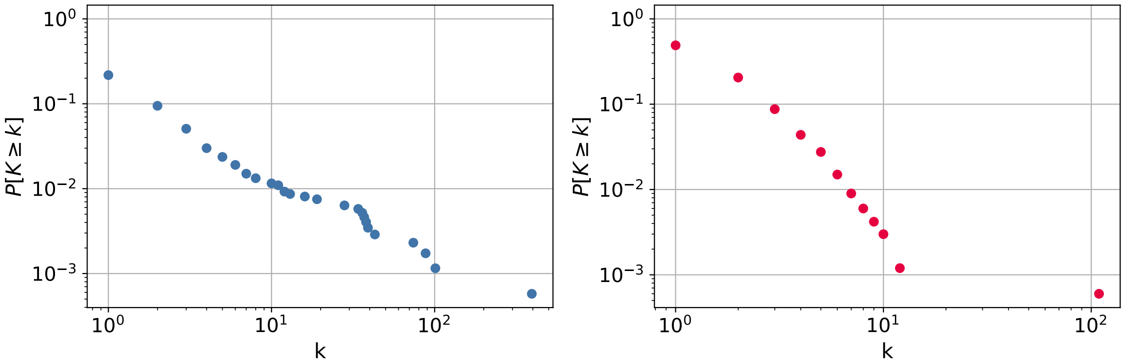

Figure 4.9: ECCDF for in-degree distribution of induced subgraphs of human (left) and bot accounts (right) - `cresci-stock` ($L = 1$)

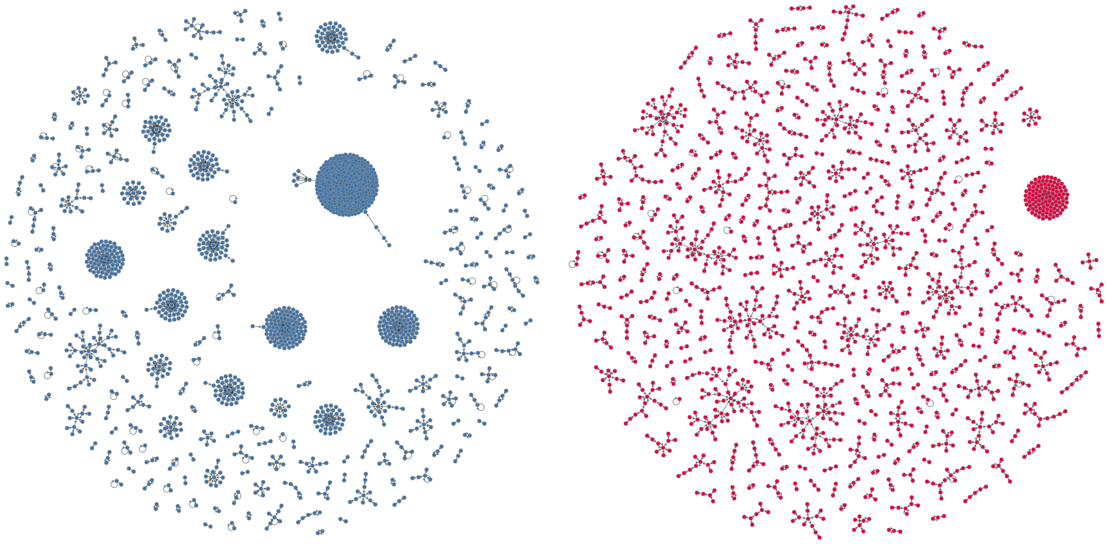

Figure 4.10: Induced subgraphs of human (left) and bot accounts (right) - `cresci-stock` ($L = 1$)

Table 4.3: Structural analysis of induced subgraphs - `cresci-stock` ($L = 2$)

	ER	**GCC**	**in-degree distribution**			
			max	**min**	**mean**	**median**
humans	0.161	0.029	394	0	1.76	0
bots	0.304	0.057	109	0	1.72	1

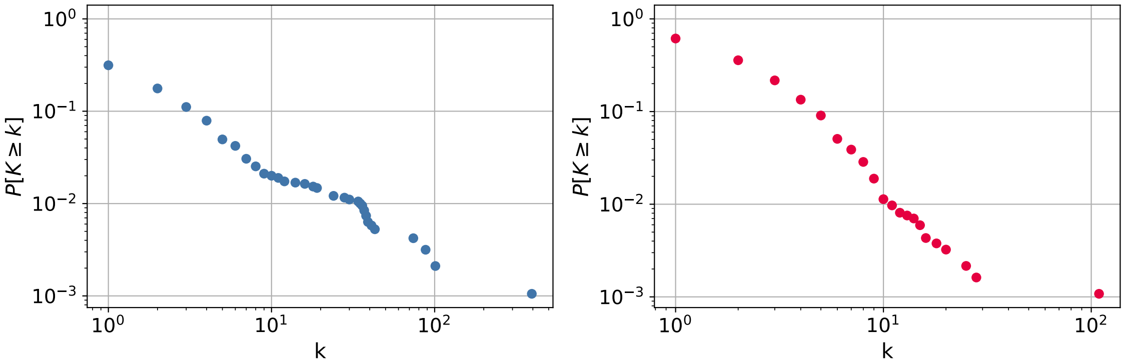

Figure 4.11: ECCDF for in-degree distribution of induced subgraphs of human (left) and bot accounts (right) - `cresci-stock` ($L = 2$)

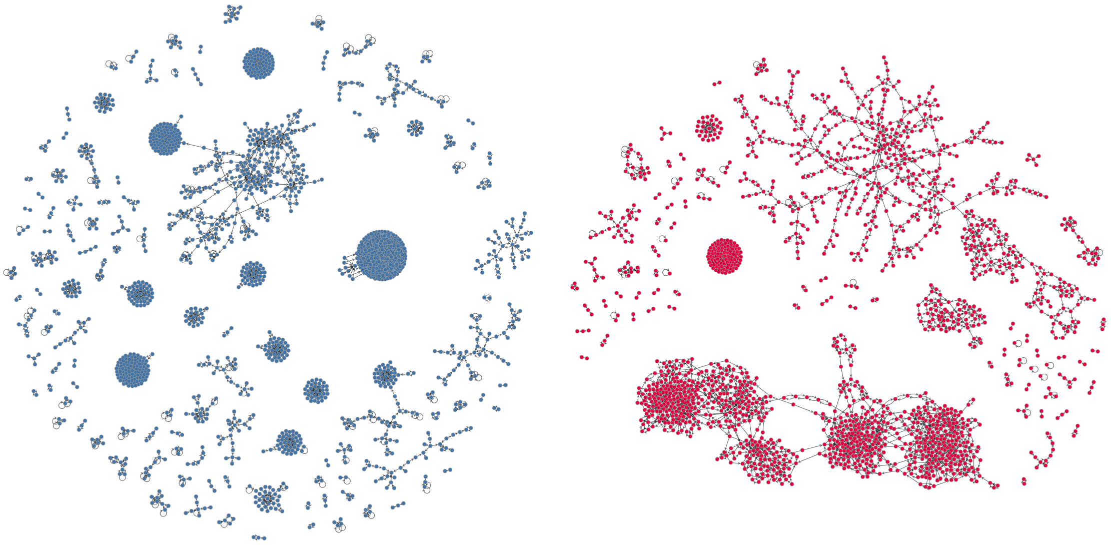

Figure 4.12: Induced subgraphs of human (left) and bot accounts (right) - `cresci-stock` ($L = 2$)

Table 4.4: Structural analysis of induced subgraphs - `cresci-stock` ($L = 3$)

	ER	GCC	in-degree distribution			
			max	min	mean	median
humans	0.177	0.057	394	0	2.55	0
bots	0.334	0.105	109	0	2.52	1

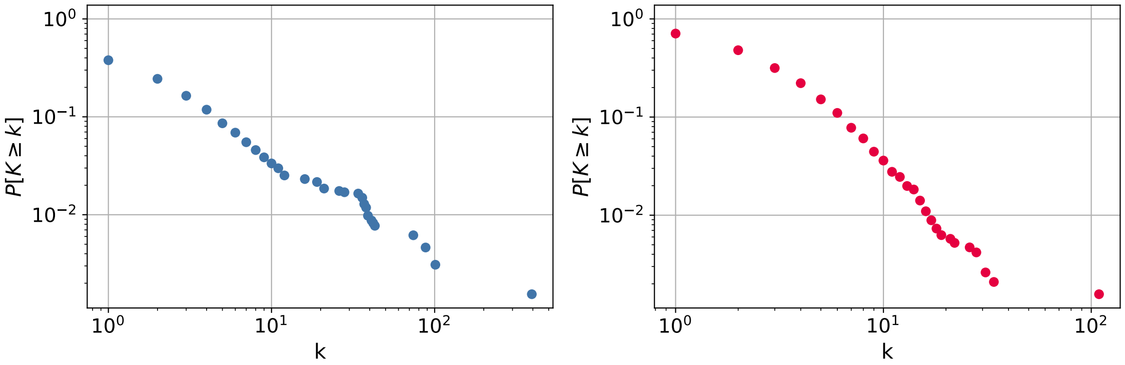

Figure 4.13: ECCDF for in-degree distribution of induced subgraphs of human (left) and bot accounts (right) - `cresci-stock` ($L = 3$)

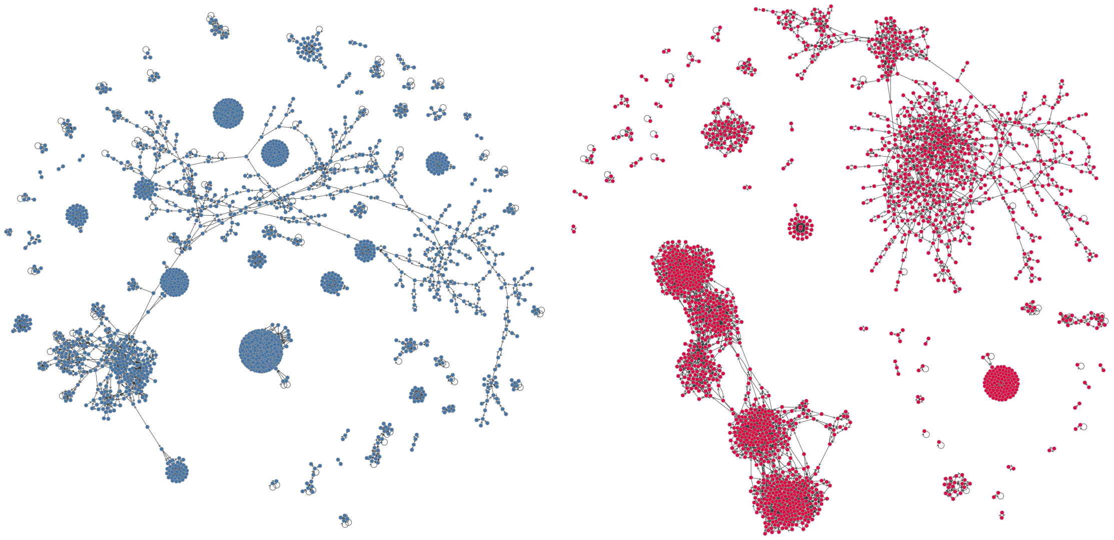

Figure 4.14: Induced subgraphs of human (left) and bot accounts (right) - `cresci-stock` ($L = 3$)

Similarly, Tables 4.5, 4.6, 4.7 show the metrics for the induced subgraphs of each class extracted from `rtbust` networks with $L = 1$, 2 and 3, respectively. Figures 4.15, 4.17, 4.19 show the ECCDF for the in-degree distribution of each subgraph, and Figures 4.16, 4.18, 4.20 provide visualizations for each induced subgraph.

