Hindawi
Complexity
Volume 2022, Article ID 8924468, 20 pages
https://doi.org/10.1155/2022/8924468

WILEY | Hindawi

Research Article

Modeling the Interaction Networks about the Climate Change on Twitter: A Characterization of its Network Structure

Mary Luz Mouronte-López [1,2] **and Marta Subirán** [1,2]

[1]*Higher Polytechnic School, Universidad Francisco de Vitoria, Madrid, Spain*
[2]*Telefónica Chair at Universidad Francisco de Vitoria, Madrid, Spain*

Correspondence should be addressed to Mary Luz Mouronte-López; maryluz.mouronte@ufv.es

Received 12 March 2022; Accepted 20 May 2022; Published 20 June 2022

Academic Editor: Peican Zhu

This work studies the interaction networks (replying, retweeting, and quoting) that arise on Twitter in relation to such a relevant topic as climate change. We detected that the largest connected component of these networks presents low values of average degree and betweenness, as well as a small diameter compared to the total number of nodes in the network. The largest connected component of retweeting and quoting networks also exhibits very low negative assortativity. The quoting and retweeting networks have a more hierarchical structure than the replying network. We also find that the process of emergence of new links in the interaction networks can be properly modeled (with high accuracy) through a Support Vector Machine model using the embeddings provided by the Node2Vec algorithm. A Random Forest model using certain similarity measures as explanatory variables between nodes also provides high accuracy. In addition, we analyze the communities existing in each interaction network by means of the Louvain method. The cumulative probability distributions of hashtags per community are also examined.

1. Introduction

Many real systems can be characterized as graphs, where nodes symbolize objects and links represent the relations between them. The social networks, which consist of individuals and their relations and whose analysis has drawn interest from several fields, can be studied as graphs [1].

Some investigations propose methods of classifying interactions on social networks according to emotions or frameworks for categorizing or comparing approaches that use social context [2–4]. Also, link prediction is a relevant matter, since it allows the identification of hidden links from the observable part of the interaction networks or the anticipation of future links from the current network topology. Pieces of research exist which carry out a comprehensive review, analyzing and discussing the state of the art of the link prediction on social networks [5]. Node-based metrics, topology-based metrics, and social-theory-based metrics are studied. Additionally, several growth models have been proposed in the literature [6, 7]. Barabási and Albert's growth model [8] also named preferential attachment model which has been frequently utilized to generate scale-free networks and provided a foundation for understanding the mechanisms that give rise to certain properties in various real networks. There are also investigations that propose that the degree is not the only key factor in influencing the growth of scale-free networks, as also a "fitness" exists in each node which symbolizes its propensity to attract links [9, 10]. Research explains that social networks are at best weakly scale-free [11], exhibiting different characteristics. In [11], the authors describe that, considering the analyzed 251 social networks, half of them lack any direct (power law is itself a good model of the degrees) or indirect evidence (power-law distribution is not necessarily a good model of the degrees, but it is a relatively better model than alternatives) of scale-free structure (50% not scale-free), while indirect evidence is slightly less prevalent (41% superweak).

This research aims to examine the interaction networks on Twitter which are concerned with climate change. In these networks, the nodes are Twitter users, and the links are

the exchanges that exist between them. A total of 631,027 tweets were analyzed. The objectives of this investigation are to describe the structure and to characterize the link formation process in replying, retweeting, and quoting networks. Additionally, we aim to discover if within these networks there are communities of users with common interaction patterns. To our knowledge, this has not been done previously in relation to a matter such as climate change.

2. Materials and Methods

2.1. Overview of Used Resources

2.1.1. T-Hoarder Tool. For downloading data from Twitter, the T-Hoarder tool [12] was utilized. It is a software program that is able to perform tweet crawling and data filtering and display summary information about Twitter activity on a specific topic. The tool provides two APIs (Application Programming Interfaces) to download data: the Rest API and the Streaming API. The first works in a synchronous way, with the restriction of searching for data from the previous week. The second makes it possible to carry out real-time data downloading.

T-Hoarder tool is implemented in UNIX as an operating system and utilizes Python as its programming language. Using T-Hoarder, the following data can be collected for each Tweet: tweet ID, timestamp, author, text, app, author ID, followers' author, following author, statuses author, location, URL, geolocation, name, bio, URL media, type media, and lang. A more detailed description of these fields can be found in the Supplementary Materials (available here) Document.

A total of 839,968 tweets were downloaded from Twitter from March 2021 to May 2021.

2.1.2. Software Programs. Several programs in Python [13] and R [14] were developed for the purpose of carrying out the following functionalities:

(i) Handling of the data, which was performed through R language, and pandas library in Python was used.

(ii) Network characterization and graph handling, which was performed by applying the Networkx package in Python [15] and the igraph package in R [16].

(iii) Modeling was carried out utilizing the scikit-learn and StellarGraph packages in Python. The caret, LiblineaR, and e1071 packages were also used in R. The estimation of similarities between nodes was done utilizing the link prediction package in R.

(iv) Communities using CDLIB [17].

The Gephi platform was used to draw the interaction networks [18].

2.2. Overview of Used Methods

2.2.1. Obtaining the Tweets. A total of 839,968 tweets were downloaded from Twitter utilizing the T-Hoarder tool from March to May 2021. The tweets include messages produced from Twitter users as well as their interactions (retweeting, replying, and quoting). It must be noted that only tweets in the English language are considered. For the purpose of deciding on the most appropriate keywords to filter out the tweets which are unrelated to climate change, a small group of individuals were gathered in order to perform the selection. In the cases where more than 15 keywords were proposed, they would be grouped utilizing an affinity diagram [19]. After that, the keywords would be filtered utilizing a multiple voting system. In the end, seven keywords were taken. These were global warming, greenhouse effect, climate change, climate crisis, climate disaster, climate emergency, and climate action. Both with and without spacing to consider hashtags, a total of 589,272 retweets, 94,084 replies, and 156,612 quotes were obtained. Additionally, a filtering system is executed to eliminate possible duplicates which could happen from how the data was acquired (Streaming and Rest APIs). After this, 631,027 tweets remained.

2.2.2. Building the Interaction Network. We use graphs to study the interactions network on climate change on Twitter. In each tweet, node1 is the author's ID and node2 is the user's ID with whom this interacts. Because there are interactions with several users within the same tweet, as many links are generated as interactions exist. The types of interaction are replying, quoting, and retweeting. The generated graphs are undirected and unweighted. Based on each kind of interaction, three types of graphs are created.

The main statistical network parameters are shown in Table 1. These are the number of nodes and links, as well as the maximum and the mean degree. Information on the total connectivity of the network, and other data such as the number of nodes and links, as well as the size of the giant component is also included. Various characteristics of the *GC* are analyzed such as number of nodes and links, diameter (d), average path length ($\langle lp \rangle$), average degree ($< k >$), betweenness centrality ($< bc >$), and assortativity coefficient (r). These metrics are defined as follows.

(i) The degree of a node l, $k(l)$, for an undirected graph, G, such as an interaction network on Twitter, is [20, 21]

$$k(l) = \sum_{i=1}^{N} A_{lj}, \tag{1}$$

where A_{lj} is the element lj of the adjacency matrix. $A_{lj} = 1$ if node l is connected to node j and 0 otherwise. N symbolizes the total number of nodes in the network.

(ii) The betweenness centrality of node l in G, $bc(l)$, is [21, 22]

Table 1: Main structural properties of the interaction networks.

Type of interaction	Number of nodes	Number of links	Maximum degree	$< k >$	Completely connected
Retweeting	294,209	360,387	37,896	2.4	No
Replying	74,152	46,518	275	1.3	No
Quoting	122,512	121,887	13,387	2	No

$$bc(l) = \sum_{u \neq l \neq w \in G} \frac{\sigma_{u,w}(l)}{\sigma_{u,w}}, \quad (2)$$

where $\sigma_{u,w}$ is the total number of shortest paths from node u to node w and $\sigma_{u,w}(l)$ symbolizes the number of those paths that pass through l.

(iii) The degree assortativity [23] is defined as a Pearson correlation between the "expected degree" distribution q_k and the "joint degree" distribution $e_{j,k}$ [23] in G. The first distribution represents the probability distribution of passing through the links of G and discovering nodes with degree k at the termination of the links. The second distribution symbolizes the probability distribution of a link having degree j on one termination and degree k on another termination.

In the undirected case, the normalized Pearson coefficient of $e_{j,k}$ and q_k provides the assortativity coefficient (r) of the network, which can be described as [23, 24]

$$r = \left[\frac{1}{\sigma_q^2}\left(\sum_{jk \in G} jke_{j,k}\right) - \gamma_q^2\right]. \quad (3)$$

Here γ_q and σ_q are the expected value or mean and standard deviation of q_k. If a network is perfectly assortative $(r = 1)$, then its nodes join only with other nodes with analogous degree.

(iv) The average path length of G $(\langle lp \rangle)$ is described as the average number of links that must be passed through the shortest path $sp_{l,j}$ between any two pairs of nodes l and j. If it is considered that $sp_{l,j} = 0$ when $l = j$; that is, if any path between l and j exists, then $\langle lp \rangle$ can be described as

$$\langle lp \rangle = \frac{1}{N * (N-1)} \sum_{l=1}^{N} \sum_{j=1}^{N} sp_{l,j}. \quad (4)$$

(v) The diameter of G, d, symbolizes the length (in number of links) of the longest geodesic path between any two nodes [25].

2.2.3. Modeling the Interaction Network

(1) Overview of the Process. The process of link formation between pairs of nodes is modeled using two mechanisms:

(i) The first procedure uses the Node2Vec algorithm [26]. As is well known, Node2Vec is a semi-supervised method for scalable feature learning in networks, which utilizes a custom graph-based objective function that uses the Stochastic Gradient Descendent method [27]. The algorithm provides a feature representation that maximizes the likelihood of preserving network neighborhoods of nodes in an s-dimensional feature space. A 2^{nd}-order random walk approach is utilized to produce the network neighborhoods for each node [26].

To generate a feature representation of links for two nodes u and v, the authors of the algorithm define a binary operator over the corresponding feature vectors $f(u)$ and $f(v)$ with the purpose of generating a representation $g(u, v)$ such as $g: V \times V \longrightarrow R^s$, where s is the space dimension for the pair (u, v). The operators for any pair of nodes is established even if a link does not exist between them. All operators produce link embeddings that have equal dimensionality to the input node embeddings. Then, given any two nodes, named u and v, and their feature vectors, $f(u)$ and $f(v)$, the operators are defined as follows:

(i) Hadamard: $f(u) * f(v)$
(ii) l1: $|f(u) - f(v)|$
(iii) l2: $||f(u) - f(v)|^2$
(iv) Average: $(f(u) + f(v))/2$

(ii) The second procedure uses certain similarity indexes. In this method, analogously to [28], we calculated local, quasi-local, and global similarity metrics between nodes. Specifically, the local measures were resource allocation [29], Leicht-Holme-Newman [30], common neighbors, cosine [31], cosine similarity on L+ [25], hub promoted [32], Jaccard [33], hub depressed [32], preferential attachment [8], and Sørensen [34]. The global similarity measures used were average commute time [25], Katz [35], L+ directly [25], matrix forest [36], and random walk with restart [37]. Finally, the following quasi-local measures of the similarity were utilized: graph distance and local path [29]. These indexes are explained in detail in the Supplementary Materials Document.

