



EDGly: detection of influential nodes using game theory

Minni Jain¹ · Aman Jaswani¹ · Ankita Mehra¹ · Laqshay Mudgal¹

Received: 24 May 2020 / Revised: 9 February 2021 / Accepted: 17 August 2021 /
Published online: 9 October 2021

© The Author(s), under exclusive licence to Springer Science+Business Media, LLC, part of Springer Nature 2021

Abstract

Identifying those nodes that play a critical role within a network is of great importance. Many applications such as gossip spreading, disease spreading, news dispersion, identifying prominent individuals in a social network, etc. may take advantage of this knowledge in a complex network. The basic concept is generally to identify the nodes with the highest criticality in a network. As a result, the centrality principle has been studied extensively and in great detail, focusing on creating a consistent and accurate location of nodes within a network in terms of their importance. Both single centrality measures and group centrality measures, although, have their certain drawbacks. Other solutions to this problem include the game-theoretic Shapley Value (SV) calculations measuring the effect of a collection of nodes in complex networks via dynamic network data propagation process. Our novel proposed algorithm aims to find the most significant communities in a graph with community structure and then employs the SV-based games to find the most influential node from each community. A Susceptible-Infected-Recovered (SIR) model has been employed to distinctly determine each powerful node's capacity to spread. The results of the SIR simulation have also been used to show the contrast between the spreading capacity of nodes found through our proposed algorithm and that of nodes found using SV-algorithm and centrality measures alone.

Keywords Centrality · Game-theory · Shapely Value (SV) · Susceptible-Infected-Recovered (SIR) model · EdMot · Ensemble Clustering for Graph (ECG)

✉ Minni Jain
minnijain@dtu.ac.in

Aman Jaswani
ajaswani11@gmail.com

Ankita Mehra
mehra.ankita99@gmail.com

Laqshay Mudgal
laqshay.mudgal@gmail.com

¹ Delhi Technological University, Delhi, India

1 Introduction

The identification of influential nodes in complex networks has attracted considerable attention within the research community [36]. Social networking has become a modern tool for people to connect and spread the news with the development of science and technology. Thus, the critical issue of detecting a collection of the most prominent individuals in a network can help monitor the dissemination of social networking messages and rumours, discovering social influencers, etc. Particularly in the case of rumour propagation, as rumours spread quickly and widely and they have great power of destruction. It is therefore of great theoretical and practical importance to decide whether there is an influential spreader and to recognize who is the influential spreader in the process of spreading or controlling the spread of information [10, 14, 20].

Various approaches to the identification of influential nodes in complex networks including single centrality and group centrality measures have been explored in the past [21, 23]. A brief understanding of these concepts is discussed in Sect. 4.2. Other than the basic centrality measures, SV-based centrality algorithms have also been employed for similar problems [1, 22]. This is based on the main concept of a game-theoretic network which means defining a cooperative game across a network where agents are nodes, coalitions are node groups, and coalition payoffs are specified to meet the requirements of a given application.

Although many methods have been proposed, most of them fail to take into account the community structure of graphs while identifying influential nodes. Community structure organizes the nodes into sets such that a dense internal connection exists among the nodes in each set. In large and complex real-world networks, it is rarely the case that a single node could be able to dissipate a piece of information to the entire network. Therefore, we aim at identifying a group of nodes that might be able to scatter information to the rest of the network. Community structures refer to a subset of nodes in which all nodes are densely connected to all the other nodes present within the same community, but loosely connected to all the nodes present in other communities [27]. Higher-order community structure deals with small dense subgraph patterns and also the density of connections between two different communities. The benefit of identifying these is that we are able to find such a set of communities (rather than an individual nodes or just one community), that can dissipate information to majority of the network.

Various approaches have been used in the past for leveraging higher-order community structure such as Harmonic Motif Modularity [12], HBNmining [15], Attribute Homogeneous Motif [16], etc. The major drawback with all of them, however, is that they do not optimally deal with the lower-order topological structure. They do not take into consideration the smaller, isolated, and fragmented components that may be present in a large real-world network, and focus only on the larger connected components. EdMot [17] approach overcomes this by rewiring the network and by taking into account the higher-order community structure instead of only the lower-order one.

Graph clustering aims at partitioning the vertices of a graph into good clusters (based on different objective functions). Yang et al. [34] have studied and compared eight state-of-the-art clustering algorithms. They have also mentioned the various merits and demerits of each of the approaches. Poulin et al. [26] proposes the ECG algorithm which outperforms all the algorithms studied by Yang et al. [34]. The ECG algorithm has been used to detect communities in a graph and is based on Multi-level Louvain algorithm [5]. Furthermore, the ECG algorithm also provides the information about strength between different communities present.