Table 4.5: Structural analysis of induced subgraphs - `rtbust` ($L = 1$)

	ER	**GCC**	**in-degree distribution**			
			max	**min**	**mean**	**median**
humans	0.136	0	19	0	0.97	0
bots	0.188	0	9	0	0.95	0

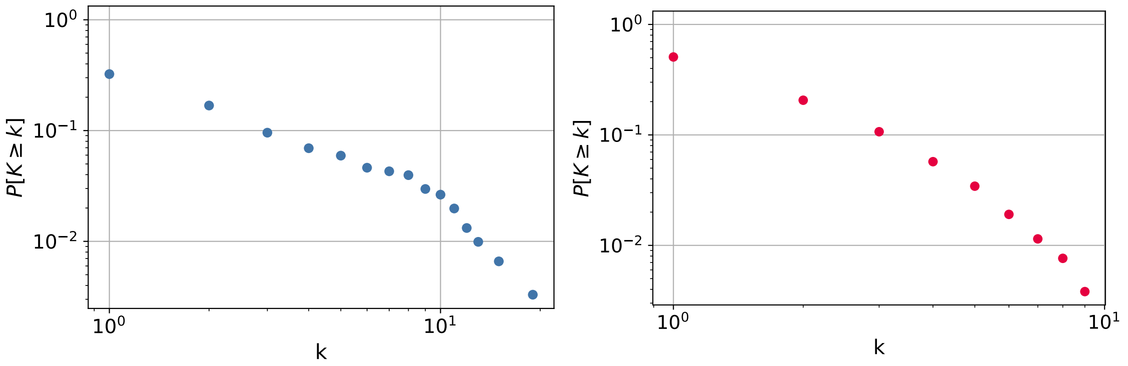

Figure 4.15: ECCDF for in-degree distribution of induced subgraphs of human (left) and bot accounts (right) - `rtbust` ($L = 1$)

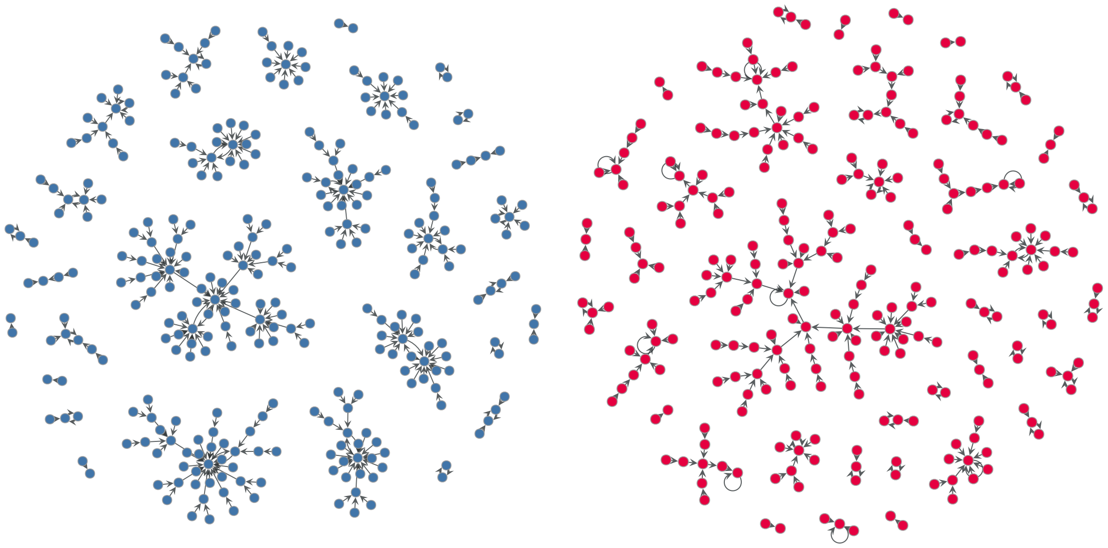

Figure 4.16: Induced subgraphs of human (left) and bot accounts (right) - `rtbust` ($L = 1$)

Table 4.6: Structural analysis of induced subgraphs - `rtbust` ($L = 2$)

	ER	**GCC**	**in-degree distribution**			
			max	**min**	**mean**	**median**
humans	0.168	0.148	27	0	1.80	0
bots	0.256	0.264	12	0	1.79	1

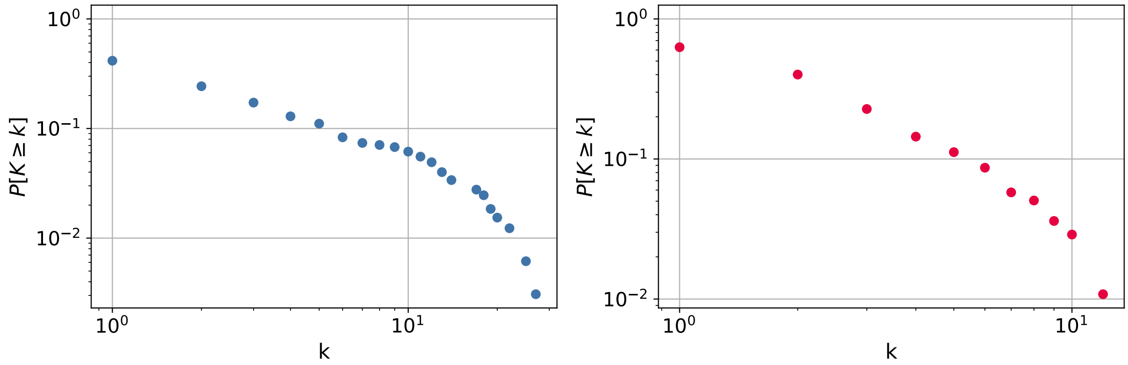

Figure 4.17: ECCDF for in-degree distribution of induced subgraphs of human (left) and bot accounts (right) - `rtbust` ($L = 2$)

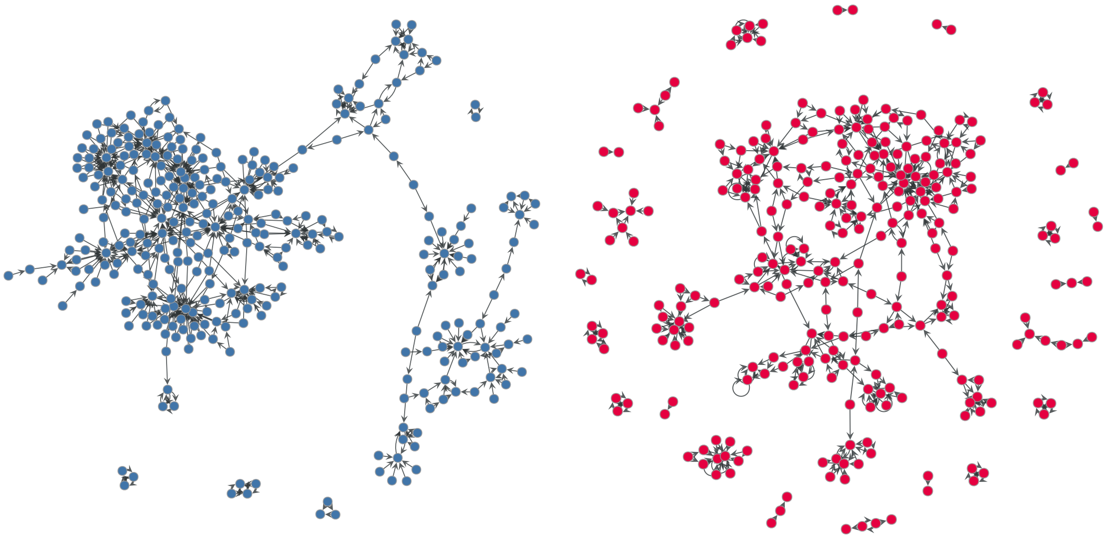

Figure 4.18: Induced subgraphs of human (left) and bot accounts (right) - `rtbust` ($L = 2$)

Table 4.7: Structural analysis of induced subgraphs - `rtbust` ($L = 3$)

	ER	**GCC**	**in-degree distribution**			
			max	**min**	**mean**	**median**
humans	0.176	0.196	35	0	2.67	0
bots	0.285	0.355	20	0	2.63	2

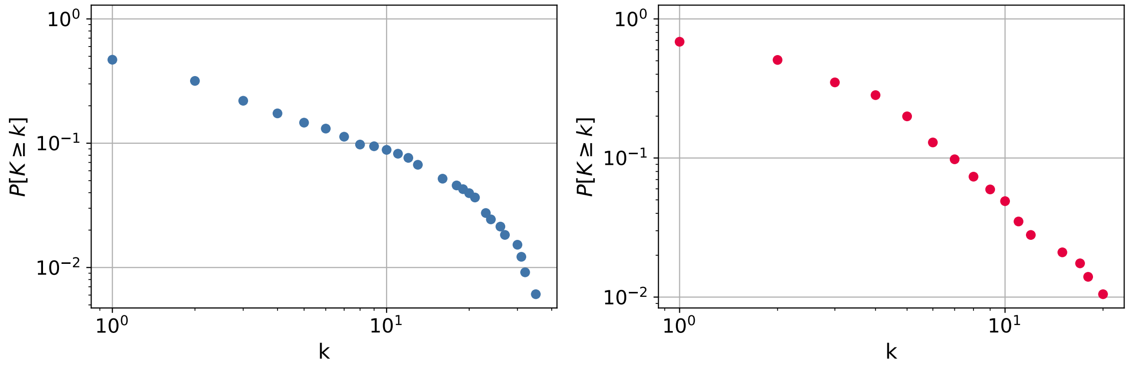

Figure 4.19: ECCDF for in-degree distribution of induced subgraphs of human (left) and bot accounts (right) - `rtbust` ($L = 3$)

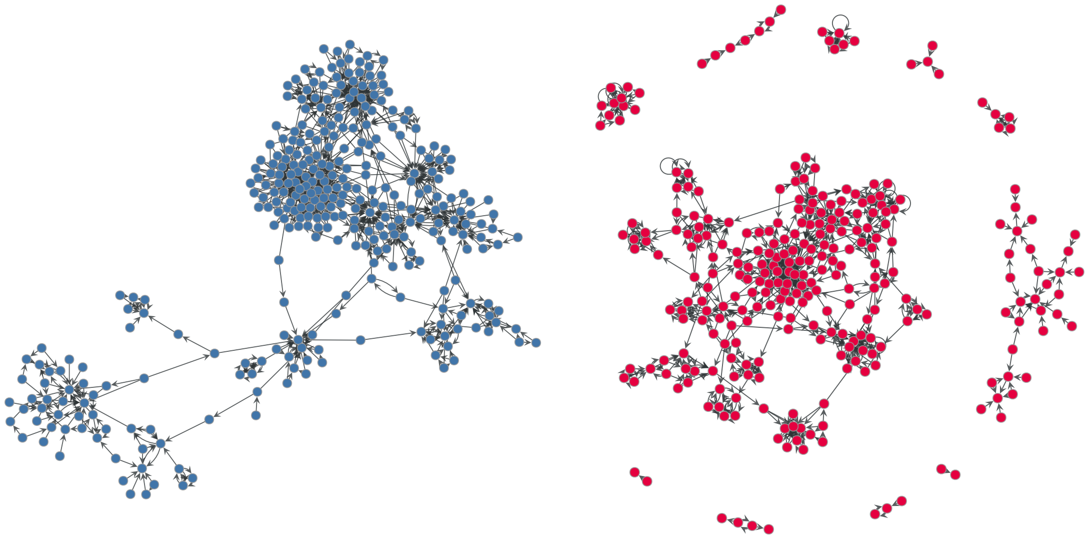

Figure 4.20: Induced subgraphs of human (left) and bot accounts (right) - `rtbust` ($L = 3$)

For all analyzed networks, the ER and GCC metrics in the Bots subgraphs were consistently higher than their counterparts in Human subgraphs, being twice larger for the `cresci-stock` networks. This observed behavior suggests that the automated accounts were indeed acting in a coordinated way, by posting more similar content among them than humans post in the analyzed time interval.

Interestingly, the in-degree distribution of bots has a tail that is shorter (lighter) than that of humans, in both networks and for all values of L. This suggests that bot accounts are possibly acting in a distributed fashion, where the accounts divide the similar posts among them, as opposed to post the same content of a few accounts. This characteristic may indicate that these bot users were at service of diverse interests, popularizing different stock cashtags or retweeting content about several topics during the analyzed period in `cresci-stock` and `rtbust` datasets, respectively.