As input variables, the model has either the characteristics that have been obtained for each link, through algorithm, or the similarity indexes between pairs of nodes. The output variable of the model is the mark of whether or not a link exists between a pair of nodes (its value is 1 or 0). Using supervised learning, from examples in which both the input and output variables are known, the model anticipates the value of the output for new inputs, corresponding to cases not utilized in the learning (training process).

Cross validation is applied to generate the model. As an alternative of splitting the dataset into training and testing subsets, in the aforementioned mechanism, δ equal partitions of the dataset are carried out. The model is trained δ times: each time one of the partitions is chosen as a test set, and the model is trained with the remaining $\delta - 1$ folds. Each fold is utilized once as a test set. Hence, at the end, there are various predictions about the whole dataset. Because of the above, δ evaluations of any parameter (PAR) determining the efficiency of the model exist. This parameter can be average [28]:

$$\langle PAR \rangle = \frac{1}{\partial} \sum_{j=1}^{\partial} PAR. \tag{5}$$

In this investigation, we consider PAR, Accuracy, AUC, $F1$, and $GMean$. All these parameters are defined in Section 2.2.2.

Finally, an independent evaluation of the previously indicated parameters is performed utilizing a validation set.

As a dataset to which the cross-validation procedure was applied, we took an amount corresponding to 75% of the total links (t75). The same value for nonexistent links between unconnected pairs of randomly selected nodes was considered (t75). As a validation set, 25% of the total number of links (t25) was contemplated and an analogous value (t25) was taken for nonexistent links.

(2) Obtaining Embeddings through Node2Vec. Node embeddings are calculated through Node2Vec in such manner that the nodes which are close in the graph remain close in the embedding space. It is a two-stage process which first involves running random walks on the graph to obtain context pairs and second uses these walks to train a Word2Vec model [26]. To compute the embeddings, we use the StellarGraph package in Python, and several parameters must be specified:

(i) p, which controls the probability in a walk of going back to the node from which one comes. Values in [0.1, 2] were taken in steps of size 0.1.

(ii) q, which manages the probability of exploring undiscovered parts of the graphs. It determines the dimensionality of Node2Vec embeddings, that is, the size of the feature vector. Values in [0.1, 2] were taken in steps of size 0.1.

(iii) num. walks, which defines the number of walks made from each node. Values in [0.1, 10] were chosen in steps of size 0.5.

(iv) walk length, which symbolizes the length of each random walk. Values in [19, 38] were chosen in steps of size 5.

(v) window size, which specifies the context window size for Word2Vec. Values in [39, 40] were selected in steps of size 1.

(vi) num iter, which represents the number of SGD iterations (epochs) to run. Values in [38, 41] were selected in steps of size 1.

To optimize the hyperparameters, we calculate the closeness centrality (or nearness centrality) of all nodes in G. This parameter can be defined as

$$\eta_l = \frac{1}{f_l}, \qquad f_l = \sum_{j \in G - \{l\}} lp_{lj}. \tag{6}$$

In the above formula, f_l is named farness centrality. lp_{lj} symbolizes the number of links in the shortest path between nodes l and j.

Considering the embeddings obtained for each node l, according to each selected hyperparameter option, we calculate the metric ηe_l, which can be described as

$$\eta e_l = \frac{1}{f_{el}}, \qquad f_{el} = \sum_{j \in G - \{l\}} de_{lj}. \tag{7}$$

In the above formula, de_{lj} is the Euclidean distance between the vectors corresponding to l and j nodes. Finally, the correlation between $\eta = \{\eta l\}$ and $\eta e = \{\eta e_l\} \nabla l \in G$ is obtained.

The Pearson correlation [42] or the Spearman correlation [42] would be utilized, depending on whether η and ηe variables would exhibit a normal distribution or not. The normality of the distribution is checked utilizing the Anderson-Darling test [43] with a significance level $\alpha = 0.05$. The considered hypotheses are as follows:

(i) H_0: "the samples derived from a normal distribution."

(ii) H_a: "the samples did not derive from a normal distribution."

If $p - \text{value} \leq \alpha$, H_0 must be rejected and H_a is taken; otherwise H_0 must be accepted.

One combination of hyperparameters for which a correlation greater than 0.9 is obtained was selected. We use $p = 1.0$, $q = 1.0$, dimensions = 64, num. walks = 5, walk length = 50, window size = 10, and num iter = 1.

It must be noted that if a new node is added to the network, the execution of Word2Vec is required on the whole graph to generate the new embeddings.

(3) Obtaining Similarities between Nodes. As we have previously indicated, to calculate the similarity between nodes, the link prediction package in R is used. The similarity indexes are described in the Supplementary Material Document.

2.2.4. Building the Model. Three models were tested: Random Forest (RF) [44], Logistic Regression (LR) [45], and Support Vector Machines (SVM) [46] model. Each model is optimized by 5-fold cross-validated grid search over a

parameter grid in order to find the best parameters for each one.

The link prediction package is only available in R language. As a result, we implement the model in R, if the similarity indexes are used as explanatory variables, whereas if the embeddings are obtained from Node2Vec, it is developed in Python language.

In Python language, the applied hyperparameters for the RF model are as follows: number of trees in the forest (n_estimators) and minimum number of samples required to split an internal node (min_samples_split). In R, the utilized hyperparameters are as follows: number of trees in the forest (max_depth) and number of variables randomly sampled as candidates at each split (mtry) (minimal node size (min.node.size) is equal to 1). Minimal node size Gini is taken as splitting rule in both R and Python languages.

For LR and SVM models, the hyperparameters are identical in R and Python languages. They are, for the LR model, the best inverse of regularization strength (Cs) and an L2 [47] penalty term (penalty). For the SVM model, the inverse of regularization strength (C) and Kernel coefficient (gamma) are utilized.

(i) RF model (Python): n_estimators: [100, 150, 300], max_depth: [3, 5, 10, None], and min_samples_split: [2, 5, 10]

(ii) RF model R: num.trees: [100, 150, 300], mtry: [2, 5, 10], and max_depth: none

(iii) LR model (Python and R): Cs: [1, 5, 10, 20], penalty: [l1, l2] (caret package with ranger method, which we used to build the RF, does not allow us to specify limits for max_depth).

(iv) SVM model (Python and R): C: $10^{[-2,-1,0,12]}$, gamma: $10^{[-2,-1,0,12]}$

To measure the performance of the model, the following metrics are computed:

(i) Area under curve (AUC). Receiver Operating Characteristic curve (ROC) is a probability curve where each point symbolizes a true positive rate (TPR)/false positive rate (FPR) pair corresponding to a one decision threshold. TPR and FPR can be described as

$$\begin{aligned} \text{TPR} &= \frac{\text{TP}}{\text{TP} + \text{TN}}, \\ \text{FPR} &= \frac{\text{FP}}{\text{TP} + \text{TN}}. \end{aligned} \tag{8}$$

In the above formula, TP, TN, FP, and FN symbolize the true positives, the true negatives, the false positives, and the false negatives, respectively.

If ROC*(t)* is the function associated with the ROC curve, the area under curve (AUC) can be denoted as

$$\text{AUC} = \int_0^1 ROC(t)\mathrm{d}t, \tag{9}$$

The AUC allows us to estimate the performance of the classifier, establishing its ability to discriminate between classes; that is, 0 indicates that there is no link between a pair of nodes and 1 indicates that a link between a pair of nodes exists.

(ii) Accuracy, in binary classification, as the one we are dealing with, symbolizes the proportion of correct predictions (both true positives (TP) and true negatives (TN)) among the total number of cases examined. It is computed as follows [28]:

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}}. \tag{10}$$

(iii) Sensitivity or Recall, which represents the ability of the classifier to correctly identify the positive samples (1: a link between a pair of nodes exists), can be defined as [28]

$$\text{Sensitivity} = \frac{\text{TP}}{\text{TP} + \text{FN}}. \tag{11}$$

(iv) Specificity or Selectivity, which symbolizes the ability of the classifier to identify a negative sample (there is any link between nodes), is defined as [28]

$$\text{Specificity} = \frac{\text{TN}}{\text{TN} + \text{FP}}. \tag{12}$$

(v) Precision is defined as

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}}. \tag{13}$$

(vi) F1 score can be interpreted as a weighted average of the precision, that is, the ability of the classifier not to label a sample as positive that is actually negative, and recall the ability of the classifier to find all the positive samples. This score is in the range [0-1] and it is defined as

$$F1 = 2 * \frac{\text{Precision} * \text{Recall}}{\text{Precision} + \text{Recall}}. \tag{14}$$

(viii) Geometric mean measures the balance between classification performances in both the majority and minority classes. A low GMean indicates poor performance in the classification of the positive cases even if the negative cases are correctly classified. This score is defined as

$$GMean = \sqrt{\text{Sensitivity} * \text{Specificity}}. \tag{15}$$

2.2.5. Community Analysis. According to [48], one of the more analyzed forms within large-scale networks is the modular structure. In this context, a community is a dense subnetwork, that is, a set of densely connected nodes within a larger network. These conglomerates can be revealed utilizing the information encoded in the network topology. To analyze the presence of communities in the replying,

retweeting, and quoting interactions networks, various algorithms are evaluated.

(1) Louvain Algorithm. This method consists of both modularity optimization and community aggregation phases. In the first stage, each node is initially allocated to a community. After that, the corresponding modularity is estimated by eliminating node l from its community and placing it in its neighbor j's community. If a gain exists in the modularity, l is moved to this new community; otherwise, it remains in its original community. This mechanism is repeated for all nodes in the network [49].

$$\Delta Q = \left[\frac{\sum_{\text{in}}+k_{l,in}}{2*m} - \left(\frac{\sum_{\text{tot}}+k_l}{2*m}\right)^2\right] - \left[\frac{\sum \text{in}}{2*m} - \left(\frac{\sum \text{tot}}{2*m}\right)^2 - \left(\frac{k_l}{2*m}\right)^2\right], \quad (16)$$

where $\sum$ in symbolizes the sum of all the weights of the links inside the community that l is moving into. $\sum$ tot represents the sum of all the weights of the links to the nodes of the community to which i moves. k_l is the weighted degree of l. $k_{l,in}$ is the sum of the weights of the links between l and other nodes in the community that l is moving into. m is the sum of the weights of all links in the network. If the network is unweighted, the weight of each of its links is 1.

In a second stage, a new network is constructed in which the nodes are the communities obtained in the previous stage. The two stages are repeated until modularity cannot be increased further [49].

(2) Leiden Algorithm. It is based on the Louvain method. This procedure introduces a refinement phase in addition to the modularity optimization and community aggregation phases, making it slightly more complex [50]. Analogously to the Louvain algorithm, this algorithm also begins by allocating each node to a community. After that, individual nodes are moved from one community to another to obtain a gain in modularity. The next step involves the refinement of the individual communities found in the previous step. This refined partition (RP) is obtained as follows.