Our proposed algorithm initially aims to employ the EdMot approach that leverages the higher-order characteristics at the level of dense sub-graph patterns to identify the community structure in a graph. We propose an enhancement in this approach by using the Ensemble Clustering for Graph (ECG) algorithm for network partitioning, thus the rewired network is then fed to the ECG algorithm. After this, the most influential node from each partitioned community is identified according to the Shapely centrality value using EDGly algorithm. The ECG algorithm is based on the Louvain algorithm but produces more accurate results as a consequence of the ensembles of clusters used. Thus, we obtain the most significant partitioned communities from the original network and then employ SV based games to find the most significant nodes from each significant community identified. This gives us a list of all the influential nodes in the graph.

We have used the measures of prime centrality, the game-theoretical approach and our proposed algorithm to identify the most prominent top-k nodes and demonstrated a distinct and detailed contrast between all the approaches. We have finally used the SIR model to evaluate the performance of the proposed algorithm in comparison to the centrality-based measures and the SV games. The SIR model simulates an information spreading process in which our most significant nodes, identified via any of the algorithms, are considered as infected and tries to spread the information in the entire network. This helped to portray the different aspects and accuracy of the varied approaches vividly. More details on the working of SIR model and the evaluation criteria are mentioned in Section 7.3.

The main contribution of our work is to detect influential nodes in a graph by incorporating the community structure. EDGly algorithm uses and leverages the EdMot approach—an Edge enhancement approach for Motif-aware community detection, and ECG algorithm—a graph clustering method. The SV of each node is then calculated and the most influential nodes are obtained.

The remainder of this paper is organized as follows. Section 2 gives a detailed study of the various works done in the related field. Section 3 explains the datasets used. Section 4.1 explains the preliminaries while Sect. 6 explains the algorithmic flow. Section 7.1 illustrates the working of the proposed algorithm with the help of a diagram. Section 7.3 describes the evaluations performed and the results obtained. Section 9. concludes the research work with detailed insight into its future scope.

2 Literature review

One of the fundamental research discussions in the literature on network analysis is the topic of connectivity. The first to experiment to detect the primary top-k nodes were Domingos and Richardson [8]. They developed an algorithmic model to address this problem by modelling social media network as Markov random fields which mathematically characterized the probability of occurrence of an event. Chen and Teng [7] explained that single node centrality measures are suitable for assessing individual influence in isolation while Shapley centrality assesses individuals' performance in group influence settings.

Wei et al. [32] explored the need to learn distributed representation of the vectors for each vertex in a network. They laid emphasis on node classification and link prediction. A unique approach to identify influential nodes is to consider the k-shell value of each vertex as the mass and shortest path distance between two vertices as the distance, and then apply the concept of gravity centrality index [20].

We further investigated various kinds of centrality measures used for finding the most influential nodes in a network. Degree Centrality (DC), discussed by Gao et al. [10], is used to efficiently measure the significance of nodes. However, it does not take into consideration the overall, detailed anatomy of the network. Eigen-Vector Centrality (EC), according to Stephenson and Zelen [29], overcomes the defects associated with DC. It takes into account the influence of neighbours of the node in consideration. Betweenness Centrality (BC), as explored by Freeman [9], learns topology-related data of networks in advance. Al-Garadi et al. [2] describes how Closeness Centrality (CC) can be efficiently used to identify more than one prominent spreader.

Previous research by Tan et al. [30] on spreading rumours focused primarily on communities' viral epidemics. The normal (and somewhat standard) model for viral epidemics is called the restored or SIR model that is susceptible-infected-recovered. Various methods have been defined for the same, including weighted k-core decomposition method [2] and rumour centrality with a mass centre technique [30]. Algorithms with low complexity have been devised that can be applied to large scale networks by introducing entropy into a complex network and using the SIR model for evaluation. Such a model [37] can be used for rumour management and targeted advertisements.

We studied the disadvantages associated with using centrality algorithms to find the most significant node in [7, 19, 24]. An attempt has been made to find the most influential node in a network using Mapping Entropy (ME) that reflects the correlation between a node and its neighbours [24]. We particularly inspected the application of ME using ENRON email dataset which is commonly used for the study of social networks [28].

Game-theory is a significant paradigm that finds its applications in various fields. It is used in statistics and business analytics for prototyping the interactivity among participating agents [4]. One of its most prominent applications is finding the most influential node within a network. Typical social network analysis cannot capture the dynamics of strategic interactions among the individuals in the network. Our proposed model is based on cooperative game theory that solves this issue [33]. The elemental constituents of intricate interactivities in a network can be efficiently processed using a rich class of games, called influence games, as has been demonstrated in [13]. Game-theoretic network centrality measures along with their computational complexity and axiomatic properties have been neatly explained in [31].