4.3 Purity scores and multilayer community detection

Proceeding with the framework, the next step generates partitions of users into coordinated groups in a unsupervised way, using the multilayer version of the Louvain algorithm for community detection in networks.

As mentioned in Chapter 3, we have to choose the maximum number of layers in the network as well as the contribution (weight) of each layer during the partitioning process. We analyzed different settings for these parameters and evaluated the quality of the different partitions using a metric for homogeneity that we call *purity score*.

The purity score computes the fraction of users belonging to the majority class in a given community. For example, consider a community of 5 users, where 4 users belong to the c_1 class, and the remaining one belongs to c_2 class. Thus, the purity score will be $\frac{4}{5} = 0.8$ in this case. As we are aiming to obtain highly homogeneous groups with respect to the class of their members (bots or humans), therefore highlighting different types of coordinated actions, we employ the purity score as a metric for choosing the best parameter configuration.

The following weight schemes for the multilayer community detection were evaluated (each scheme defines the contribution of the k-th network layer):

- `square`: $\dfrac{1}{k^2}$
- `linear`: $\dfrac{1}{k}$
- `sqrt`: $\dfrac{1}{\sqrt{k}}$
- `uniform`: 1

As aforementioned, the edges from layers with lower values of k provides a stronger signal of similarity among user's language models, while higher values of this parameter generates denser networks, providing a clearer community structure, facilitating the identification of communities within the connected components that exhibit traits of content coordination. Note that the `square` scheme is the one that penalizes higher layers (larger values for k), while the `uniform` scheme gives the same weight to all layers.

Figures 4.21, 4.22, 4.23 and 4.24 show purity and community results considering 5, 10, 15 and 20 layers per dataset. The x-axis represents the different weight schemes evaluated, and in the y-axis we have the median of the purity score (across all communities), after labeling the communities by the majority class of users present in each of them. Considering this division, we can verify whether the Louvain algorithm can better identify the coordinated groups of a particular class of users or whether it can successfully partition both types of users into highly homogeneous communities.

The red and blue points encodes the median purity score for communities labeled as bots and humans, respectively. Finally, the size and the annotation near each point provides the number of communities of that type.

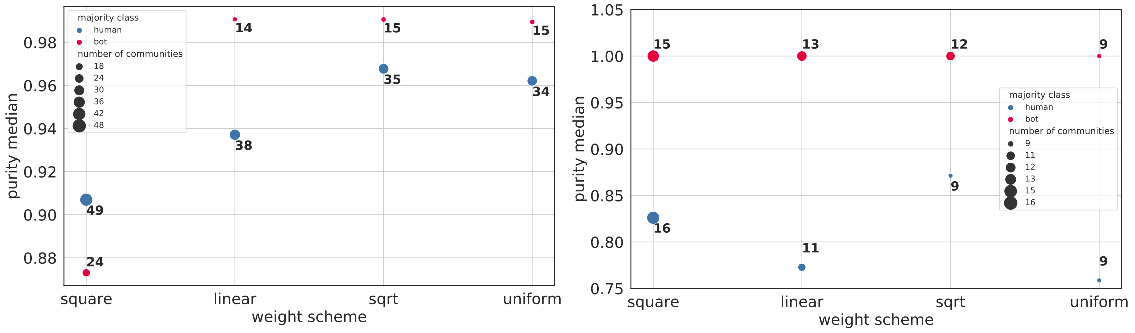

Figure 4.21: Median purity scores for different weight schemes in the `cresci-stock` (left) and `rtbust` (right) datasets - 5 layers

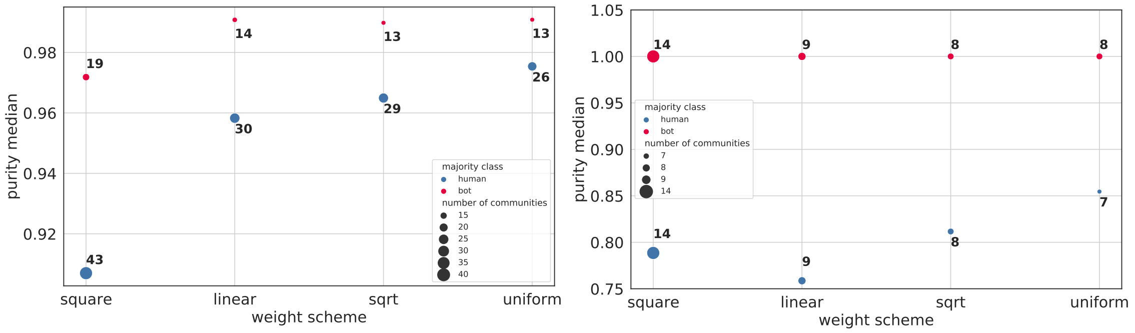

Figure 4.22: Median purity scores for different weight schemes in the `cresci-stock` (left) and `rtbust` (right) datasets - 10 layers

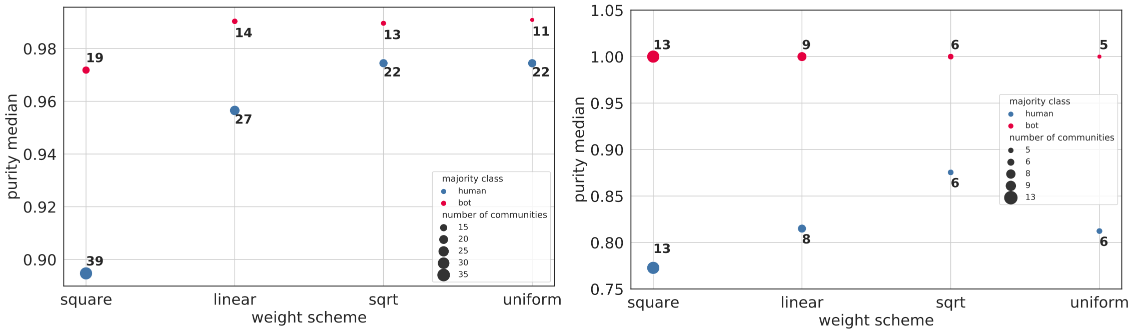

Figure 4.23: Median purity scores for different weight schemes in the `cresci-stock` (left) and `rtbust` (right) datasets - 15 layers

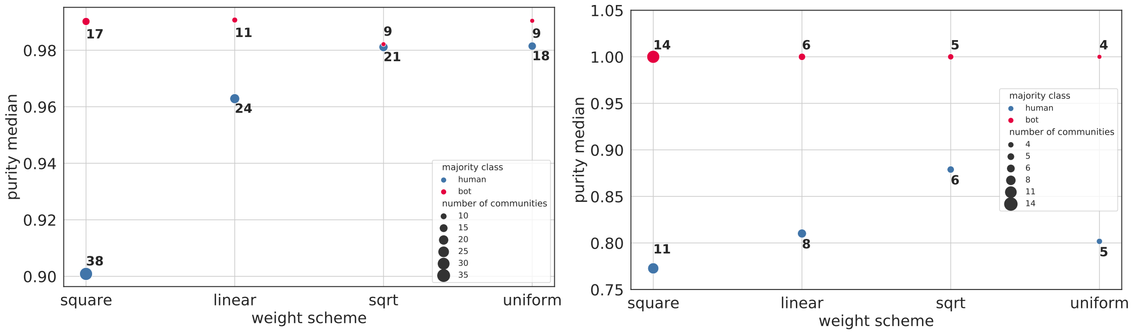

Figure 4.24: Median purity scores for different weight schemes in the `cresci-stock` (left) and `rtbust` (right) datasets - 20 layers

The results indicate that the purity of bot communities is significantly higher than human communities, for both datasets and different weight schemes. A possible explanation of this result is that the `bot` communities are highly focused on their target content, while `human` communities post about more diverse topics, that can also be the focus of some `bot` communities, or engage themselves in discussions initiated by different types of users.

Moreover, the number of communities of each class remains relatively constant as the number of layers increase. Furthermore, the results show that the `sqrt` and `uniform` weight schemes generate communities with the highest purity scores for all number of layers. The first scheme provides the best overall results in both datasets, even though is less effective than the `uniform` configuration in the 10-layer setting. These weight schemes also partition the nodes into a smaller number of communities, compared to the `square` and `linear` counterparts. In the latter (`square` and `linear`), the contribution of higher layers vanishes faster than the former (`square` and `uniform`), preventing the Louvain algorithm from effectively using the information and community structure of these higher layers.

We resume the analysis by evaluating the impact of the number of layers in the final results, considering the purity score of communities as the quality measure.

The Figures 4.25, 4.26, 4.27 and 4.28 show the results for different number of layers, ranging from 1 to 20 layers. Similar to the previous experiment, the y-axis encodes the median purity score across the communities, separated by the majority class. The x-axis is the number of layers in each network. Also, the size of each point (as well as the annotation near the point) represents the number of communities found by the Louvain algorithm (left plot), and the size of each point (as well as the annotation) represents the median size of the communities (right plot).

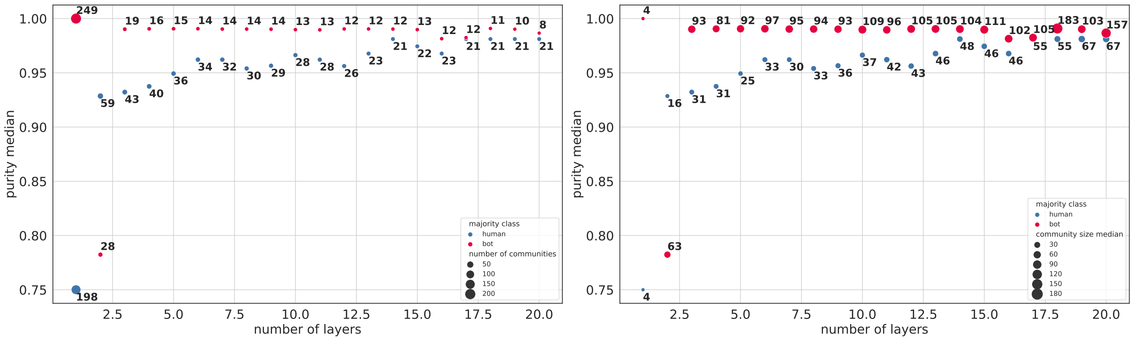

Figure 4.25: Median purity scores for different number of layers in the `cresci-stock` dataset Using `sqrt` weight scheme. Left plot provides the number of communities, right plot provides the median size of the communities.

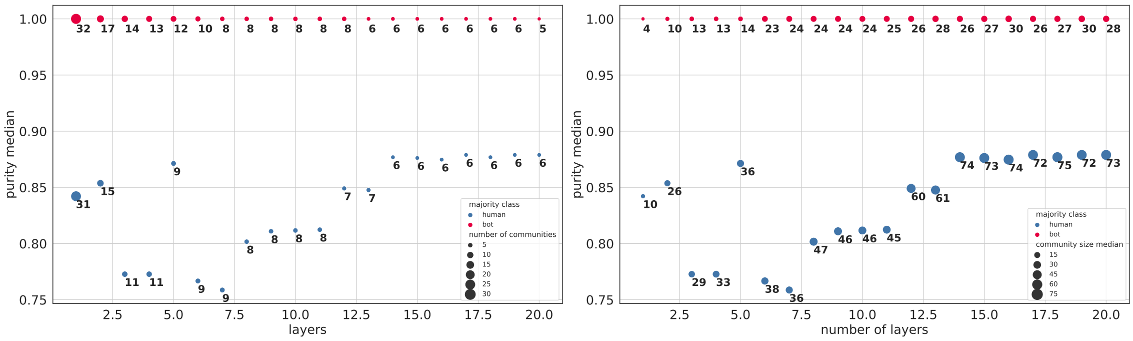

Figure 4.26: Median purity scores for different number of layers in the `rtbust` dataset Using `sqrt` weight scheme. Left plot provides the number of communities, right plot provides the median size of the communities.