Initially, the refined partition (RP) is set to a unique partition, in which each node is in its own community [50]. The algorithm then locally brings nodes together in RP: nodes that are on their own in a community in RP can be merged with a different community. It should be noted that mergers are performed only within each community of the partition obtained in the first stage. In addition, a node is joined to a community in RP only if both are sufficiently well joined to their community in the first stage [50].

After the refinement step, the primary community might split into multiple subcommunities ensuring well-connected communities. After that, assembly of the community nodes is carried out based on the refined partition RP. These steps are repeated until no more improvement can be made in terms of modularity [50].

(3) Label Propagation Algorithm (LPA). It operates as follows: first, the network is initialized, so that each node is allocated a unique label. Then, every node selects a great number of neighbors, applying a label to itself. If more than one label is utilized by the same maximum number of neighbors, one of them is selected at random. After various repetitions, the identical label tends to be connected with all elements of a conglomerate [51]. LPA reaches convergence when each node has the majority label of its neighbors [52].

(4) Surprise Algorithm. [39, 41, 53] The authors propose a different global performance measure which is named "Surprise" to evaluate the computation of conglomerates. They establish the community structure of a network computing the distributions of intra- and intercommunity links with a cumulative hypergeometric distribution [54]. The method assumes that a null model exists through which links between nodes emerge at random. The departure of the observed partition from the expected distribution of nodes and links into conglomerates is measured considering this null model. The following cumulative hypergeometric distribution is utilized [39]:

$$S = -\log \sum_{j=p}^{M \text{ in } (M,n)} \frac{\binom{M}{j}+\binom{F-M}{n-j}}{\binom{F}{n}}, \quad (17)$$

where F is the maximum possible number of links in a network $[k^2 - k]/2$, with k being the number of nodes. n is the observed number of links, M is the maximum possible number of intracommunity links for a specific partition, and p is the total number of intracommunity links observed in that specific partition.

This parameter makes it possible to estimate the exact probability of the distribution and nodes for the established communities in the network for a specific partition [53].

The four aforementioned algorithms have been chosen because they have been shown to be effective in several forms of research [39, 50, 55, 56] and because, for the analyzed network, their execution is completed in a short time (less than 2 minutes), when run on a computer with the following characteristics:

(i) Processor: 11th Gen Intel(R) Core(TM) i7-1160G7 @ 1.20 GHz 2.11 GHz

(ii) RAM: 16.0 GB

Three metrics are calculated to assess the performance of the community detection algorithms:

(i) Modularity [13], which evaluates the strength of divisions. A large modularity represents dense connection between nodes within communities and sparse connections between nodes located in different communities. Then the modularity Q of a partition is defined as [13]

$$Q = \frac{1}{2*m}\sum_{lj\in G}\left(A_{lj} - \gamma\frac{k_l * k_j}{2*m}\right)\partial\left(c_l, c_j\right), \quad (18)$$

where m is the number of links, A_{lj} is the lj element of the adjacency matrix of G, k_l k_j are the degrees of

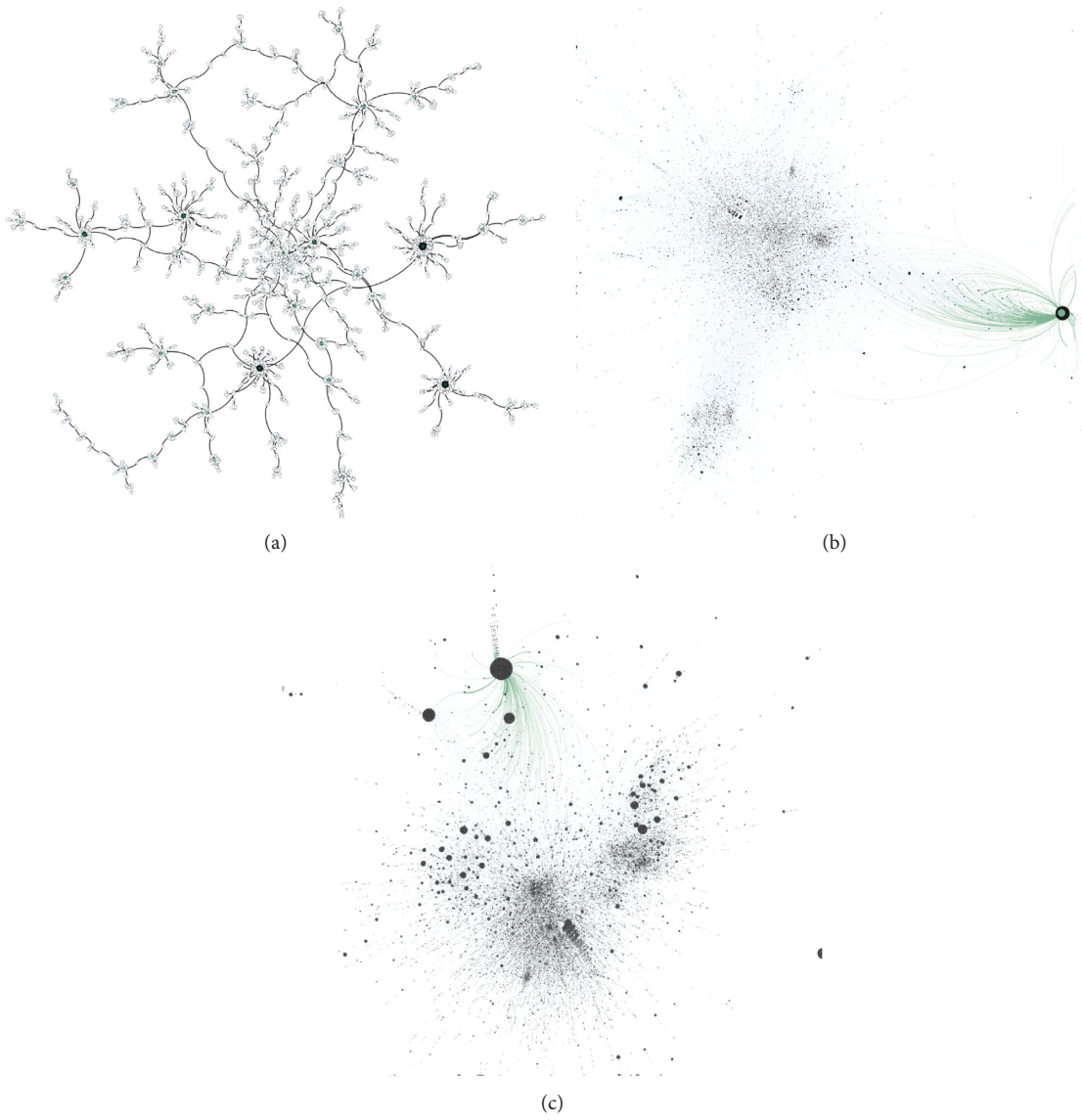

Figure 1: From March 11 until May 26 in 2021 replying (a), retweeting (b), and quoting (c) interactions networks.

Table 2: Main structural properties of the *GC* of the interaction networks.

Type of interaction	Number of nodes	Number of links	<*lp*>	*d*	<*k*>	Assortativity	<*bc*>
Retweeting	237,968	323,773	6.49178	24	2.72115	−0.07357	0.00002
Replying	2,174	2,194	18.05216	47	2.01840	−0.21675	0.007851
Quoting	84,581	98,219	6.42998	22	2.32248	−0.09679	0.00006

nodes l and j, and $\partial(c_l, c_j)$ is a resolution parameter, which is equal to 1 if l and j are in the same community and 0 otherwise. γ is a resolution parameter. If it is lower than 1, Q inclines towards larger communities. If, on the contrary, its value is higher than 1, Q is in favor of smaller communities. A value equal to 1 is taken in this investigation.

The value of modularity is in the range [−1/2, 1] for unweighted and undirected graphs [13, 58].

(ii) Performance [58].

$$P(P) = \frac{\left|\left\{(l,j) \in E, C_l = C_j\right\}\right| + \left|\left\{(l,j) \notin E, C_l \neq C_j\right\}\right|}{n * (n-1)/2}, \tag{19}$$

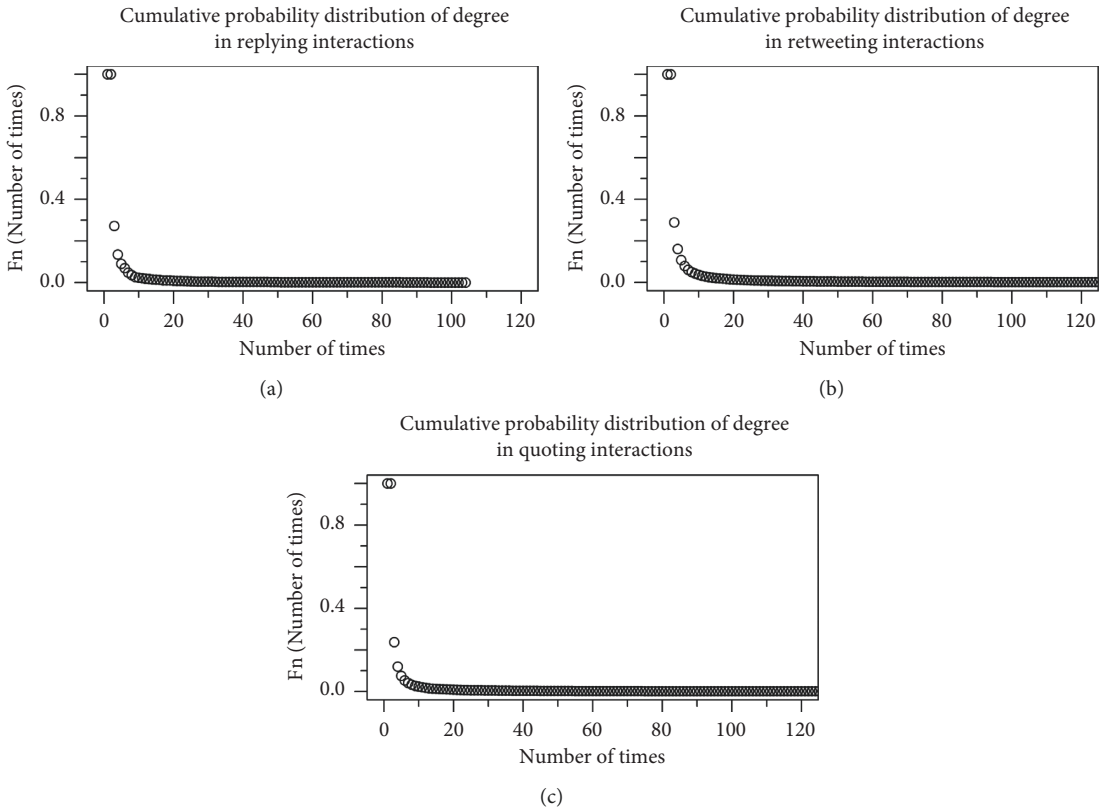

Figure 2: From March 11 until May 26 in 2021, cumulative probability distribution of degree in replying (a), retweeting (b), and quoting (c) interactions.

Table 3: Utilizing the obtained embeddings through Node2Vec, best hyperparameters for LR, SVM, and RF models for replying interaction from March 12 until April 12 in 2021[1].