We studied about Explosion-Trust (ET) Game Model by referring to [18]. It remarkably explains how a rumour spreading model can be constructed using game-theory by considering two very significant factors—rumour explosion degree and trust degree of the source node. In complex networks, dense clusters [14] can be formed by detecting influential nodes using degree centrality and selecting the most influential nodes as the initial spreaders. This ensures that not just the spreader nodes are prominent themselves, but also that the distance among the nodes is relatively scattered. In [36], the probability of transfer of information was used between any pair of nodes and the clustering method, called k-medoid. Such an efficient algorithm helps to classify large scale powerful nodes in complex networks with community structures.

Amati et al. [3] aimed to find the most influential nodes from various Twitter scraped datasets. Primitive centrality measures like degree centrality, closeness centrality, eigen-vector centrality, and page-rank centrality were employed for the same. They also studied how the influence of spreaders evolved over time with the help of Dynamic Retweet Graph (DRG) that focused on how retweets of a tweet were made over time. This helped them better analyze the relationship between users. They concluded that closeness centrality gave

too many nodes with high centrality and was, therefore, not a good indicator. Furthermore, the remaining measures gave almost similar results in terms of the most influential nodes.

3 Data sets

This section gives an elaborate description of the datasets that have been used for our implementation. We have used two datasets which have been described below.

The first dataset is the USAir97 dataset [6] which has been transformed into an undirected network, created by 332 nodes, where one airport represents a node, and 2126 edges, with each edge reflecting a direct air-line between two American airports, if any. Here, weights represent the normalized distance between two airports.

The second dataset is a tiny Facebook network [25] made up of 899 users serving as nodes and having 7089 edges. An edge occurs between two vertices if a few messages have been exchanged between the two users and are weighted by the number of messages exchanged.

3.1 Unweighted graph

An unweighted graph can be technically defined as a graph $G(N, E)$ having ‘ n ’ nodes represented by set N and ‘ e ’ edges represented by set E consisting of unordered pairs, such that $(n_1, n_2) = (n_2, n_1)$ and $(n_1, n_2) \in E$ and $n_1, n_2 \in N$. Games 1 and 2 are played by creating an unweighted network from both the datasets.

3.2 Weighted graph

A weighted graph can be technically defined as a graph $G(N, E)$ having ‘ n ’ nodes represented by set N and ‘ e ’ edges represented by set E consisting of ordered pairs, such that $(n_1, n_2) \neq (n_2, n_1)$ and $(n_1, n_2) \in E$ and $n_1, n_2 \in N$. Games 3–5 are played by creating a weighted network from both the datasets.

4 Preliminaries

Section 4.2 explains the centrality measures used while Sect. 4.3 describes the process used for community detection in detail.

4.1 Centrality measures

We have used certain centrality measures in our implementation to compare the results our proposed algorithm with these already existing measures. For more insight into these centrality measures along with their mathematical derivations, refer to the work of Zhan et al. [35]. The centrality measures used in our study are as follows:

- (i) Eigen-vector Centrality (EC): EC [29] considers the relative power or significance of the nodes. Here, each node is assigned a value representing its relative significance considering the fact that nodes which are connected to high-power nodes have high influence over the network in comparison to those who are connected to low-power nodes.

$$C_e(v_i) = \frac{1}{\lambda} \sum_{j \in M(v_i)} C_e(j) \quad (1)$$

In Eq. (1), $M(v_i)$ is set of neighbours of v_i , and λ is a constant. $C_e(v_i)$ is the measure of EC for vertex v_i .

- (ii) Betweenness Centrality (BC): BC [9] measures how strongly two nodes are connected via a given node. It is estimated as the ratio of the aggregate of shortest distances between any two nodes in the network, on which the node lies, to the shortest path between the two nodes considered.

$$C_b(v_i) = \sum_{j=1, k \neq 1}^N \frac{g_{jik}}{g_{jk}} \quad (2)$$

In Eq. (2), g_{jk} is the total number of shortest paths from v_j to v_k , and g_{jik} is the number of paths that pass through v_i . $C_b(v_i)$ is the measure of betweenness centrality for vertex v_i .

- (iii) Closeness Centrality (CC): CC [2] measures how quickly information can be spread from one node to all the other nodes in a network. It is measured as the inverse of the total sum of all shortest path distances between a given node to all other nodes in a network.

$$C_c(v_i) = \sum_{j=1}^N \frac{1}{d(v_i, v_j)} \quad (3)$$

In Eq. (3), $d(v_i, v_j)$ is the distance between vertices v_i and v_j , and $C_c(v_i)$ is the measure of closeness centrality for vertex v_i .

Single centrality measures suffer from certain disadvantages due to the failure to recognize the effects when considered in groups on node functionality. Group centrality measures place great focus on operating in groups of nodes and not on their individual functionalities. Nonetheless, group centrality also suffers from a drawback by focusing on a-priori-determined node groups and contributing to confusion when prioritizing individual nodes within the network.