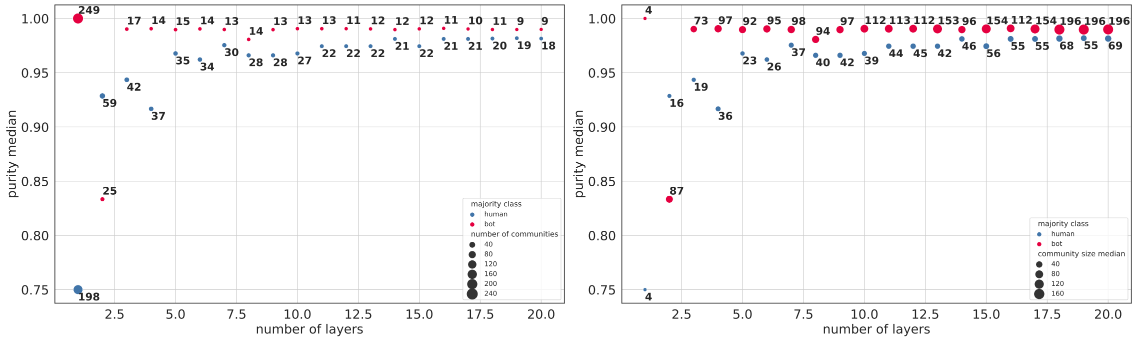

Figure 4.27: Median purity scores for different number of layers in the `cresci-stock` dataset Using `uniform` weight scheme. Left plot provides the number of communities, right plot provides the median size of the communities.

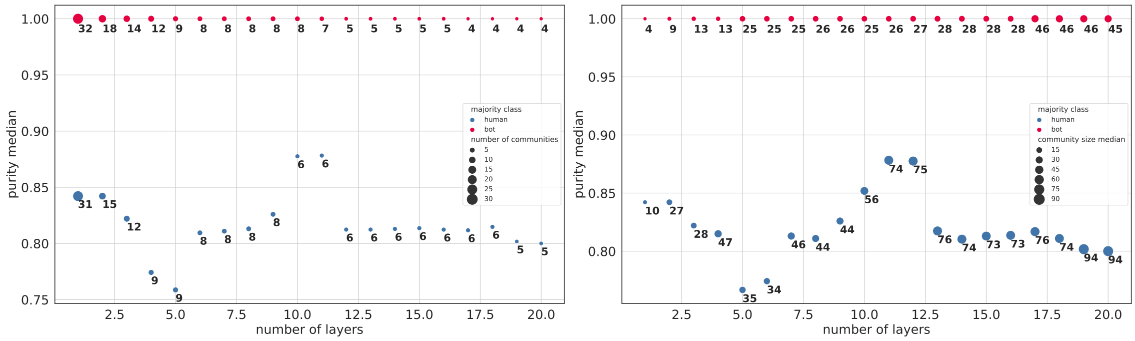

Figure 4.28: Median purity scores for different number of layers in the `rtbust` dataset Using `uniform` weight scheme. Left plot provides the number of communities, right plot provides the median size of the communities.

Note that, in general, the median purity score increases as we introduce more layers. Also, note that the purity score for bot communities is always higher, for both datasets and almost all layers, reaching the maximum value (1) in the `rtbust` dataset for all layers and both weight schemes, which also indicates the existence of targeted and coordinated actions. Moreover, the number of communities decreases while the median size of communities increases as more layers are considered. In the `cresci-stock` dataset, this increase in the number of layers also generated communities with higher purity score. In the meantime, after the 12-layer setting, the median purity score in the `rtbust` dataset starts decreasing.

As mentioned before, networks from higher layers tend to present a clearer community structure, making potentially coordinated groups more evident, which may reflect in a higher purity score for both classes of orchestration. Note that the Louvain algorithm tends to output a smaller number of communities as the number of layer increases, since it will receive denser networks.

The quality of results exhibits a stable trend after 15 layers, showing negligible variations for higher values. Based on these observations, the suggestion is to apply the framework using the multilayer Louvain algorithm with 15 layers and `sqrt` weight scheme, for obtaining a good compromise within the results.

Considering this parametrization, the number of communities in each dataset divided by the majority class as well as the descriptive statistics for the purity score are described in Tables 4.8 and 4.9. Note that the purity for the bot communities is very larger, being perfect (value 1) for all communities in `rtbust`. Interestingly, `rtbust` exhibits the same number of communities for bots and humans, while in `cresci-stock` human communities is almost twice as many.

Table 4.8: Descriptive statistics of purity score for communities in the `cresci-stock` dataset

	number of communities	**min**	**max**	**mean**	**median**
humans	22	0.53	1	0.89	0.97
bots	13	0.62	1	0.89	0.99

Table 4.9: Descriptive statistics of purity score for communities in the `rtbust` dataset

	number of communities	**min**	**max**	**mean**	**median**
humans	6	0.57	0.96	0.83	0.88
bots	6	1	1	1	1

4.4 Communities description from user's language models

We applied the procedure defined in the framework to generate a description for the topic that is posted within some of the communities. In `cresci-stock`, two communities composed by bots had the following topic:

- `$AAPL $FB $SPY @NasdaqReporter $SHIP`
- `$CHRO $OWCP $BVTK $DOLV $AMLH`

In the first, we observe cashtags from big tech companies, like Apple and Inc. and Facebook, Inc. , whose stocks values surpasses hundreds of dollars, alongside the `SHIP` cashtag, whose stocks worth considerably less, indicating that the corresponding bot community was performing the *cashtag piggybacking* technique to artificially increase the popularity of this specific company.

In the second, we have the `CHRO` cashtag, whose stock reach a peak value of 44.98 dollars in Feb 2020, alongside cashtags from companies that are worth less than one cent of dollar per stock, indicating another case of a coordinated group that artificially promotes low-value stocks.

In the `rtbust` dataset, two of the bot communities have the following topic, according to the methodology of the proposed framework:

- `@Valerio_Scanu grande #capovolgoilmondo nuovo singolo`
- `@peugeotitalia @citroenitalia @motorionline #Peugeot Nuovo`

The first community is a coordinated group promoting the latest album release of an Italian Pop singer. We also detect other bot communities with similar description, indicating a distributed effort of several bot groups in promoting the artist's album. The second group corresponds to bots focused on cars, mentioning European car manufacturers. Interestingly, the communities described here were also reported in the original `rtbust` work [2].

Using the communities detected by the Louvain algorithm, we proceed to the next step of the framework, that aims to automatically differentiate between organic orchestrations, which naturally emerges from users engagement in supporting a subject or cause, and artificial coordination, where automated user accounts are mobilized to share and increase the reach and/or popularity of a topic.

Chapter 5

Empirical Analysis of the Supervised Component

This Chapter covers results regarding the supervised component of the proposed framework, responsible for classifying communities according their features of organic or automated coordination. We start presenting preliminary investigations and exploratory analysis of structural and linguistic characteristics of communities mainly composed by bots and humans, highlighting their similarities and differences. We proceed by training a classifier using features of the communities returned by the framework as the dataset, taking the majority class inside each community as its label. Using the trained model, we apply the framework to recent events in Brazil, using data collected from the Twitter API in the appropriate period. Finally, we compare the predictions of the framework at user-level, with the predictions of a bot detection system used as reference in the literature.

5.1 Preliminary investigations upon community language models

The first experiment aims to assess whether there are linguistic differences between communities labeled as bots and humans. In order to exploit potential distinctions, we compute a language model for each community, considering the union of all tweets belonging to all users in the group as the community corpus, and plotting the LM in a way that resemble the Zipf's Law [70]: the words are sorted by their frequency (in our case, their relative frequency) in descending order, and plotted according to their ranks, as shown in Figure 5.1. The x-axis is the rank of a term and y-axis is its relative frequency, in logarithmic scale. Each curve represents a distinct community language model, where a hue towards red represent communities labeled as bots, and a hue towards blue represent communities labeled as humans, for both datasets.

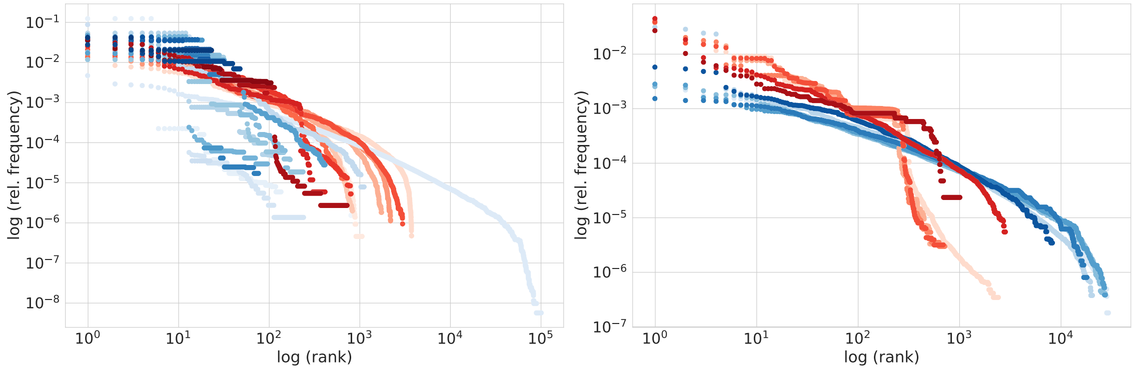

Figure 5.1: Zipf's Law for community language models in `cresci-stock` (left) and `rtbust` (right) datasets

The results show that there is no apparent differences among LM's from bots and human communities in the `cresci-stock` dataset. We hypothesize that, due to the nature of the dataset, focused on financial market and cashtags, both organic and automated groups share a common vocabulary, posting content about companies and using similar financial jargons with relative similar frequencies. As mentioned in the section where the datasets were introduced, the coordinated action of bots accounts promoting specific low-value stocks could initiate organic discussions about these stocks by authentic users, as their popularity increase. Also, as the *cashtag piggybacking* technique leverages the popularity of high-value companies to promote low-value ones, it is reasonable to expect that both types of users share content at similar frequencies and participates in discussions concerning the same organizations.

On the other hand, we can observe a clearer distinction among the language models of bots and human communities in the `rtbust` dataset. In particular, the human communities exhibit a heavier tail in their distribution, suggesting that the content shared by human communities is more diverse and decentralized, as different groups engages themselves with discussions about different topics, leading to a broader vocabulary and a broader range of relative frequencies. In turn, bots communities tend to be mobilized towards the promotion of a specific subject, which results in a more restricted set of terms and shorter tail in the distribution, as they are focused on delivering and popularizing the same content.

5.2 Analysis of features for the community classifier

We resume our investigation of features that could be used to distinguish between bots and humans communities. Recall the metrics analyzed in Chapter 4: edge reciprocity, global clustering coefficient and the descriptive statistics of the in-degree distribution of nodes in each kind of community. Moreover, we consider the community size and the density of edges in the induced subgraph. Finally, we compute some features derived from the community language model, namely: the mean and the standard deviation of the rank distribution, and the vocabulary size. Table 5.1 summarizes the chosen features.

Table 5.1: Features for classification

Community Language Model	**Community Structural Characteristics**
mean rank	community size
standard deviation	edge density
vocabulary size	edge reciprocity
	global clustering coefficient
	in-degree max
	in-degree median

We calculate the metrics upon communities resulting from the multilayer Louvain algorithm, using 15 layers and the `sqrt` weight scheme ($\frac{1}{\sqrt{k}}$ discussed in the previous chapter). As in the previous experiments, we split the communities by their majority class. Figures 5.2 and 5.3 show the distribution of each feature across the communities in the `cresci-stock` and `rtbust` datasets respectively, where the values have been standardized (subtracting the feature mean and then dividing the difference by the feature standard deviation) for better visualization. The red boxplots indicate the distribution for communities for `bot` communities, while the blue ones correspond to communities labeled as `humans`.