LR_Hadamard	LR_l1	LR_l2	LR_avg
Cs: 20, penalty: l1	Cs: 5, penalty: l1	Cs: 20, penalty: l2	Cs: 5, penalty: l2
SVM_Hadamard	SVM_l1	SVM_l2	SVM_avg
C: 100.0, gamma: 0.01	C: 0.01, gamma: 0.01	C: 0.1, gamma: 0.01	C: 10.0, gamma: 0.01
RF_Hadamard	RF_l1	RF_l2	RF_avg
max_depth: 3, min_samples_split: 10, n_estimators: 300	max_depth: 10, min_samples_split: 2, n_estimators: 150	max_depth: 5, min_samples_split: 2, n_estimators: 150	max_depth: none, min_samples_split: 5, n_estimators: 150

[1]LR_Hadamard, SVM_hadamard, and RF_hadamard: LR, SVM, and RF models using the Hadamard operator. LR_l1, SVM_l1, and RF_l1: LR, SVM, and RF using operator l1; LR_l2, SVM_l2, and RF_l2: LR, SVM, and RF utilizing the l2 operator; LR_average, SVM_average, and RF_average: LR, SVM, and RF applying the average operator.

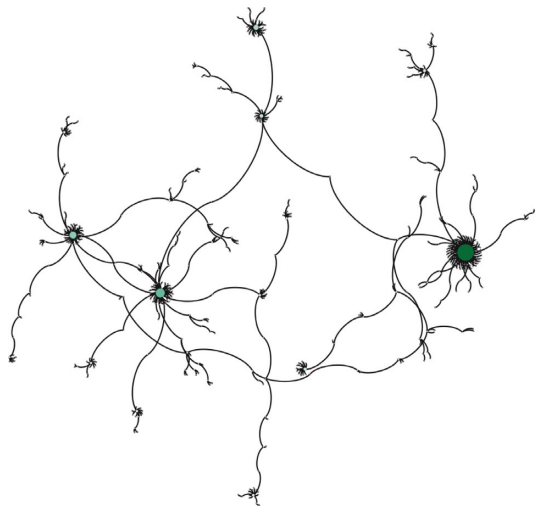

Figure 3: For replying interactions, new links between nodes from March 12 until April 12 in 2021.

Table 4: For replying interaction from March 12 until April 12 in 2021, utilizing the obtained embeddings through Node2Vec, performance of the LR and SVM models with operators Hadamard, l1, l2, and average[1,2].

Metrics	LR_Hadamard	LR_l1	LR_l2	LR_avg	SVM_Hadamard	SVM_l1	SVM_l2	SVM_avg
Acc_tr + ts	0.96809	0.93617	0.95745	0.80851	0.96809	0.95745	0.94681	0.83511
Acc_val	0.96813	0.96813	0.97211	0.79283	0.97211	0.97610	0.95618	0.81275
F1_tr + ts	0.96202	0.92500	0.95062	0.77500	0.96154	0.95062	0.93902	0.81212
F1_val	0.96875	0.96850	0.97276	0.79365	0.97255	0.97656	0.95785	0.81569
ROC_tr + ts	0.99216	0.95531	0.95215	0.85714	0.99239	0.95144	0.95203	0.86089
ROC_tr + ts	0.98844	0.97200	0.97048	0.87422	0.98883	0.97060	0.97041	0.87689
GMean_tr + ts	0.97085	0.93974	0.96329	0.80800	0.96898	0.96329	0.95389	0.83995
GMean_val	0.96793	0.96806	0.97183	0.79283	0.97199	0.97590	0.95535	0.81260

[1]LR_Hadamard, SVM_Hadamard, and RF_Hadamard: LR, SVM, and RF models using the Hadamard operator. LR_l1, SVM_l1, and RF_l1: LR, SVM, and RF models using operator l1; LR_l2, SVM_l2, and RF_l2: LR, SVM, and RF utilizing operator l2; LR_average, SVM_average, and RF_average: LR, SVM, and RF models applying the average operator. [2]In cross-validation procedure, Acc_tr + ts, F1_tr + ts, ROC_tr + ts, and GMean_tr + ts represent the average Accuracy, F1, ROC, and GMean. In final validation, Acc_val, F1_val, ROC_val, and GMean_val symbolize Accuracy, F1, ROC, and GMean.

Table 5: For replying interaction from March 12 until April 12 in 2021, utilizing the obtained embeddings through Node2Vec, performance of the RF model with operators Hadamard, l1, l2, and average[1,2].

Metrics	RF_Hadamard	RF_l1	RF_l2	RF_avg
Acc_tr + ts	0.93085	0.94150	0.94149	0.94681
Acc_val	0.97211	0.94821	0.95618	0.96414
F1_tr + ts	0.91925	0.92994	0.92994	0.93902
F1_val	0.97255	0.94694	0.95547	0.96471
ROC_tr + ts	0.97373	0.98444	0.97520	0.99485
ROC_ts	0.97965	0.98921	0.98273	0.99683
GMean_tr + ts	0.93513	0.94248	0.94248	0.95389
GMean_val	0.97199	0.94791	0.95605	0.96402

[1]LR_Hadamard, SVM_Hadamard, and RF_Hadamard: LR, SVM, and RF models using the Hadamard operator. LR_l1, SVM_l1, and RF_l1: LR, SVM, and RF models using operator l1; LR_l2, SVM_l2, and RF_l2: LR, SVM, and RF utilizing operator l2; LR_average, SVM_average, and RF_average: LR, SVM, and RF models applying the average operator. [2]In cross-validation procedure: Acc_tr + ts, F1_tr + ts, ROC_tr + ts, and GMean_tr + ts represent the average Accuracy, F1, ROC, and GMean. In final validation, Acc_val, F1_val, ROC_val, and GMean_val symbolize Accuracy, F1, ROC, and GMean.

Table 6: For replying interactions from March 12 until April 12 in 2021, utilizing the obtained similarities between nodes, best hyperparameters for LR, SVM, and RF models.

LR	SVM	RF
Cs: 5; penalty: l1	C: 0.1; gamma: 1	num.trees = 300; mtry = 5

Table 7: For replying interactions from March 12 until April 12 in 2021, utilizing the similarities between nodes, performance of the LR, SVM, and RF models[1].

Metrics	LR	SVM	RF
Acc_tr + ts	0.85638	0.93617	0.996
Acc_val	0.83600	0.93617	1
F1_tr + ts	0.85246	0.94000	0.99601
F1_val	0.82700	0.94000	1
ROC_tr + ts	0.95439	0.96247	0.99600
ROC_ts	0.83600	0.94205	1
GMean_tr + ts	0.85597	0.93399	0.99600
GMean_val	0.83438	0.93399	1

[1]In cross-validation procedure: Acc_tr + ts, F1_tr + ts, ROC_tr + ts, and GMean_tr + ts represent the average Accuracy, F1, ROC, and GMean. In final validation, Acc_val, F1_val, ROC_val, and GMean_val symbolize Accuracy, F1, ROC, and GMean.

where l and j are nodes l and j, $\{l, j \in E, C_l = C_j\}$ represents two nodes belonging to identical community and joined by a link, $\{(l, j) \notin E, C_l \neq C_j\}$ represents two nodes belonging to different communities and not joined by a link, and C_l, C_j are the communities where nodes l and j are located. n is the number of nodes in G.

$$0 \leq P(P) \leq 1. \tag{20}$$

Table 8: For retweeting interaction from May 21 until May 26 in 2021, utilizing the obtained embeddings through Node2Vec, best hyperparameters for LR, SVM, and RF models[1].

LR_Hadamard	LR_l1	LR_l2	LR_avg
}textttCs: 10, penalty: l2	Cs: 10, penalty: l2	Cs: 5, penalty: l2	Cs: 20, penalty: l1
SVM_Hadamard	SVM_l1	SVM_l2	SVM_avg
C: 10.0, gamma: 0.01	C: 0.01, gamma: 0.01	C: 1.0, gamma: 0.01	C: 100.0, gamma: 0.01
RF_Hadamard	RF_l1	RF_l2	RF_avg
max_depth: none, min_samples_split: 2, n_estimators: 150	max_depth: 10, min_samples_split: 5, n_estimators: 150	max_depth: 10, min_samples_split: 5, n_estimators: 100	max_depth: none, min_samples_split: 2, n_estimators: 300

[1]LR_Hadamard, SVM_Hadamard, and RF_Hadamard: LR, SVM, and RF models using the Hadamard operator. LR_l1, SVM_l1, and RF_l1: LR, SVM, and RF using operator l1; LR_l2, SVM_l2, and RF_l2: LR, SVM, and RF models utilizing operator l2; LR_average, SVM_average, and RF_average: LR, SVM, and RF models applying the average operator.

Table 9: For retweeting interaction from May 21 until May 26 in 2021, utilizing the obtained embeddings through Node2Vec, performance of the LR and SVM models with operators Hadamard, l1, l2, and average[1,2].

Metrics	LR_Hadamard	LR_l1	LR_l2	LR_avg	SVM_Hadamard	SVM_l1	SVM_l2	SVM_avg
Acc_tr + ts	0.97417	0.95868	0.96074	0.73037	0.97934	0.96178	0.95971	0.73554
Acc_val	0.96902	0.95585	0.96127	0.77924	0.97521	0.96204	0.95817	0.77769
F1_tr + ts	0.97492	0.95910	0.96154	0.73286	0.97963	0.96251	0.96049	0.74245
F1_val	0.96820	0.95362	0.95994	0.76998	0.97436	0.96083	0.95666	0.77348
ROC_tr + ts	0.99772	0.97494	0.97534	0.79600	0.99646	0.97489	0.97541	0.80181
ROC_ts	0.99484	0.97656	0.97642	0.82149	0.99221	0.97631	0.97565	0.81815
GMean_tr + ts	0.97375	0.95865	0.96055	0.73034	0.97927	0.96161	0.95954	0.73507
GMean_val	0.96983	0.95578	0.96184	0.77913	0.97581	0.96269	0.95868	0.77840

[1]LR_Hadamard, SVM_Hadamard, and RF_Hadamard: LR, SVM, and RF using the Hadamard operator. LR_l1, SVM_l1, and RF_l1: LR, SVM, and RF models using operator l1; LR_l2, SVM_l2, and RF_l2: LR, SVM, and RF models utilizing operator l2; LR_average, SVM_average, and RF_average: LR, SVM, and RF models applying the average operator. [2]In cross-validation procedure, Acc_tr + ts, F1_tr + ts, ROC_tr + ts, and GMean_tr + ts,represent the average Accuracy, F1, ROC, and GMean. In final validation, Acc_val, F1_val, ROC_val, and GMean_val symbolize Accuracy, F1, ROC, and GMean.

Table 10: For retweeting interaction from May 21 until May 26 in 2021, utilizing the obtained embeddings through Node2Vec, performance of the RF model with operators Hadamard, l1, l2, and average[1].