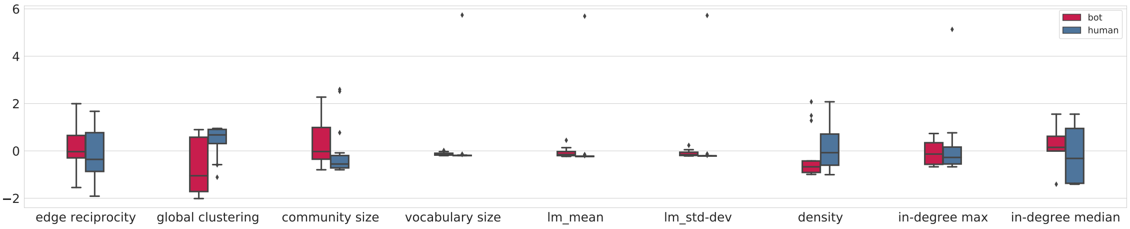

Figure 5.2: Boxplot of features, splitted by the majority class in communities - `cresci-stock` dataset

Considering the features extracted from the `cresci-stock` dataset, we observe that the median edge reciprocity for `bot` communities is higher than for `humans`, indicating a stronger signal of coordination between pairs of users in this scenario. In the meantime, we observe a considerably lower GCC median from `bot` communities, which may indicate that the automated accounts are not acting in a pattern that forms triangles in the network. The median community size for `bot` communities is also higher than the `human` communities, which suggests that the automated users have been organized in larger coordinated groups. In accordance with results observed in the previous section, there is no clear distinction between the features extracted from the language models of `bots` and `humans` communities.

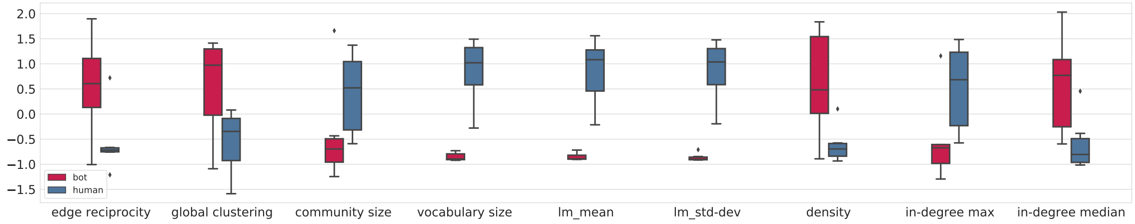

Figure 5.3: Boxplot of features, splitted by the majority class in communities - `rtbust` dataset

In contrast, the `rtbust` dataset exhibits clearer distinctions in the distributions of `humans` and `bots` communities across all features. In particular, the `bot` communities show stronger signals of coordination, considering both ER and GCC, with considerably higher values for Q1, Q2 and Q3 quartiles in the distributions, compared to the `human` counterparts. Furthermore, the `bots` communities present a more concentrated distribution of their language model features, compared to the `human` groups' distribution, corroborating with the hypothesis that automated users were acting in a coordinated and targeted pattern, increasing the popularity of specific subjects, leading to less diversity in the vocabulary, while the `human` communities participate in a broader number of topics and use a more diverse vocabulary. The edge density of bots groups also shows considerably higher values for all quartiles, which indicates a clearer community structure and stronger traits of coordination in these communities.

5.3 Classifier training and evaluation

In order to determine whether a coordinated group exhibits traits of organic or automated actions, we built a classifier using the communities obtained by the framework in the `cresci-stock` and `rtbust` datasets, extracting the features presented in the previous section for each of them, and labeling the communities according to their majority class. This choice of labels for communities is motivated by the high values of the mean and median of the purity scores observed in the previous chapter, indicating a high degree of homogeneity inside communities detected by the framework. The compiled dataset in summarized in Table 5.2:

Table 5.2: Dataset of communities used in the classifier

	cresci-stock	rtbust	**Total**
bot	13	6	19
human	22	6	28

We proceed by training a Random Forest model for supervised classification using nested cross-validation [71], considering both datasets jointly, fitting a single model for both. The Random Forest classifier has been previously used in several bot detection datasets, exhibiting a great performance in all of them [58], [72], [73], [59].

The method of nested cross-validation is exemplified in Figure 5.4, where five folds are considered in the external k-fold (outer loop) and two in the internal k-fold (inner loop).

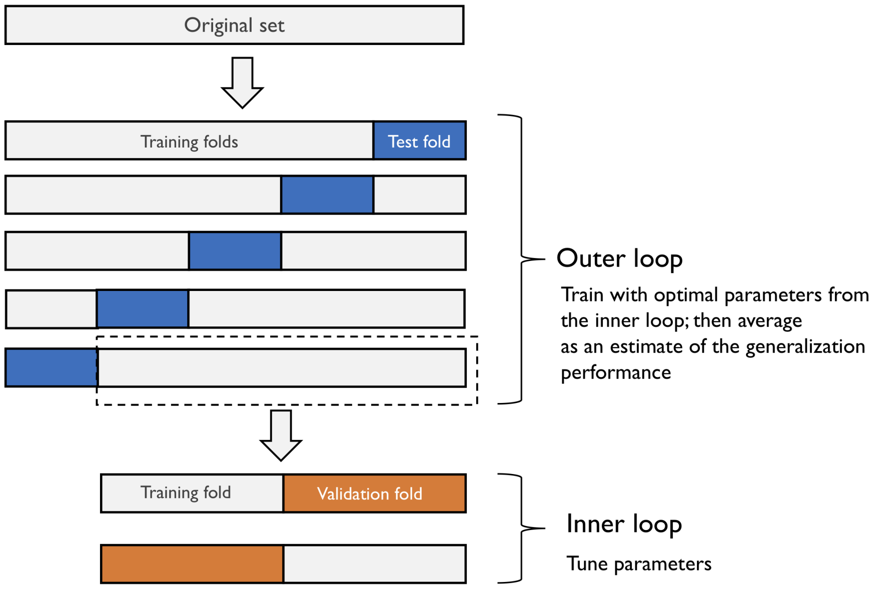
Original set
Training folds
Test fold
Outer loop
Train with optimal parameters from the inner loop; then average as an estimate of the generalization performance
Training fold
Validation fold
Inner loop
Tune parameters

Figure 5.4: Nested cross-validation scheme. Extracted from [3].

In each iteration of the outer loop, one fold is taken as the test set, and the remaining ones will comprise the training folds. Then, an additional k-fold is performed upon the training folds in the inner loop, where one of the folds is taken as the validation set, and the remaining ones will compose the training set.

The internal k-fold (inner loop) comprises the model selection and hyperparameter tuning stages, where the models are trained with several configurations, and their generalization capabilities are estimated using the validation set. Thereon, a new model is trained considering all training folds and using the hyperparameter setting with best performance in the validation set. Finally, the generalization performance of this trained model is evaluated using the test set, and an iteration of the outer loop is finished. Averaging the results obtained in the external k-fold (outer loop), one gets an almost unbiased estimate of the generalization performance [74].

The experiments have used $k = 5$ folds for both internal and external k-folds, implemented using the `scikit-learn` python library [75]. In both stages, the folds have been made in a stratified fashion, preserving the proportion of samples for each kind of community.

In the internal k-fold, the following hyperparameters have been tuned (the range of values evaluated for each of them is described below):

- **Forest hyperparameter**
  - overall number of trees;
    - [10, 20, 30, 40, 50, 100]
- **Tree-specific hyperparameters**
  - maximum depth of the tree;
    - [3, 5, 7]
  - maximum number of features when choosing the best split;
    - [3, 5, 7]
  - minimum number of samples to split an internal node;
    - [2, 4, 8, 10]
  - minimum number of samples for a node to be considered as a leaf node;
    - [1, 2, 4]
  - whether use bootstrap (drawn training samples with replacement);
    - [`True`, `False`]
  - percentage of samples to draw from the training set (only applicable if `bootstrap` is used);
    - [0.5, 0.8, 0.95]
  - weights associated with classes in the training set;
    - [`None`, `balanced`, `balanced_subsample`]

Furthermore, in the internal k-fold, a randomized search procedure has been used, where each setting is sampled from the possible combinations of hyperparameters [76]. In each iteration, 2000 configurations have been selected. The candidate models were evaluated using the F1-score metric, which provides the best trade-off between precision and recall.

The descriptive statistics of the performance in the external k-fold for several different performance metrics are reported in Table 5.3. The results indicate a good generalization performance, in particular since the two datasets are very different and a single model was trained to classify the communities. In subsequent analysis, the model with the highest F1-score in the test set across the iterations of the external k-fold have been used.

Table 5.3: Model performance in external k-fold

	mean	**std. dev.**	**min**	**max**	**median**
F1-score	**0.83**	0.1	0.75	1	0.8
Precision	**0.8**	0.19	0.6	1	0.75
Recall	**0.9**	0.14	0.75	1	1
Accuracy	**0.85**	0.1	0.78	1	0.8

Using the impurity-based criterion for feature importance in the Random Forest model, the 5 most important features for the best model are the following:

- mean rank;
- standard deviation;
- vocabulary size;
- community size;
- in-degree median;

The set of most important features are composed by features derived from both language model and structure of the communities. Interestingly, the linguistic features are the most helpful features to distinguish the different kinds of coordination, which corroborates with the hypothesis of the existence of significant differences in the way `human` and `bot` communities post content in online social networks.

5.4 Application in novel contexts

We proceed by applying the proposed framework in different scenarios, in order to investigate the presence of coordinated and automated groups of users in other contexts.

We consider three recent events in Brazil, namely:

(i) The beginning of vaccination campaign against COVID-19. The analyzed period was from 17 to 19 Jan, 2021 (inclusive);

(ii) The public pronouncement of the current Brazilian President when he canceled an early agreement for COVID vaccines with a Chinese laboratory in late 2020. The analyzed period was the day of 21 Out, 2020;

(iii) A public campaign against the most popular TV station in Brazil, the Rede Globo, in early 2021. The analyzed period was from 02 to 04 Jan, 2021 (inclusive);

The data was collected by the TWIST Systems [1] company, using the Standard Tier version of the Twitter API. Considering the collected tweets that fall inside each timestamp and are related to each of the topics (via hashtags and other keywords), we performed a subsample of 5000 profiles, biased by their number of posts in the respective period. Thus, users with a higher number of posts had a higher probability of being chosen to be part of the dataset to be analyzed. The selected users for each context were submitted to the framework, and the results are described in the following subsections.

5.4.1 Beginning of vaccination campaign against COVID-19 in Brazil

The framework returned a partition of the 5000 selected users into 34 different communities, where 8 of them were classified as `human` communities, and the remaining 26 ones were considered as `bot`. These 5000 users were responsible for 39981 of the collected tweets.

After inspection of tweets in some communities, we observe that, among the organic coordinated groups, the main subject that circulated inside each of them corresponds to support to the beginning of vaccination campaign; the news coverage from TV Channels about the regulatory approval of CoronaVac (the first vaccine approved by Anvisa, the Regulatory Agency of Food and Drugs in Brazil), and the application of the first shot of vaccine by the state of São Paulo.

Among the automated coordinated communities, we detected a group discussing the anticipation of the beginning of vaccination by the state of São Paulo before other states have received doses, as well as a supposed obligation of mandatory vaccination by the Supreme Federal Court of Brazil. It's worth noticing that most tweets of users inside this group are retweets of messages from a Brazilian Congressman aligned with the government, whose Twitter account is currently disabled. We also identified a community labeled as `bot` criticizing the lack of public pronouncements by the Brazilian president regarding the start of vaccination in Brazil.

[1] https://www.twist.systems/en/

5.4.2 Cancellation of an early agreement for COVID vaccines

The selected users were partitioned in 33 communities, where 14 of them were classified as `human` groups and the remaining (19) as `bots`. The users inside this sample authored 28620 posts.

Among the organic communities (labeled as humans), some disseminate a post from a Brazilian journalist about the large financial support by the Federal Government on the medicine called Hydroxychloroquine and the vaccine produced by the University of Oxford, both without proven efficiency at the time. It's worth noticing that Oxford vaccine had its effectiveness attested afterwards. We also identified communities inquiring about the CoronaVac vaccine and the lack of efficiency results (at the time). Groups criticizing the government campaign about the medicine and the anti-vaccine movement in Brazil were also observed.

Considering the communities labeled as bots, we observed users positing themselves in favor of Hydroxychloroquine, arguing that the medicine has been tested for more than 50 years and therefore could be widely used against COVID. This group also supported the subject by retweeting content from a large Twitter influencer and government supporter. We also identified communities discussing and reverberating the declarations of the Brazilian president about the cancellation of the agreement for CoronaVac vaccines.