Metrics	RF_Hadamard	RF_l1	RF_l2	RF_avg
Acc_tr + ts	0.96694	0.95145	0.94731	0.96901
Acc_val	0.96669	0.94888	0.95120	0.96669
F1_tr + ts	0.96674	0.95130	0.94693	0.96945
F1_val	0.96467	0.94536	0.94798	0.96541
ROC_tr + ts	0.99514	0.98681	0.98672	0.99576
ROC_ts	0.991664	0.98562	0.98520	0.99601
GMean_tr + ts	0.96696	0.95147	0.94732	0.96893
GMean_val	0.96614	0.94783	0.95031	0.96713

[1]LR_Hadamard, SVM_Hadamard, and RF_hadamard: LR, SVM, and RF models using the Hadamard operator. LR_l1, SVM_l1, and RF_l1: LR, SVM, and RF models using operator l1; LR_l2, SVM_l2, and RF_l2: LR, SVM, and RF models utilizing operator l2; LR_average, SVM_average, and RF_average: LR, SVM, and RF applying the average operator.

(iii) Coverage.

It can be defined as the ratio of the number of intra-community links by the total number of links [58].

High values of Q correspond with appropriate partitions. Consequently, the partition corresponding to its maximum value in G should be the best [58]. Therefore, to study the interaction networks, the partition provided by the method with a higher value of Q is selected but considering, at the same time, that the method allows us to obtain a good value for the performance and coverage metrics. All this is done with the smallest possible number of communities.

2.2.6. Analysis of Probability Cumulative Distributions of Hashtags. We also analyze the cumulative probability distributions of hashtags for all types of interactions, globally and by community. The Kolmogorov-Smirnov test [38] with a significance level equal to 0.05 is utilized for the comparison of the distributions. The following hypotheses are considered:

(i) Null hypothesis (H_0): "the samples derive from the identical distribution."

(ii) Alternative hypothesis (H_a): "the samples derive from different distribution."

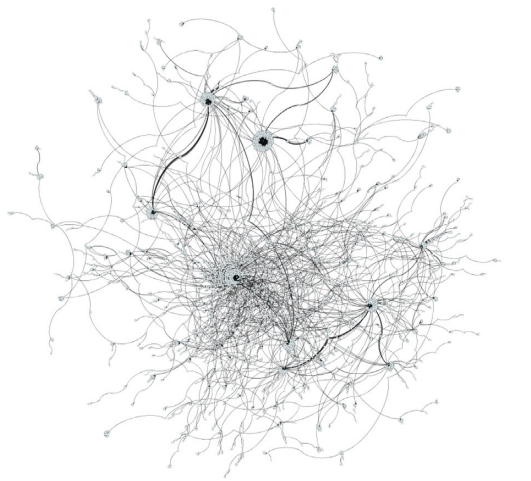

Figure 4: For retweeting interactions, new links between nodes from May 21 until May 26 in 2021.

Table 11: For retweeting interactions from May 21 until May 26 in 2021, utilizing the obtained similarities between nodes, best hyperparameters for LR, SVM, and RF models.

LR	SVM	RF
Cs: 20; penalty: l1	C: 0.1; gamma: 10	num. trees = 300; mtry = 10

Table 12: For retweeting interactions from May 21 until May 26 in 2021, utilizing the similarities between nodes, performance of the LR, SVM, and RF models[1].

Metrics	LR	SVM	RF
Acc_tr + ts	0.91632	0.95702	0.99400
Acc_val	0.91395	0.95713	0.99922
F1_tr + ts	0.91904	0.95795	0.99401
F1_val	0.91747	0.95795	0.99923
ROC_tr + ts	0.91632	0.95697	0.99400
ROC_ts	0.83600	0.95712	0.99920
GMean_tr + ts	0.91570	0.95676	0.99400
GMean_val	0.91296	0.95693	0.99922

[1]In cross-validation procedure, Acc_tr + ts, F1_tr + ts, ROC_tr + ts, and GMean_tr + ts represent the average Accuracy, F1, ROC, and GMean. In final validation, Acc_val, F1_val, ROC_val, and GMean_val symbolize Accuracy, F1, ROC, and GMean.

Table 13: Utilizing the obtained embeddings for nodes and links through Node2Vec, best hyperparameters for LR, SVM, RF models, for quoting interaction April 12 until April 22 in 2021[1].

LR_Hadamard	LR_l1	LR_l2	LR_avg
Cs: 10, penalty: l2	Cs: 10, penalty: l1	Cs: 20, penalty: l1,	Cs: 10, penalty: l2
SVM_Hadamard	SVM_l1	SVM_l2	SVM_avg
C: 100.0, gamma: 0.01	C: 100.0, gamma: 0.01	C: 1.0, gamma: 0.01,	C: 10.0, gamma: 0.01
RF_Hadamard	RF_l1	RF_l2	RF_avg
max_depth: none, min_samples_split: 2, n_estimators: 300	max_depth: none, min_samples_split: 5, n_estimators: 100	max_depth: none, min_samples_split: 2, n_estimators: 300	max_depth: none, min_samples_split: 2, n_estimators: 150

[1]LR_Hadamard, SVM_Hadamard, and RF_Hadamard: LR, SVM, and RF models using the Hadamard operator. LR_l1, SVM_l1, and RF_l1: LR, SVM, and RF models using operator l1; LR_l2, SVM_l2, and RF_l2: LR, SVM, and RF models utilizing operator l2; LR_average, SVM_average, and RF_average: LR, SVM, and RF applying the average operator.

Table 14: For quoting interactions from April 12 until April 22 in 2021, utilizing the obtained embeddings through Node2Vec, performance of the LR and SVM models with operators Hadamard, l1, l2, and average[1,3].

Metrics	LR_Hadamard	LR_l1	LR_l2	LR_avg	SVM_Hadamard	SVM_l1	SVM_l2	SVM_avg
Acc_tr + ts	0.98798	0.96374	0.96901	0.83853	0.98798	0.96648	0.96796	0.84275
Acc_val	0.98577	0.95178	0.95557	0.85028	0.98688	0.95336	0.95431	0.85296
F1_tr + ts	0.98804	0.96390	0.96932	0.84342	0.98804	0.96670	0.96833	0.84905
F1_val	0.98591	0.95244	0.95647	0.85370	0.98700	0.95407	0.95521	0.85819
ROC_tr + ts	0.99662	0.97397	0.97377	0.87731	0.99652	0.97373	0.97344	0.87393
ROC_ts	0.99671	0.95686	0.95671	0.87804	0.99648	0.95628	0.95626	0.87351
GMean_tr + ts	0.98797	0.96373	0.96896	0.83795	0.98798	0.96646	0.96789	0.84171
GMean_val	0.98573	0.95168	0.95535	0.84996	0.98683	0.95324	0.95410	0.85217

[1]LR_Hadamard, SVM_Hadamard, and RF_Hadamard: LR, SVM, and RF using the Hadamard operator. LR_l1, SVM_l1, and RF_l1: LR, SVM, and RF models using operator l1; LR_l2, SVM_l2, and RF_l2: LR, SVM, and RF models utilizing operator l2,;LR_average, SVM_average, and RF_average: LR, SVM, and RF models applying the average operator. [2]In cross-validation procedure, Acc_tr + ts, F1_tr + ts, ROC_tr + ts, and GMean_tr + ts represent the average Accuracy, F1, ROC, and GMean. In final validation, Acc_val, F1_val, ROC_val, and GMean_val symbolize Accuracy, F1, ROC, and GMean.

Table 15: For quoting interactions from April 12 until April 22 in 2021, utilizing the obtained embeddings through Node2Vec, performance of the RF model with operators Hadamard, l1, l2, and average[1].

Metrics	RF_Hadamard	RF_l1	RF_l2	RF_avg
Acc_tr + ts	0.98841	0.96944	0.97028	0.98714
Acc_val	0.98545	0.96506	0.96569	0.98957
F1_tr + ts	0.98836	0.96914	0.97002	0.98725
F1_val	0.98547	0.96500	0.96566	0.98966
ROC_tr + ts	0.99901	0.99589	0.99638	0.99928
ROC_ts	0.99864	0.99446	0.99524	0.99928
GMean_tr + ts	0.98840	0.96939	0.97024	0.98710
GMean_val	0.98546	0.96506	0.96569	0.98952

[1]In cross-validation procedure, Acc_tr + ts, F1_tr + ts, ROC_tr + ts, and GMean_tr + ts represent the average accuracy, F1, ROC, and GMean. In final validation, Acc_val, F1_val, ROC_val, and GMean_val symbolize Accuracy, F1, ROC, and GMean.

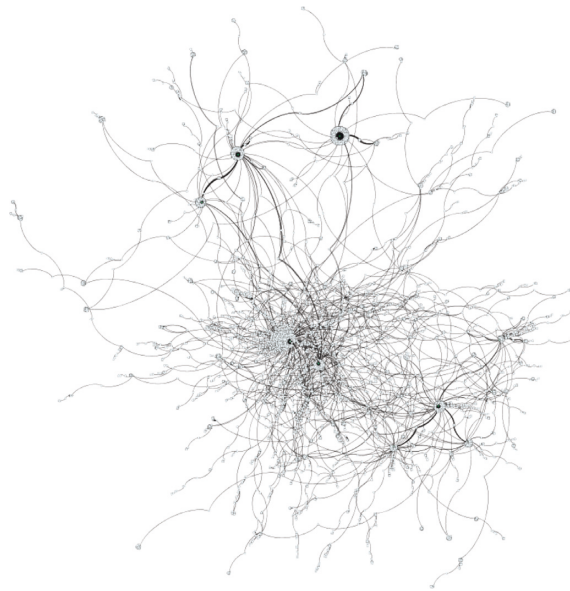

Figure 5: For quoting interactions, new links between nodes from April 12 until April 22 in 2021.

Table 16: For quoting interactions from April 12 until April 22 in 2021, utilizing the obtained similarities between nodes, best hyperparameters for LR, SVM, and RF models.

LR	SVM	RF
Cs: 10; penalty: l1	C: 0.1; gamma: 10	num. trees = 150; mtry = 5.

Table 17: For quoting interactions from April 12 until April 22 in 2021, utilizing the similarities between nodes, performance of the LR, SVM, and RF models.

Metrics	LR	SVM	RF
Acc_tr + ts	0.99088	0.85033	0.99800
Acc_val	0.98972	0.85034	0.99842
F1_tr + ts	0.99087	0.85745	0.99800
F1_val	0.98968	0.85745	0.99842
ROC_tr + ts	0.99088	0.85232	0.99800
ROC_ts	0,98972	0.85034	0.99840
GMean_tr + ts	0.99088	0.85002	0.99800
GMean_val	0.98972	0.84887	0.99842

[1]In cross-validation procedure, Acc_tr + ts, F1_tr + ts, ROC_tr + ts, and GMean_tr + ts represent the average Accuracy, F1, ROC, and GMean. In final validation, Acc_val, F1_val, ROC_val, and GMean_val symbolize Accuracy, F1, ROC, and GMean.

Table 18: For replying interaction networks from March 11 until May 26 in 2021, useful metrics for evaluation of the community detection algorithms, and number of communities identified in each.

Method	Modularity	Performance	Coverage	Number of communities
Louvain	0.95159	0.97536	0.97756	51
LPA	0.79219	0.99417	0.79979	537
Leiden	0.95252	0.97625	0.97756	537
Surprise	0.88817	0.99376	0.89633	269

Table 19: For quoting interaction networks from March 11 until May 26 in 2021, useful metrics for evaluation of the community detection algorithms, and number of communities identified in each.

Method	Modularity	Performance	Coverage	Number of communities
Louvain	0.88874	0.95835	0.92848	196
LPA	0.74547	0.97317	0.76672	10138
Leiden	0.89186	0.95917	0.93067	10138
Surprise	0.63240	0.99945	0.63858	23940

Table 20: For retweeting interactions network from March 8 until May 26 in 2021, useful metrics for evaluation of the community detection algorithms, and number of communities identified in each.

Method	Modularity	Performance	Coverage	Number of communities
Louvain	0.84313	0.94797	0.89374	337
LPA	0.69360	0.97207	0.71618	27,133
Leiden	0.85169	0.94888	0.90046	27,133
Surprise	0.58374	0.99981	0.58809	60,301

If p value < 0.05 is obtained in the test, the null hypothesis must be rejected; otherwise, it must be taken.

3. Results and Discussion

3.1. Main Structural Properties of the Interaction Networks. Figure 1 shows the graphs corresponding to replying, retweeting, and quoting interactions networks for the utilized tweets from March 11 to May 26 .

Tables 1 and Table 2 show the structural properties of the interaction networks from March 11 until May 26 in 2021 as well as their *GC*. As previously mentioned, the betweenness centrality estimates the number of times a node lies on the shortest path between other nodes. With respect to the *GC*, as can be observed in Table 2, the average betweenness is low (< 0.008) in all analyzed networks. This means that there are few users who play an intermediation role linking other users. The average degree is also low (< 2.8), as shown in Figure 2, where many nodes exist with a low degree, while only a few exhibit a high degree. Both average path length and diameter are similar in retweeting and quoting networks. However, these are much higher in the replying interaction network, demonstrating that there is worse connectivity. The connection of a network is determined by its diameter, which defines the capacity of any two nodes to interact with each other.

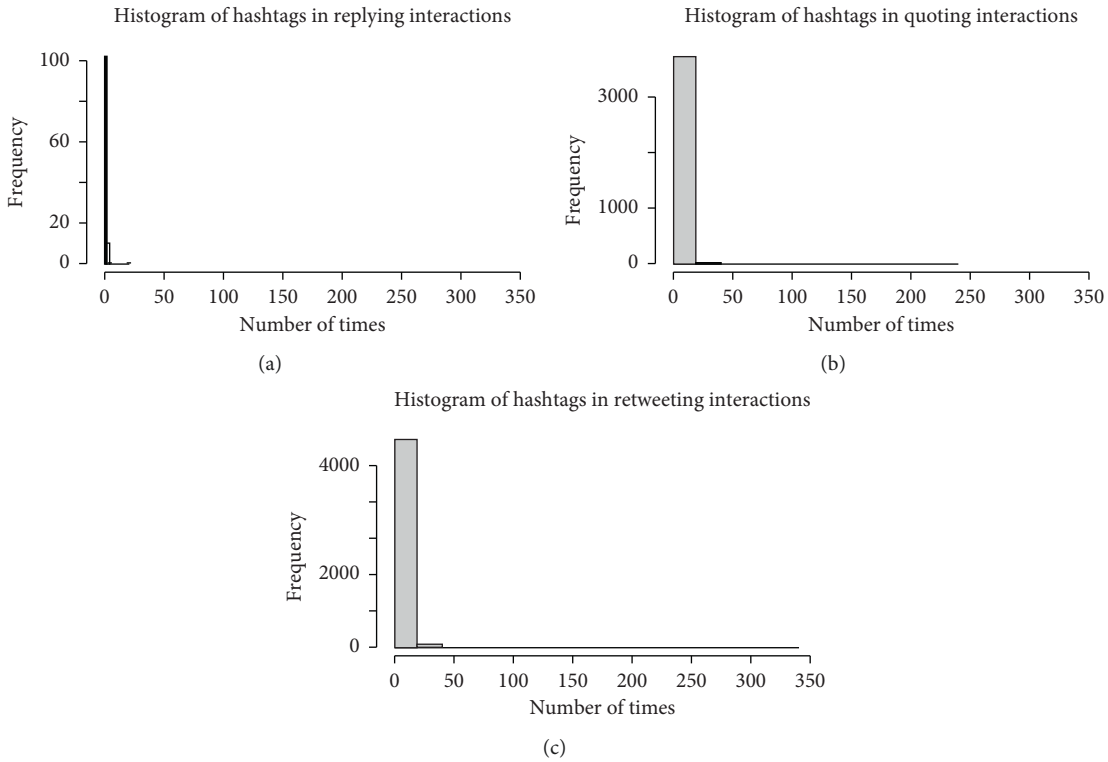

Figure 6: For replying (a), retweeting (b), and quoting (c) interactions, histograms of hashtags.

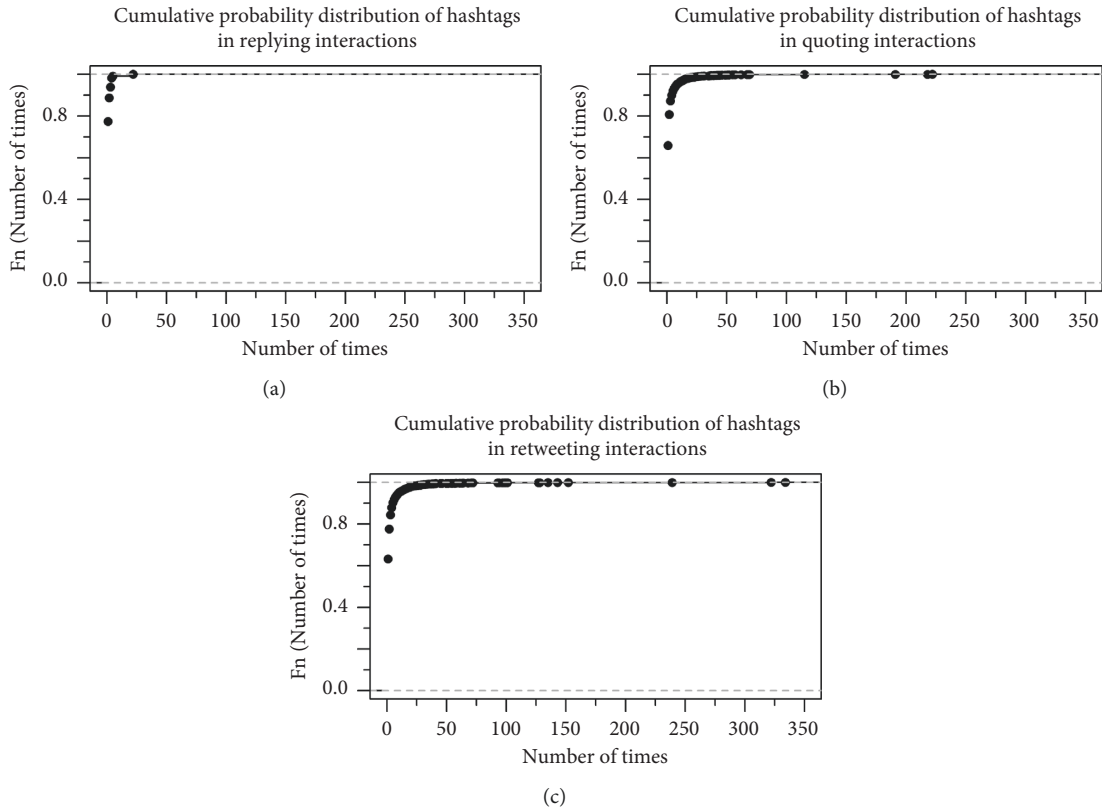

Figure 7: For replying (a), retweeting (b), and quoting (c) interactions, cumulative distribution of hashtags.

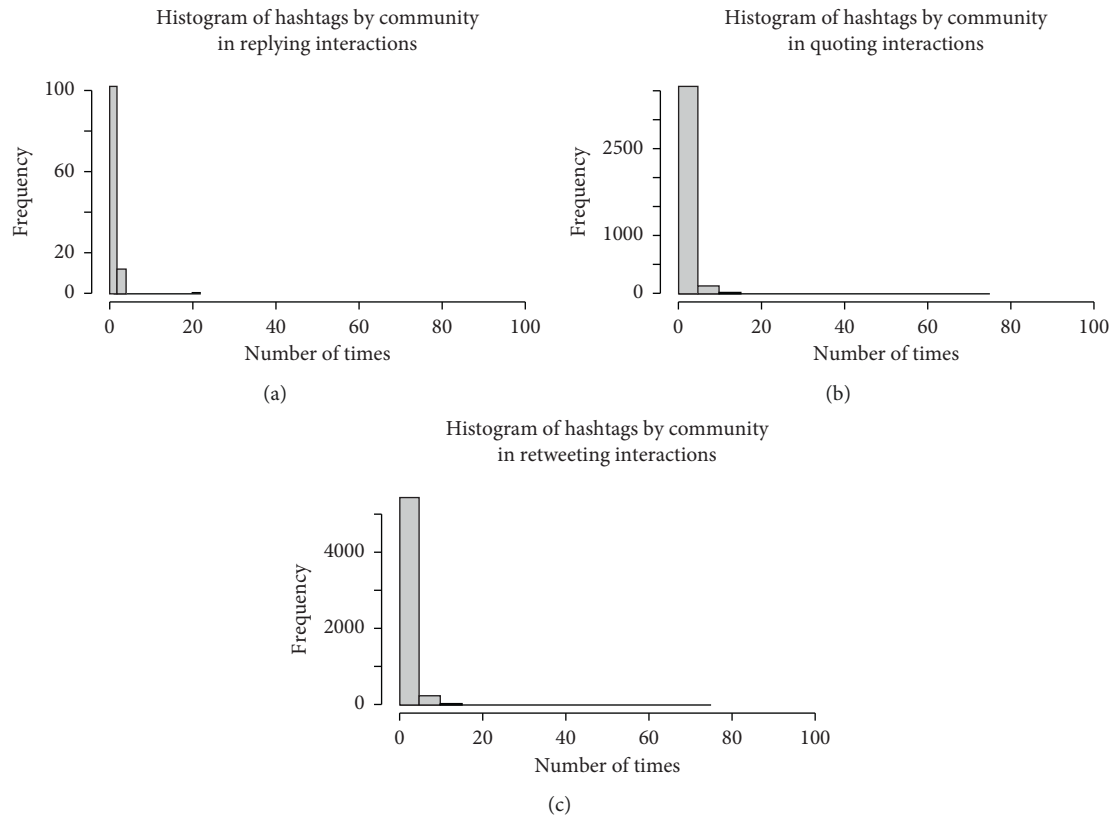

Figure 8: For replying (a), retweeting (b), and quoting (c) interactions, histograms of hashtags by community.