5.4.3 Public campaign against Rede Globo

The campaign was focused on promoting the hashtag `#GloboLixo` in Twitter in Jan 3, 2021. The sampled users were partitioned in 31 communities, 14 `human` and 17 `bot`, and were responsible for 30336 posts during the analyzed period.

We identified `human` communities supporting the rejection of Big Brother Brazil, a reality show exhibited by Rede Globo, and the repulsion directed towards people who post about the program. Some communities were pointing out a supposed agglomeration induced by a journalist and host of a news program in the TV Channel, on an occasion when the journalist was not wearing a face mask. Later it was clarified that the situation happened in 2019. Other organic groups were retweeting posts from a Twitter influencer showing her support to the campaign.

Among the bot communities, we observe users supporting the campaign and criticizing the TV Channel for positioning itself against Rede Globo the actions of the Federal Government during the pandemic.

5.5 Comparison of individual accounts with botometer predictions

The communities classified as bot by the framework are expected to be mainly composed of accounts that are actually bots. In order to verify this hypothesis, we conduct an experiment to evaluate the agreement between the communities generated by the framework and the predictions of Botometer v4 [12], a state-of-the-art tool for identifying bot accounts in Twitter and used as reference in the literature [59], [77], [78]. Although our work proposes a method for identifying bot groups, and not bot accounts individually, we compare it with a method for classifying individual accounts.

For each of the 3 datasets analyzed in the previous section, we selected a community classified as bot and another classified as human. We submit the users that compose each community to the Botometer v4 API [2], which returns the probability that the account is a bot. We have collected the Botometer predictions for 1925 users altogether. Figures 5.5, 5.6 and 5.7 show the ECCDF of Botometer scores (probability that a profile is a bot) for each dataset, where the red curve represents the distribution for accounts belonging to the bot community, while the blue curve corresponds to the human community, according to the framework.

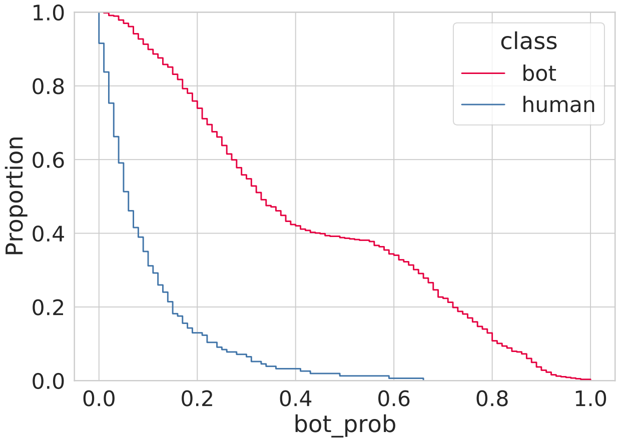

Figure 5.5: ECCDF of Botometer scores for a bot and human community - "Beginning of vaccination campaign against COVID-19 in Brazil"

[2] https://rapidapi.com/OSoMe/api/botometer-pro/

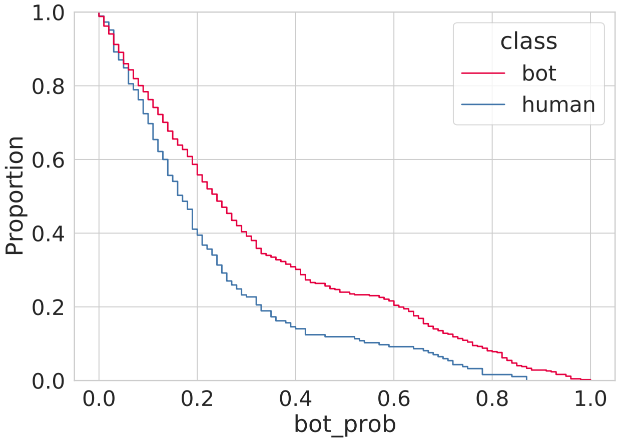

Figure 5.6: ECCDF of Botometer scores for a `bot` and `human` community - "Cancelation of an early agreement for COVID vaccines"

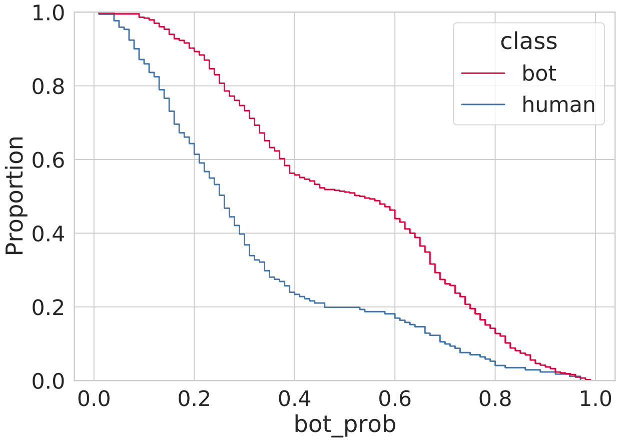

Figure 5.7: ECCDF of Botometer scores for a `bot` and `human` community - "Public campaign against Rede Globo"

We observe a clear distinction between the bot score distributions of each class across the datasets. This distinction is clearer in the first event (Beginning of vaccination campaign against COVID-19 in Brazil), where less than 20% of the users in the `human` community exhibits a bot score higher than 0.2, while almost 40% of users belonging to the `bot` group have a bot score higher than 0.4. Moreover, note that no user in the `human` community has a bot score higher than 0.65 in the tail of the distribution, while almost 30% of the users in the `bot` community have a score higher than 0.65.

In the third scenario (Public campaign against Rede Globo), we observe that less than 30% of users from the `human` community received a bot score higher than 0.4, while almost 60% of the users in the `bot` community scored higher than 0.4. Furthermore, only 20% of the profiles in the `human` community had a score higher than 0.6, while this value is almost 50% for users in the `bot` community. Interestingly, there is no clear difference in the tail of the distribution for the `bot` and `human` communities, as both communities have users exhibiting very high bot scores.

In the meantime, we observe a relative small difference between the distributions in the second event (Cancellation of an early agreement for COVID vaccines), though the users belonging to the `bot` community did received higher bot scores.

The results suggest that, in general, the communities classified as `bot` or `human` are in fact composed by users that share exhibit features that identify them as bots or humans, taking the predictions generated by Botometer as reference. Thus, the framework can effectively and automatically reveal communities that are likely comprised of bots or humans.

Chapter 6

Conclusion and Future Work

The growing popularity and adoption of online social networks led to several challenges regarding the dissemination of low-credibility and fraudulent content. Different segments of the society have been using online social networks to manipulate opinion and support their interests, using different strategies to achieve their goals, like automated accounts, known as bots, that can coordinate their actions (posts) to promote the goals of their operators.

This work proposed a framework to identify groups of automated and authentic users that post similar content in online social networks, possibly indicating coordinated action among them, by leveraging the linguistic similarities among users' language models to build networks and uncover these communities. To the best of our knowledge, this is the first framework to identify coordinated content posted by bots, since all existing methodologies and frameworks in the literature focus on the individual bot identification, putting aside the aspects of synchronization reported in the recent literature [24], [2].

The proposed framework is organized in two major components: an unsupervised and supervised component. In the former, using the language model representations generated for each user, the framework builds several networks that encodes different levels of similarities among users, capturing both signals of coordination between users and the community structure generated by large groups of bots. In the latter component, using linguistic traits and structural characteristics of a community, the framework introduces a method to assess whether users within this community exhibits traits of organic (the `human` class), where real users engage themselves towards a common cause, or automated coordination (the `bot` class), where bots are mobilized to artificially popularize a topic or subject of interest.

The analyses upon the unsupervised component clearly indicated the structural differences between the different types of communities, reporting considerably stronger signals of coordination between pairs and triplets of automated users when compared to communities of users labeled as humans in both analyzed datasets (rtbust and cresci-stock). Observed results have indicated that the edge reciprocity and the global clustering coefficient metrics in the induced subgraphs of bots are almost twice as high as the human counterparts. We have also investigated the impact of several weight schemes and different number of layers in the quality of the communities identified by the variation of the Louvain algorithm for multilayer community detection, finding the most suited parameter configuration for obtaining high-quality communities.

Considering the supervised component, the differences between language models of communities of bots and humans users in both datasets show that automated users tend to be focused on promoting specific topics, yielding a distribution over the vocabulary with a lighter tail. We have also introduced a methodology to classify the nature of coordination of a group, using the resulting communities returned by the unsupervised component as the dataset for training and evaluating the classifier. Observed results have indicated a good generalization performance, with an accuracy of 0.85 and F1-score of 0.83 when considering a single classifier for the combined dataset.

Last, the proposed framework was applied to different scenarios, investigating the presence of coordinated groups in each of them as well as the focus of these groups (in terms of topics). We have observed groups of bots supporting Brazilian congresspersons, a medicine without proven efficiency against COVID-19 and a campaign against a brazilian TV Station. We have also evaluated the agreement between the predictions of the kind of communities identified by the framework at individual user level and a bot detection system used as reference in the literature, showing that, in general, users in communities labeled as bots are likely to have a higher bot score, in comparison to users in communities labeled as humans. For example, in the "Beginning of vaccination campaign against COVID-19 in Brazil" dataset, no user in the human community have exhibited a bot score higher than 0.65, while almost 30% of the users in the bot community have had a score higher than 0.65.

6.1 Future Work

The following is a list of possible extensions and promising directions to investigate in order to enhance the proposed framework:

- Explore language models of higher orders: this work uses the unigram language model to represent users. Despite its simplicity, the model has proved itself useful to encode textual similarities among profiles, facilitating the identification of coordinated groups. However, higher order LMs could capture more complex similarities and relationships, highlighting more elaborate coordination among bot accounts, like an effort of bots towards promoting semantically similar contents.

- Define a similarity threshold to connect two profiles: in our framework, every user will be connected to its K nearest neighbors, independently of the actual similarity value between them. We hypothesize that, by defining a minimum similarity threshold among users LMs, we would be able to make the networks more sparse, isolating accounts with a more unique discourse, while preserving the connections among users exhibiting highly similar LMs, which would highlight the community structure and facilitate their identification.

- Use a suitable data structure for fast nearest-neighbor searches: In the current framework, the distance between the language models of every pair of users is computed, which scales as $\mathcal{O}(n^2)$, where n is the number of users. This step is needed in order to identify the K most similar user of every user. This computational complexity could be prohibitive for very large networks. One possible way reduce this complexity is to perform a preprocessing step, using a data structure like a Ball Tree [79] or a VP-Tree [80] to organize the LMs in the $\mathcal{V}$-dimensional space ($\mathcal{V}$ corresponds to the vocabulary size) in order to enable fast nearest-neighbors searches, scaling as $\mathcal{O}(n \log n)$.

- Evaluate the framework in different online social networks: the evaluation performed in this work considered only Twitter but the framework can be applied to any online social network (OSN) where users generate content. It would be interesting to see results for other OSNs, such as Facebook or Instagram.

References

[1] FORTUNATO, S., HRIC, D. “Community detection in networks: A user guide”, , v. 659, pp. 1–44, nov. 2016. doi: 10.1016/j.physrep.2016.09.002.

[2] MAZZA, M., CRESCI, S., AVVENUTI, M., et al. “RTbust: Exploiting Temporal Patterns for Botnet Detection on Twitter”. In: *Proceedings of the 10th ACM Conference on Web Science*, WebSci ’19, p. 183–192, New York, NY, USA, 2019. Association for Computing Machinery. ISBN: 9781450362023. doi: 10.1145/3292522.3326015. Disponível em: <https://doi.org/10.1145/3292522.3326015>.

[3] RASCHKA, S. “Model evaluation, model selection, and algorithm selection in machine learning Part IV - Comparing the performance of machine learning models and algorithms using statistical tests and nested cross-validation”. https://sebastianraschka.com/blog/2018/model-evaluation-selection-part4.html. Accessed: 2021-06-16.

[4] “Digital 2021 - We Are Social”. https://wearesocial.com/digital-2021. Accessed: 2021-06-01.

[5] BUDAK, C., AGRAWAL, D., EL ABBADI, A. “Limiting the Spread of Misinformation in Social Networks”. In: *Proceedings of the 20th International Conference on World Wide Web*, WWW ’11, p. 665–674, New York, NY, USA, 2011. Association for Computing Machinery. ISBN: 9781450306324. doi: 10.1145/1963405.1963499. Disponível em: <https://doi.org/10.1145/1963405.1963499>.

[6] WU, L., MORSTATTER, F., CARLEY, K. M., et al. “Misinformation in Social Media: Definition, Manipulation, and Detection”, *SIGKDD Explor. Newsl.*, v. 21, n. 2, pp. 80–90, nov. 2019. ISSN: 1931-0145. doi: 10.1145/3373464.3373475. Disponível em: <https://doi.org/10.1145/3373464.3373475>.

[7] SAMANTHA BRADSHAW, H. B. . P. N. H. *Industrialized Disinformation: 2020 Global Inventory of Organised Social Media Manipulation.* Re-

latório técnico, Oxford, UK: Project on Computational Propaganda, jan 2021. Disponível em: <https://demtech.oii.ox.ac.uk/research/posts/industrialized-disinformation/>.

[8] SHAO, C., CIAMPAGLIA, G. L., VAROL, O., et al. "The spread of low-credibility content by social bots", *Nature Communications*, v. 9, n. 1, pp. 4787, Nov 2018. ISSN: 2041-1723. doi: 10.1038/s41467-018-06930-7. Disponível em: <https://doi.org/10.1038/s41467-018-06930-7>.

[9] DAVIS, C. A., VAROL, O., FERRARA, E., et al. "BotOrNot: A System to Evaluate Social Bots". In: *Proceedings of the 25th International Conference Companion on World Wide Web*, WWW '16 Companion, p. 273–274, Republic and Canton of Geneva, CHE, 2016. International World Wide Web Conferences Steering Committee. ISBN: 9781450341448. doi: 10.1145/2872518.2889302. Disponível em: <https://doi.org/10.1145/2872518.2889302>.

[10] MILLER, Z., DICKINSON, B., DEITRICK, W., et al. "Twitter spammer detection using data stream clustering", *Information Sciences*, v. 260, pp. 64–73, 2014. ISSN: 0020-0255. doi: https://doi.org/10.1016/j.ins.2013.11.016. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0020025513008037>.

[11] KUDUGUNTA, S., FERRARA, E. "Deep Neural Networks for Bot Detection", *ArXiv*, v. abs/1802.04289, 2018.

[12] SAYYADIHARIKANDEH, M., VAROL, O., YANG, K.-C., et al. "Detection of Novel Social Bots by Ensembles of Specialized Classifiers". In: *Proceedings of the 29th ACM International Conference on Information amp; Knowledge Management*, CIKM '20, p. 2725–2732, New York, NY, USA, 2020. Association for Computing Machinery. ISBN: 9781450368599. doi: 10.1145/3340531.3412698. Disponível em: <https://doi.org/10.1145/3340531.3412698>.

[13] JIANG, M., CUI, P., BEUTEL, A., et al. "Catching Synchronized Behaviors in Large Networks: A Graph Mining Approach", *ACM Trans. Knowl. Discov. Data*, v. 10, n. 4, jun. 2016. ISSN: 1556-4681. doi: 10.1145/2746403. Disponível em: <https://doi.org/10.1145/2746403>.

[14] JIANG, M., CUI, P., BEUTEL, A., et al. "Inferring Lockstep Behavior from Connectivity Pattern in Large Graphs", *Knowl. Inf. Syst.*, v. 48, n. 2, pp. 399–428, ago. 2016. ISSN: 0219-1377. doi: 10.

1007/s10115-015-0883-y. Disponível em: <https://doi.org/10.1007/s10115-015-0883-y>.

[15] BESKOW, D. M., CARLEY, K. M. "Bot Conversations Are Different: Leveraging Network Metrics for Bot Detection in Twitter". In: *Proceedings of the 2018 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, ASONAM '18, p. 825–832. IEEE Press, 2018. ISBN: 9781538660515.

[16] VARGAS, L., EMAMI, P., TRAYNOR, P. "On the Detection of Disinformation Campaign Activity with Network Analysis". In: *Proceedings of the 2020 ACM SIGSAC Conference on Cloud Computing Security Workshop*, CCSW'20, p. 133–146, New York, NY, USA, 2020. Association for Computing Machinery. ISBN: 9781450380843. doi: 10.1145/3411495.3421363. Disponível em: <https://doi.org/10.1145/3411495.3421363>.

[17] CRESCI, S., PIETRO, R. D., PETROCCHI, M., et al. "DNA-Inspired Online Behavioral Modeling and Its Application to Spambot Detection", *IEEE Intelligent Systems*, v. 31, n. 05, pp. 58–64, sep 2016. ISSN: 1941-1294. doi: 10.1109/MIS.2016.29.

[18] ANWAR, A., YAQUB, U. "Bot Detection in Twitter Landscape Using Unsupervised Learning". In: *The 21st Annual International Conference on Digital Government Research*, dg.o '20, p. 329–330, New York, NY, USA, 2020. Association for Computing Machinery. ISBN: 9781450387910. doi: 10.1145/3396956.3401801. Disponível em: <https://doi.org/10.1145/3396956.3401801>.

[19] VO, N., LEE, K., CAO, C., et al. "Revealing and Detecting Malicious Retweeter Groups". In: *Proceedings of the 2017 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining 2017*, ASONAM '17, p. 363–368, New York, NY, USA, 2017. Association for Computing Machinery. ISBN: 9781450349932. doi: 10.1145/3110025.3110068. Disponível em: <https://doi.org/10.1145/3110025.3110068>.

[20] CRESCI, S., DI PIETRO, R., PETROCCHI, M., et al. "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*, WWW '17 Companion, p. 963–972, Republic and Canton of Geneva, CHE, 2017. International World Wide Web Conferences Steer-

ing Committee. ISBN: 9781450349147. doi: 10.1145/3041021.3055135. Disponível em: <`https://doi.org/10.1145/3041021.3055135`>.

[21] ECHEVERRÏ£¡A, J., DE CRISTOFARO, E., KOURTELLIS, N., et al. "LOBO: Evaluation of Generalization Deficiencies in Twitter Bot Classifiers". In: *Proceedings of the 34th Annual Computer Security Applications Conference*, ACSAC '18, p. 137–146, New York, NY, USA, 2018. Association for Computing Machinery. ISBN: 9781450365697. doi: 10.1145/3274694.3274738. Disponível em: <`https://doi.org/10.1145/3274694.3274738`>.

[22] CRESCI, S., PETROCCHI, M., SPOGNARDI, A., et al. "Better Safe Than Sorry: An Adversarial Approach to Improve Social Bot Detection". In: Boldi, P., Welles, B. F., Kinder-Kurlanda, K., et al. (Eds.), *Proceedings of the 11th ACM Conference on Web Science, WebSci 2019, Boston, MA, USA, June 30 - July 03, 2019*, pp. 47–56. ACM, 2019. doi: 10.1145/3292522.3326030. Disponível em: <`https://doi.org/10.1145/3292522.3326030`>.

[23] YANG, K.-C., VAROL, O., DAVIS, C. A., et al. "Arming the public with artificial intelligence to counter social bots", *Human Behavior and Emerging Technologies*, v. 1, n. 1, pp. 48–61, 2019. doi: https://doi.org/10.1002/hbe2.115. Disponível em: <`https://onlinelibrary.wiley.com/doi/abs/10.1002/hbe2.115`>.

[24] FERRARA, E., VAROL, O., DAVIS, C., et al. "The Rise of Social Bots", *Commun. ACM*, v. 59, n. 7, pp. 96–104, jun. 2016. ISSN: 0001-0782. doi: 10.1145/2818717. Disponível em: <`https://doi.org/10.1145/2818717`>.

[25] BENGIO, Y., DUCHARME, R., VINCENT, P., et al. "A Neural Probabilistic Language Model", *J. Mach. Learn. Res.*, v. 3, n. null, pp. 1137–1155, mar. 2003. ISSN: 1532-4435.

[26] MIKOLOV, T., KARAFIÁT, M., BURGET, L., et al. "Recurrent neural network based language model." In: Kobayashi, T., Hirose, K., Nakamura, S. (Eds.), *INTERSPEECH*, pp. 1045–1048. ISCA, 2010. Disponível em: <`http://dblp.uni-trier.de/db/conf/interspeech/interspeech2010.html#MikolovKBCK10`>.

[27] MIKOLOV, T., DEORAS, A., KOMBRINK, S., et al. "Empirical Evaluation and Combination of Advanced Language Modeling Techniques." In: *INTERSPEECH*, pp. 605–608. ISCA, 2011.

Disponível em: <http://dblp.uni-trier.de/db/conf/interspeech/interspeech2011.html#MikolovDKBC11>.

[28] BENGIO, Y., SIMARD, P., FRASCONI, P. "Learning long-term dependencies with gradient descent is difficult", *IEEE Transactions on Neural Networks*, v. 5, n. 2, pp. 157–166, 1994. doi: 10.1109/72.279181.

[29] HOCHREITER, S., SCHMIDHUBER, J. "Long short-term memory", *Neural computation*, v. 9, n. 8, pp. 1735–1780, 1997.

[30] SUNDERMEYER, M., SCHLÜTER, R., NEY, H. "LSTM Neural Networks for Language Modeling". In: *INTERSPEECH*, 2012.

[31] PETERS, M., NEUMANN, M., IYYER, M., et al. "Deep Contextualized Word Representations". In: *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pp. 2227–2237, New Orleans, Louisiana, jun. 2018. Association for Computational Linguistics. doi: 10.18653/v1/N18-1202. Disponível em: <https://www.aclweb.org/anthology/N18-1202>.

[32] HOWARD, J., RUDER, S. "Universal Language Model Fine-tuning for Text Classification". In: *ACL*. Association for Computational Linguistics, 2018. Disponível em: <http://arxiv.org/abs/1801.06146>.

[33] VASWANI, A., SHAZEER, N., PARMAR, N., et al. "Attention is All you Need". In: Guyon, I., Luxburg, U. V., Bengio, S., et al. (Eds.), *Advances in Neural Information Processing Systems*, v. 30. Curran Associates, Inc., 2017. Disponível em: <https://proceedings.neurips.cc/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf>.

[34] DEVLIN, J., CHANG, M.-W., LEE, K., et al. "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding". In: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp. 4171–4186, Minneapolis, Minnesota, jun. 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1423. Disponível em: <https://www.aclweb.org/anthology/N19-1423>.

[35] DAI, Z., YANG, Z., YANG, Y., et al. "Transformer-XL: Attentive Language Models beyond a Fixed-Length Context". In: *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp.

2978–2988, Florence, Italy, jul. 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1285. Disponível em: <https://www.aclweb.org/anthology/P19-1285>.

[36] RADFORD, A., WU, J., CHILD, R., et al. "Language Models are Unsupervised Multitask Learners", 2019.

[37] SANH, V., DEBUT, L., CHAUMOND, J., et al. "DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter", *CoRR*, v. abs/1910.01108, 2019. Disponível em: <http://arxiv.org/abs/1910.01108>.

[38] WANG, A., SINGH, A., MICHAEL, J., et al. "GLUE: A Multi-Task Benchmark and Analysis Platform for Natural Language Understanding". In: *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pp. 353–355, Brussels, Belgium, nov. 2018. Association for Computational Linguistics. doi: 10.18653/v1/W18-5446. Disponível em: <https://www.aclweb.org/anthology/W18-5446>.

[39] KITAEV, N., KAISER, L., LEVSKAYA, A. "Reformer: The Efficient Transformer". In: *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020. Disponível em: <https://openreview.net/forum?id=rkgNKkHtvB>.

[40] WANG, S., LI, B. Z., KHABSA, M., et al. "Linformer: Self-Attention with Linear Complexity", *CoRR*, v. abs/2006.04768, 2020. Disponível em: <https://arxiv.org/abs/2006.04768>.