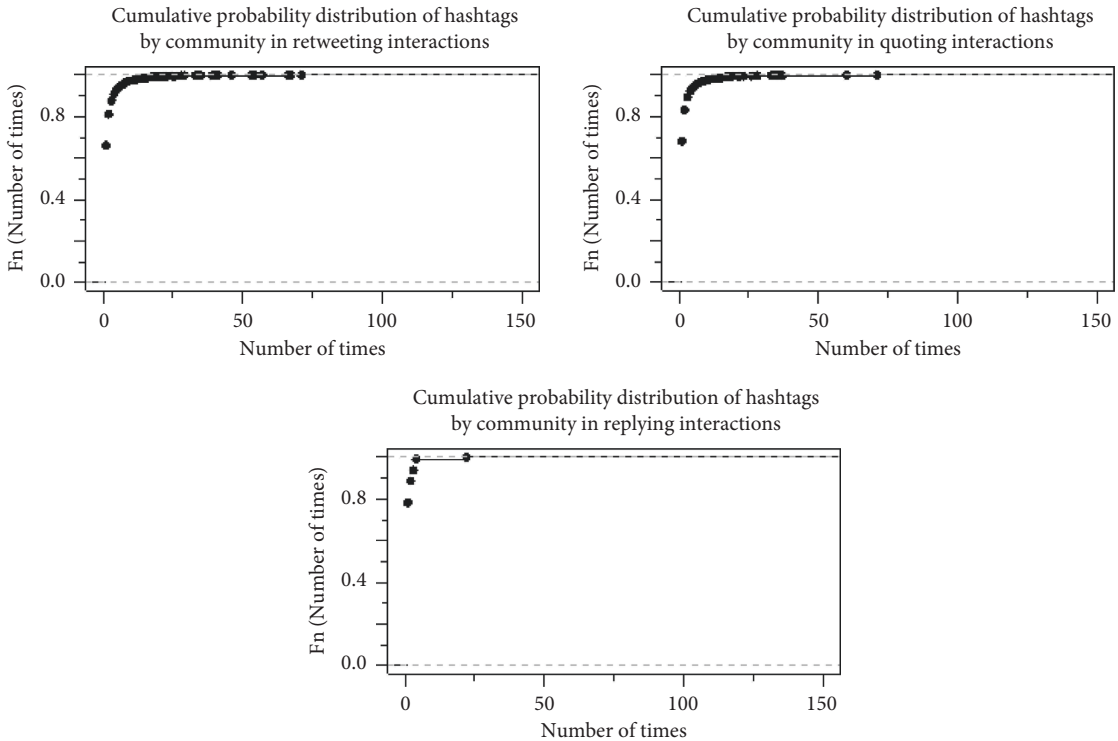

Figure 9: For replying, retweeting, and quoting interactions, cumulative distribution of hashtags by community.

Figure 10: For replying (a), retweeting (b), and quoting (c) interactions, word cloud representation of hashtags.

We also analyze the k-core decomposition [59] in G, which allows us to detect specific subsets (k-cores) in the graph. These are calculated by recursively eliminating all the nodes of degree lower than k, until the degree of all remaining nodes is higher than or equal to k. Those highest values of k correspond to nodes with a higher degree and more central position in G. The k-core decomposition determines a hierarchy of nested subgraphs, in which the 1-core comprises the 2-core, which equally includes the 3-core and so on, until the highest k-core is obtained. Higher values of k imply a more relevant and central subgraph.

We have identified for replying, quoting, and retweeting networks 2, 10, and 22 k-cores. For the replying network, the highest percentage of nodes is in the first k-core (94.16%). Meanwhile, in quoting and retweeting networks, the largest proportion is in the first and second k-core ((83.72/12.66%) and (76.60%/14.37%)). With respect to the percentage, the users registered 5.84% in the highest k-core for replying interaction and 0.04% in the rest of the networks. According to the above, these users that show the highest k-core have a larger relevance.

3.1.1. Modeling the Replying Interactions. Link formation for each type of interaction is modeled at various time periods. This is because, particularly in retweeting and quoting networks, a high number of links exist from March to May 2021. The days on which tweets were downloaded for the periods analyzed are detailed in the Supplementary Material Document.

(1) Modeling from Obtained Embeddings Using Node2Vec. Table 3 depicts the best hyperparameters for each model using Node2Vec for replying interactions from March 11 until April 10, 2021. Figure 3 shows the new interactions for that time (503 new links were formed). Tables 4 and 5 display

the performance metrics for each operator and model used. According to the results, the model that exhibits a higher Accuracy is the SVM model. Specifically, operator l1 is utilized and hyperparameters are taken as C: 0.01 and gamma: 0.01. More periods of analysis are included in the Supplementary Material Document. At those times, the best model was also SVM but using the Hadamard operator, which exhibits a slight difference with respect to operator l1, hyperparameters are taken as C: 100.0 and gamma: 0.01. However, as we see later, a better Accuracy is obtained for both cases by using the similarities between nodes as explanatory variables.

(2) Modeling from Obtained Similarities between Nodes. For replying interactions from March 12 until April 12 in 2021, Table 6 displays the best obtained hyperparameters for each model using the similarity metrics between nodes indicated in 2.2.2. Table 7 depicts the performance metrics for each operator and model utilized. The results show that the best Accuracy is obtained for the RF model, with hyperparameters num.trees = 300, mtry = 5, and min.node.size = 1. This Accuracy is slightly higher than that achieved if the model is built using the obtained embedding applying Node2Vec. The study of other time periods has been incorporated in the Supplementary Material Document. At these times, a slightly better accuracy is obtained for the RF model with hyperparameters num.trees = 300, mtry = 10, and min.node.size = 1.

3.1.2. Modeling the Retweeting Interactions

(1) Modeling from Obtained Embeddings using Node2Vec. In Table 8, the best obtained hyperparameters for each model utilizing Node2Vec for retweeting interactions from May 21 until May 26 in 2021 can be observed. Table 9 and Table 10 display the performance metrics for each operator and applied model. Figure 4 displays the new interactions in the aforementioned period; 2,581 new links were built. Additional time periods have been included in the Supplementary Material Document. Similarly to what happened in the replying interaction networks, for all analyzed time intervals, the best obtained model is SVM, applying the Hadamard operator. The hyperparameters of the model vary depending on the period. Particularly, the utilized hyperparameters for the period specified above were C: 10.0 and gamma: 0.01.

(2) Modeling from Obtained Similarities between Nodes. For retweeting interactions from May 21 until May 26 in 2021, Table 11 displays the best obtained hyperparameters for each model using the similarity metrics according to 2.2.2. In Table 12, the performance metrics for each model used can be seen. More times are described in the Supplementary Material Document. The highest Accuracy is obtained for the RF model, with hyperparameters num.trees = 300, mtry = 10, and min.node.size = 1. This accuracy is slightly higher than the one achieved for the best model using the obtained embeddings through Node2Vec. It can be noted that this happens for all analyzed times.

3.1.3. Modeling the Quoting Interactions

(1) Modeling from Obtained Embeddings using Node2Vec. In Table 13, the best obtained hyperparameters for each model using Node2Vec for quoting interactions from April 12 until April 22 in 2021 are depicted. Table 14 and Table 15 show the performance metrics for each operator and model applied. Additional times are included in the Supplementary Material Document. Figure 5 shows the 12,651 new links between nodes for the aforementioned time period. The best value for the Accuracy is obtained utilizing the Hadamard operator, for the SVM model with hyperparameters max_depth: none, min_samples_split: 2, and n_estimators: 300. The SVM model using the Hadamard operator also exhibits the highest Accuracy for the rest of studied time intervals.

(3) Modeling from Obtained Similarities between Nodes. For quoting interactions from April 12 until April 22 in 2021, Table 16 depicts the best obtained hyperparameters for each model using the similarities between nodes. Table 17 displays the performance metrics for each operator and model utilized. The best accuracy is achieved for the RF model taking as hyperparameters num.trees = 150, mtry = 5, and min.node.size = 1. As in replying and retweeting interaction networks, a slightly higher accuracy than that received using the embedding calculated using Node2Vec is obtained. Other times are described in the Supplementary Material Document. For all times, a higher Accuracy for the RF model utilizing similarities as explanatory variables is also observed.

4. Community Analysis

Tables 18, 19, and 20 show a summary of the three used metrics for the evaluation of the community detection algorithms. The number of communities identified in each method is also shown. We choose the method that gives a good value for all the performance parameters considered and also provides a smaller number of communities.

Once the candidate algorithms to be used for the community detection were analyzed and having selected one of them as the most appropriate, the probability cumulative distribution of hashtags for all interaction networks was checked, both globally and by community. Figures 6–8 and 9 show the collected results. Over the total of 631,027 analyzed tweets, only 28,952 containing hashtags were identified (18,439 retweeting, 10,333 quoting, and 180 replying). All hashtags were formatted from #Word $_1$ Word $_2$. . ..Word t_w to Word $_1$_Word $_2$. . .._Word t_w, where tw is the maximum number of words in each tweet. Those hashtags found more than 100 times in retweeting interactions were as follows: "C_O_P26," "C_E_Ebill," "environment," "C_O_P26," "C_E_E_Bill," "Earth_Day," "C_O_V_I_D19," and "Clean_Delhi." The most frequently used hashtag in replying interactions was "Climate_Brawl" which was contained 22 times. "C_O_P26," "C_E_Ebill," "C_O_P26," and "C_E_E_Bill" hashtags are detected more than 100 times in quoting interactions. Figure 10 shows the word cloud representation of hashtags per interaction type.

5. Conclusions

Climate change and its effects are a relevant topic today. This concern has been highlighted in various international initiatives such as 2030 Agenda for Sustainable Development established by the United Nations [60], the climate emergency declared by the European Parliament [40], or the Net Zero World initiative established by the United States [61]. Social networks are a good chance for people to voice their opinions. Twitter is a social network that has more than 340 million users [62], and, for this reason, the analysis of the interactions that happen on such a site can have a high relevance. We detect that the replying, retweeting, and quoting interaction networks can be appropriately described through two models:

(i) An SVM model that utilizes the embeddings provided by Node2Vec algorithm and the Hadamard operator.

(ii) An RF model that uses as explanatory variables certain metrics describing the similarities between nodes.

We found the most relevant used hashtags by type of interaction and also found that the cumulative probability distributions of hashtags by community are similar. Globally, the cumulative distributions of replying and retweeting interactions exhibit a different pattern. To gain a better understanding of Twitter interactions on such a relevant issue as climate change, this investigation can be continued in several ways:

(i) An inspection of the evolution of temporal cooperation in the interaction networks can be performed using the conventional evolutionary game theory.

(ii) An analysis of the diffusion mechanisms as well as an examination of the dynamics of opinion formation can be performed. The above will make it possible to study the extent to which an opinion can be manipulated by algorithmic procedures (bots) as well as the effects that the structure of the interaction networks could have on it.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Disclosure

This research was carried out as a result of the Project: Hopper: Women, Society, Technology and Education which was granted in the internal call for research projects in 2021 at Universidad Francisco de Vitoria.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

This work was partially funded by Telefonica Chair at Francisco de Vitoria University. The authors thank Mari Luz Congosto Martínez for her help in training on the utilization of the T-Hoarder tool.

Supplementary Materials

Supplementary Materials include (i) overview of T-Hoarder tool, (ii) description of similarity measures (local, global, and quasi-local methods), and (iii) tables related to modeling of the interaction networks. (*Supplementary Materials*)

References

[1] S. Tabassum, F. Pereira, S. Fernandes, and J. Gama, "Social network analysis: an overview," *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, vol. 8, no. 5, pp. 1–29, 2018.