[41] NEWMAN, M. E. "Modularity and community structure in networks", *Proc Natl Acad Sci U S A*, v. 103, n. 23, pp. 8577–8582, jun. 2006. doi: 10.1073/pnas.0601602103. Disponível em: <http://www.ncbi.nlm.nih.gov/sites/entrez?cmd=retrieve&db=pubmed&list_uids=16723398&dopt=AbstractPlus>.

[42] BARABÁSI, A.-L., PÓSFAI, M. *Network science*. Cambridge, Cambridge University Press, 2016. ISBN: 9781107076266 1107076269. Disponível em: <http://barabasi.com/networksciencebook/>.

[43] BRANDES, U., DELLING, D., GAERTLER, M., et al. "Maximizing Modularity is hard". 2006. Disponível em: <http://arxiv.org/abs/physics/0608255>. cite arxiv:physics/0608255 Comment: 10 pages, 1 figure.

[44] NEWMAN, M. "Fast algorithm for detecting community structure in networks", *Physical Review E*, v. 69, September 2003. Disponível em: <http://arxiv.org/abs/cond-mat/0309508>.

[45] BLONDEL, V. D., GUILLAUME, J.-L., LAMBIOTTE, R., et al. "Fast unfolding of communities in large networks", *Journal of Statistical Mechanics: Theory and Experiment*, v. 2008, n. 10, pp. P10008, 2008. Disponível em: <http://stacks.iop.org/1742-5468/2008/i=10/a=P10008>.

[46] BASTIAN, M., HEYMANN, S., JACOMY, M. "Gephi: An Open Source Software for Exploring and Manipulating Networks". 2009. Disponível em: <http://www.aaai.org/ocs/index.php/ICWSM/09/paper/view/154>.

[47] CSARDI, G., NEPUSZ, T. "The igraph software package for complex network research", *InterJournal*, v. Complex Systems, pp. 1695, 2006. Disponível em: <https://igraph.org>.

[48] HAGBERG, A. A., SCHULT, D. A., SWART, P. J. "Exploring Network Structure, Dynamics, and Function using NetworkX". In: Varoquaux, G., Vaught, T., Millman, J. (Eds.), *Proceedings of the 7th Python in Science Conference*, pp. 11 – 15, Pasadena, CA USA, 2008. Disponível em: <http://conference.scipy.org/proceedings/SciPy2008/paper_2/>.

[49] PONS, P., LATAPY, M. "Computing Communities in Large Networks Using Random Walks". In: Yolum, p., Güngör, T., Gürgen, F., et al. (Eds.), *Computer and Information Sciences - ISCIS 2005*, pp. 284–293, Berlin, Heidelberg, 2005. Springer Berlin Heidelberg. ISBN: 978-3-540-32085-2.

[50] DE GUZZI BAGNATO, G., RONQUI, J. R. F., TRAVIESO, G. "Community detection in networks using self-avoiding random walks", *Physica A: Statistical Mechanics and its Applications*, v. 505, n. C, pp. 1046–1055, 2018. doi: 10.1016/j.physa.2018.04.0. Disponível em: <https://ideas.repec.org/a/eee/phsmap/v505y2018icp1046-1055.html>.

[51] LANCICHINETTI, A., FORTUNATO, S. "Consensus clustering in complex networks", *Scientific Reports*, v. 2, n. 1, Mar 2012. ISSN: 2045-2322. doi: 10.1038/srep00336. Disponível em: <http://dx.doi.org/10.1038/srep00336>.

[52] TANDON, A., ALBESHRI, A., THAYANANTHAN, V., et al. "Fast consensus clustering in complex networks", *Physical Review E*, v. 99, n. 4, Apr

2019. ISSN: 2470-0053. doi: 10.1103/physreve.99.042301. Disponível em: <http://dx.doi.org/10.1103/PhysRevE.99.042301>.

[53] ROSVALL, M., BERGSTROM, C. T. "Maps of random walks on complex networks reveal community structure", *Proceedings of the National Academy of Sciences*, v. 105, n. 4, pp. 1118–1123, 2008. doi: 10.1073/pnas.0706851105. Disponível em: <http://www.pnas.org/content/105/4/1118.abstract>.

[54] HUFFMAN, D. A. "A Method for the Construction of Minimum-Redundancy Codes", *Proceedings of the Institute of Radio Engineers*, v. 40, n. 9, pp. 1098–1101, September 1952.

[55] PAGE, L., BRIN, S., MOTWANI, R., et al. "The PageRank citation ranking: Bringing order to the Web". In: *Proceedings of the 7th International World Wide Web Conference*, pp. 161–172, Brisbane, Australia, 1998. Disponível em: <citeseer.nj.nec.com/page98pagerank.html>.

[56] EDLER, D., GUEDES, T., ZIZKA, A., et al. "Infomap Bioregions: Interactive Mapping of Biogeographical Regions from Species Distributions", *Systematic Biology*, v. 66, n. 2, pp. 197–204, 10 2016. ISSN: 1063-5157. doi: 10.1093/sysbio/syw087. Disponível em: <https://doi.org/10.1093/sysbio/syw087>.

[57] ALBADI, N., KURDI, M., MISHRA, S. "Hateful People or Hateful Bots? Detection and Characterization of Bots Spreading Religious Hatred in Arabic Social Media", *Proc. ACM Hum.-Comput. Interact.*, v. 3, n. CSCW, nov. 2019. doi: 10.1145/3359163. Disponível em: <https://doi.org/10.1145/3359163>.

[58] VAROL, O., FERRARA, E., DAVIS, C. A., et al. "Online Human-Bot Interactions: Detection, Estimation, and Characterization". In: *Proceedings of the Eleventh International Conference on Web and Social Media, ICWSM 2017, Montréal, Québec, Canada, May 15-18, 2017*, pp. 280–289. AAAI Press, 2017. Disponível em: <https://aaai.org/ocs/index.php/ICWSM/ICWSM17/paper/view/15587>.

[59] YANG, K., VAROL, O., HUI, P., et al. "Scalable and Generalizable Social Bot Detection through Data Selection". In: *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pp.

1096–1103. AAAI Press, 2020. Disponível em: <`https://aaai.org/ojs/index.php/AAAI/article/view/5460`>.

[60] FEDORYSZAK, M., FREDERICK, B., RAJARAM, V., et al. "Real-Time Event Detection on Social Data Streams". In: *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery amp; Data Mining*, KDD '19, p. 2774–2782, New York, NY, USA, 2019. Association for Computing Machinery. ISBN: 9781450362016. doi: 10.1145/3292500.3330689. Disponível em: <`https://doi.org/10.1145/3292500.3330689`>.

[61] CAMPELLO, R. J. G. B., MOULAVI, D., SANDER, J. "Density-Based Clustering Based on Hierarchical Density Estimates". In: Pei, J., Tseng, V. S., Cao, L., et al. (Eds.), *Advances in Knowledge Discovery and Data Mining*, pp. 160–172, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg. ISBN: 978-3-642-37456-2.

[62] FUGLEDE, B., TOPSOE, F. "Jensen-Shannon divergence and Hilbert space embedding". In: *IEEE International Symposium on Information Theory*, pp. 31–31, 2004.

[63] KULLBACK, S., LEIBLER, R. A. "On Information and Sufficiency", *Ann. Math. Statist.*, v. 22, n. 1, pp. 79–86, 1951.

[64] MUCHA, P. J., RICHARDSON, T., MACON, K., et al. "Community Structure in Time-Dependent, Multiscale, and Multiplex Networks", *Science*, v. 328, n. 5980, pp. 876–878, 2010. ISSN: 0036-8075. doi: 10.1126/science.1184819. Disponível em: <`https://science.sciencemag.org/content/328/5980/876`>.

[65] BREIMAN, L. "Random Forests", *Machine Learning*, v. 45, n. 1, pp. 5–32, 2001. ISSN: 0885-6125. doi: 10.1023/A:1010933404324. Disponível em: <`http://dx.doi.org/10.1023/A%3A1010933404324`>.

[66] CRESCI, S., LILLO, F., REGOLI, D., et al. "Cashtag Piggybacking: Uncovering Spam and Bot Activity in Stock Microblogs on Twitter", *ACM Trans. Web*, v. 13, n. 2, abr. 2019. ISSN: 1559-1131. doi: 10.1145/3313184. Disponível em: <`https://doi.org/10.1145/3313184`>.

[67] CRESCI, S., PIETRO, R. D., PETROCCHI, M., et al. "Social Fingerprinting: Detection of Spambot Groups Through DNA-Inspired Behavioral Modeling", *IEEE Trans. Dependable Secur. Comput.*, v. 15, n. 4,

pp. 561–576, 2018. doi: 10.1109/TDSC.2017.2681672. Disponível em: <https://doi.org/10.1109/TDSC.2017.2681672>.

[68] WASSERMAN, S., FAUST, K. *Social network analysis: Methods and applications*, v. 8. Cambridge university press, 1994.

[69] NEWMAN, M. "The Structure and Function of Complex Networks", *SIAM review*, v. 45, n. 2, pp. 167–256, 2003. ISSN: 0036-1445. doi: 10.1137/S003614450342480.

[70] ZIPF, G. K. *Human Behaviour and the Principle of Least Effort.* Addison-Wesley, 1949.

[71] CAWLEY, G. C., TALBOT, N. L. C. "On Over-fitting in Model Selection and Subsequent Selection Bias in Performance Evaluation", *Journal of Machine Learning Research*, v. 11, n. 70, pp. 2079–2107, 2010. Disponível em: <http://jmlr.org/papers/v11/cawley10a.html>.

[72] GILANI, Z., KOCHMAR, E., CROWCROFT, J. "Classification of Twitter Accounts into Automated Agents and Human Users". In: *Proceedings of the 2017 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining 2017*, ASONAM '17, p. 489–496, New York, NY, USA, 2017. Association for Computing Machinery. ISBN: 9781450349932. doi: 10.1145/3110025.3110091. Disponível em: <https://doi.org/10.1145/3110025.3110091>.

[73] YANG, K.-C., HUI, P.-M., MENCZER, F. "Bot Electioneering Volume: Visualizing Social Bot Activity During Elections". In: *Companion Proceedings of The 2019 World Wide Web Conference*, WWW '19, p. 214–217, New York, NY, USA, 2019. Association for Computing Machinery. ISBN: 9781450366755. doi: 10.1145/3308560.3316499. Disponível em: <https://doi.org/10.1145/3308560.3316499>.

[74] VARMA, S., SIMON, R. "Bias in error estimation when using cross-validation for model selection", *BMC Bioinformatics*, v. 7, n. 1, pp. 91, Feb 2006. ISSN: 1471-2105. doi: 10.1186/1471-2105-7-91. Disponível em: <https://doi.org/10.1186/1471-2105-7-91>.

[75] BUITINCK, L., LOUPPE, G., BLONDEL, M., et al. "API design for machine learning software: experiences from the scikit-learn project". In: *ECML PKDD Workshop: Languages for Data Mining and Machine Learning*, pp. 108–122, 2013.

[76] BERGSTRA, J., BENGIO, Y. "Random Search for Hyper-Parameter Optimization", *J. Mach. Learn. Res.*, v. 13, n. null, pp. 281–305, fev. 2012. ISSN: 1532-4435.

[77] VOSOUGHI, S., ROY, D., ARAL, S. "The spread of true and false news online", *Science*, v. 359, n. 6380, pp. 1146–1151, 2018. ISSN: 0036-8075. doi: 10.1126/science.aap9559. Disponível em: <`https://science.sciencemag.org/content/359/6380/1146`>.

[78] SHAO, C., CIAMPAGLIA, G., VAROL, O., et al. "The spread of low-credibility content by social bots", *Nature Communications*, v. 9, 2018.

[79] OMOHUNDRO, S. M. *Five Balltree Construction Algorithms.* Relatório Técnico TR-89-063, International Computer Science Institute, December 1989.

[80] YIANILOS, P. N. "Data Structures and Algorithms for Nearest Neighbor Search in General Metric Spaces". In: *Proceedings of the Fourth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '93, p. 311–321, USA, 1993. Society for Industrial and Applied Mathematics. ISBN: 0898713137.