[2] A. M. Khattak, R. Batool, F. A. Satti, J. Hussain, A. M. Khan, and B. Hayat, "Tweets classification and sentiment analysis for personalized tweets recommendation," *Complexity*, vol. 2020, Article ID 8892552, 11 pages, 2020.

[3] J. F. Sánchez-Rada and C. A. Iglesias, "Social context in sentiment analysis: formal definition, overview of current trends and framework for comparison," *Information Fusion*, vol. 52, 2019.

[4] Y. Lin, C. Weitong, L. Xue, Z. Wanli, and Y. Minghao, "A survey of sentiment analysis in social media," *Knowledge and Information Systems*, vol. 60, 2019.

[5] P. Wang, B. Xu, Y. Wu, and X. Zhou, "Link prediction in social networks: the state-of-the-art," *Science China Information Sciences*, vol. 58, 2014.

[6] M. Bell, S. Perera, M. Piraveenan, M. Bliemer, T. Latty, and C. Reid, "Network growth models: a behavioural basis for attachment proportional to fitness," *Scientific Reports*, vol. 7, Article ID 42431, 2017.

[7] K. Yamasaki, K. Matia, S. V. Buldyrev et al., "Preferential attachment and growth dynamics in complex systems," *Physical review. E, Statistical, nonlinear, and soft matter physics*, vol. 74, Article ID 35103, 2006.

[8] A. L. Barabási and R. Albert, "Emergence of scaling in random networks," *Science (New York, N.Y.)*, vol. 286, no. 5439, pp. 509–512, 1999.

[9] K. Nguyen and D. Tran, *Fitness-Based Generative Models for Power-Law Networks*, 2012.

[10] R. Bauer and M. Kaiser, "Nonlinear growth: an origin of hub organization in complex networks," *Royal Society Open Science*, vol. 4, Article ID 160691, 2017.

[11] A. D. Broido and A. Clauset, "Scale-free networks are rare," *Nature Communications*, vol. 10, p. 1017, 2019.

[12] M. Congosto, P. Basanta-Val, and L. Sánchez-Fernández, "T-Hoarder: a framework to process Twitter data streams," *Journal of Network and Computer Applications*, vol. 83, pp. 28–39, 2017.

[13] G. Van Rossum and F. L. Drake, *Python 3 Reference Manual*, CreateSpace, Scotts Valley, CA, USA, 2009.

[14] R. W.D., "The R Project for Statistical Computing," https://www.r-project.org/.

[15] A. Hagberg, P. Swart, and D. S. Chult, "Exploring network structure, dynamics, andfunction using networkx," *Tech. rep*,

Los Alamos National Lab.(LANL), Los Alamos, NM USA, 2008.
[16] Igraph W D, "Get Start with Igraph," https://igraph.org/r/.
[17] G. Rossetti, L. Milli, and R. Cazabet, "Cdlib: a Python library to extract, compareand evaluate communities from complex networks," *Applied Network Science*, vol. 4, no. 1, pp. 1–26, 2019.
[18] Gephi W D, "The Open Graph Viz Platform," https://gephi.org/.
[19] Study, *Affinity Diagrams: Definition & Examples*, Study.com, study.com/academy/lesson/affinitydiagramsdefinition examples.html, 2016.
[20] M. E. J. Newman, "The structure and function of complex networks," *SIAM Review*, vol. 452 pages, 2003.
[21] E. Estrada, D. Higham, and N. Hatano, "Communicability betweenness in complex networks," *Physica A: Statistical Mechanics and Its Applications*, vol. 3885 pages, 2009.
[22] M. L. Mouronte-López, "Analysing the vulnerability of public transport networks," *Journal of Advanced Transportation*, vol. 2021, Article ID 5513311, 22 pages, 2021.
[23] M. E. Newman, "Mixing patterns in networks," *Physical review. E, Statistical, nonlinear, and soft matter physics*, vol. 67, Article ID 26126, 2 pages, 2003.
[24] M. E. Newman, "Newman. Assortative mixing in networks," *Physical Review Letters*, vol. 89, no. 20, Article ID 208701, 2002.
[25] F. Fouss, A. Pirotte, J.-m. Renders, and M. Saerens, "Random-walk computation of similarities between nodes of a graph with application to collaborative recommendation," *IEEE Transactions on Knowledge and Data Engineering*, vol. 19, no. 3, pp. 355–369, 2007.
[26] A. Grover and J. Leskovec, "node2vec: scalable feature learning for networks," in *Proceedings of the KDD: proceedings. International Conference on Knowledge Discovery & Data Mining*, pp. 855–864, San Francisco, CA, USA, August 2016.
[27] S. Haykin, *Neural Networks: A Comprehensive Foundation*, Prentice Hall, New Jersey, NJ, USA, 2 edition, 1999.
[28] M. L. Mouronte-López, "Modeling the public transport networks: a study of their efficiency," *Complexity*, vol. 2021, Article ID 3280777, 19 pages, 2021.
[29] T. Zhou, L. Lü, and Y.-C. Zhang, "Predicting missing links via local information," *The European Physical Journal B*, vol. 71, no. 4, pp. 623–630, 2009.
[30] E. A. Leicht, P. Holme, and M. E. Newman, "Vertex similarity in networks," *Physical review. E, Statistical, nonlinear, and soft matter physics*, vol. 73, no. 2, Article ID 26120, 2006.
[31] G. Salton and M. J. McGill, *Introduction to Modern Information Retrieval*, McGraw-Hill, New York, NY, USA, 1986.
[32] E. Ravasz, A. L. Somera, D. A. Mongru, Z. N. Oltvai, and A. L. Barabási, "Hierarchical organization of modularity in metabolic networks," *Science (New York, N.Y.)*, vol. 297, no. 5586, pp. 1551–1555, 2002.
[33] P. Jaccard, "Étude comparative de la distribution florale dans une portion des Alpes et des," *Jura"" Bulletin de la Société Vaudoise des Sciences Naturelles*, vol. 37, pp. 547–579, 1901.
[34] T. Sørensen, "A method of establishing groups of equal amplitude in plant sociology based on similarity of species content and its application to analyses of the vegetation on Danish commons," *Biologiske Skrifter/Kongelige Danske Videnskabernes Selskab*, vol. 5, pp. 1–34, 1948.
[35] L. Katz, "A new status index derived from sociometric analysis," *Psychometrika*, vol. 18, no. 1, pp. 39–43, 1953.
[36] P. Yu. Chebotarev and E. V. Shamis, "A matrix-forest theorem and measuring relations in small social group," *Automation and Remote Control*, vol. 58, no. 9, pp. 1505–1514, 1997.
[37] S. Brin and L. Page, "The anatomy of a large-scale hypertextual web search engine," in *Proceedings of the Paper presented at Seventh International World-Wide Web Conference (WWW 1998)*, Brisbane, Australia, April 1998.
[38] I. M. Chakravarti, R. G. Laha, and J. Roy, *Handbook of Methods of Applied Statistics*, vol. I, pp. 392–394, John Wiley & Sons, Hoboken, NJ, USA, 1967.
[39] R. Aldecoa and I. Marín, "Surprise maximization reveals the community structure of complex networks," *Scientific Reports*, vol. 3, no. 1, p. 1060, 2013.
[40] E W D Eurostat, "The European Parliament declares climate emergency," 2019, https://www.europarl.europa.eu/news/en/press-room/20191121IPR67110/the-european- parliament-declares-climate-emergency.
[41] R. Aldecoa and I. Marin, "Deciphering network community structure by surprise," *PLoS One*, vol. 6, 2011.
[42] Patten and M. Newhart, *Understanding Research Methods: An Overview of the Essentials*, Routledge, Oxfordshire, UK, tenth edition, 2017.
[43] M. A. Stephens, "EDF statistics for goodness of fit and some comparisons," *Journal of the American Statistical Association*, vol. 69, no. 347, pp. 730–737, 1974.
[44] L. Breiman, "Random forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001.
[45] S. L. Gortmaker, D. Hosmer, and S. Lemeshow, "Applied logistic regression," *Contemporary Sociology*, vol. 23, 2013.
[46] T. Evgeniou and M. Pontil, "Support vector machines: theory and applications," *Machine Learning and Its Applications*, vol. 2049, pp. 249–257, 2001, https://doi.org/10.1007/3-540-44673-7_12.
[47] X. Xiecs, "273P Machine Learning and Data Mining," 2019, https://www.ics.uci.edu/xhx/courses/CS273P/04-linear-regression-273p.pdf.
[48] M. E. J. Newman, "Communities, modules and large-scale structure in networks," *Nature Physics*, vol. 8, no. 1, pp. 25–31, 2011.
[49] Z. Zhang, P. Pu, D. Han, and M. Tang, "Self-adaptive Louvain algorithm: fast and stable community detection algorithm based on the principle of small probability event," *Physica A: Statistical Mechanics and Its Applications*, vol. 506, pp. 975–986, 2018.
[50] V. A. Traag, L. Waltman, and N. J. Van Eck, "From louvain to leiden," *Guaranteeing Well-Connected Communities" Scientific Reports*, vol. 9, no. 1, pp. 1–12, 2019.
[51] C. Naiyue, L. Yun, C. Jun-Jun, and L. Qing, "A novel parallel community detection scheme based on label propagation," *World Wide Web*, vol. 21, 2018.
[52] S. E. Garza and S. E. Schaeffer, "Community detection with the label propagation algorithm: a survey," *Physica A: Statistical Mechanics and Its Applications*, vol. 534, 2019.
[53] S. Kullback and R. A. Leibler, "On information and sufficiency," *The Annals of Mathematical Statistics*, vol. 22, no. 1, pp. 79–86, 1951.
[54] V. Arnau, S. Mars, and I. Mar n, "Iterative cluster analysis of protein interaction data," *Bioinformatics*, vol. 21, no. 3, pp. 364–378, 2005.
[55] S. Shirazi, H. Baziyad, N. Ahmadi, and A. Albadvi, "A new application of louvain algorithm for identifying disease fields using big data techniques," *Journal of Biostatistics and Epidemiology*, vol. 5, pp. 183–193, 2019.

[56] X. K. Zhang, J. Ren, C. Song, J. Jia, and Q. Zhang, "Label propagation algorithm for community detection based on node importance and label influence," *Physics Letters A*, vol. 381, 2017.
[57] M. E. J. Newman, *Networks: An Introduction*, Oxford University Press, Oxford, UK, 2011.
[58] F. Santo, "Community detection in graphs," *Physics Reports*, vol. 486, 2009.
[59] Y. Kong, G. Y. Shi, R. J. Wu, and Y. C. Zhang, "k -core: theories and applications," *Physics Reports*, vol. 832, 2019.
[60] United Nations W.D Department of Economic, "S Affairs, Sustainable Development. The 17 goals," https://sdgs.un.org/es/goals.
[61] EnergyGov W.D. U. S. Launches, "Net-Zero World Initiative to Accelerate Global Energy System Decarbonization," https://www.energy.gov/articles/us-launches-net-zero-world-initiative-accelerate-global-energy-system-decarbonization.
[62] W D, "Digital in 2020," 2020, https://wearesocial.com/digital-2020.