4.2 Community detection

The algorithm divides the task of detecting influential nodes into two parts:

- (i) Detecting significant communities in a graph

We used the ECG algorithm to detect the communities present in the graph, primarily because of its efficiency in accuracy and the number of communities found. The latter metric enables us to experiment with the number of communities and see the effect of that on the infected scale in our SIR Model. The ECG algorithm works

with consensus clustering. It has two main steps—generation and integration. The generation process uses the Multilevel Louvain algorithm [5]. The ML algorithm creates a hierarchy of partitions, and ECG uses the randomized partitions at level-1. The number of the randomised partitions used determines the size of the ensemble of clusters. The integration step involves applying ML on the weighted version of a graph, G , where the weight of each edge is given by:

$$W_P(u, v) = \begin{cases} w_* + (1 - w_*) \cdot \left(\frac{\sum_{i=1}^k v_{P_i}(u,v)}{k} \right), & \text{if } (u, v) \text{ is in the 2-core of } G \\ w_*, & \text{otherwise} \end{cases} \tag{4}$$

In Eq. (4), $0 < w_* < 1$ is the minimum ECG weight and $v_{P_i}(u, v) = \sum_{j=1}^{l_i} 1_{C_j^i}(u) \cdot 1_{C_j^i}(v)$ indicates if the vertices u and v co-occur in the same cluster of P_i or not. The k -core of a graph is its maximal subgraph such that every vertex has degree of atleast k . In this case, the 2—core of G is its maximal subgraph such that each vertex has degree of atleast 2.

For this, we proposed an enhanced version of the EdMot algorithm by replacing the Louvain algorithm for network partitioning by the ECG algorithm. The EdMot approach for network rewiring has been used to increase connectivity within connected components in the graph. This approach takes into account the higher-order community structure instead of only lower-order structures. The algorithm aims to unravel peculiar characteristics at the level of network motifs. These are basic interaction patterns in a graph that recur throughout graph networks, much more than in random networks. This is crucial to inculcate while considering the influence of nodes in a graph.

- (ii) Finding the most influential node from each community

For finding the most influential node per community in a graph, we use *The Shapley Value*. On the basis of the concept of marginal (or borderline) contribution, an important solution concept was proposed by Shapely [1, 22].

Player i 's SV, denoted by $SV_i(v)$ is equal to the weighted mean of i 's borderline contributions to each coalition C , to which the player may belong.

$$SV_i(v) = \frac{1}{n!} \sum_{\pi \in \pi(I)} \{v(C_\pi(i) \cup \{i\}) - v(C_\pi(i))\} \tag{5}$$

In Eq. (5), the aggregate count of players is given by ' n ' while $\pi(I)$ gives the set of all permutations with ' n ' players. More insight into the SV and its derivation can be found in [1, 22].

This idea is based on the theory of cooperative games—an element of game theory that allows players to form coalitions to maximize their game yields. Cooperative games theory provides a high-level approach as it describes only the coalitions' structure, strategies and benefits.

4.3 Coalition games

Coalitions are gatherings of players that form the essential or fundamental elements of decision making. These are assumed to uphold cooperative conduct which makes it reasonable to view these games as a contest between alliances of participants and not between separate players. The core assumption here is that as the game proceeds, an eminent alliance or coalition comprising all participants will manifest eventually.

5 Proposed EDGly algorithm for influential nodes detection

Focusing on Game Theory's Shapley algorithm, we referred to the algorithms described in Michalak and Szczepański's work [1, 22]. In both weighted and unweighted networks, the exact analytical formulae for SV-based centrality were established. The SV-based centrality polynomial-time algorithms have been developed.

- Step 1: Creation of weighted and un-weighted network graphs** Graphs are created by using the networkx library in Python for both USAir97 and Facebook datasets. Games 1 and 2 require unweighted graphs whereas the remaining games require weighted graphs. Please refer Sect. 3.
- Step 2: Detection of communities** Connectivity within the network is enhanced by network rewiring as explained in Algorithm 1. After this, communities are detected using the ECG algorithm for the detection of network communities. In line with EDGly algorithm, from each community, we picked the most influential node (as described in Step 3). It was observed that the number of communities identified does not change results significantly. We have shown the seven communities and their corresponding influential nodes.

Algorithm 1: The EDGly approach

Input: Original network $A \in R^{n \times n}$, parameter K

Output: The top influential nodes from A . (One from each community detected.)

1. Construct motif adjacency matrix W_M from A via (2).
 2. Obtain connected components Φ via (5) and select top K largest connected components.
 3. Apply ECG to partition each $\phi_l \in \Phi_K, \forall l \in \{1, \dots, K\}$ and aggregate the results to form the module set $\{M_1, \dots, M_m\}$.
 4. Construct a new edge set E_{mod}^* via (8).
 5. Rewire the original network to obtain G_A^M via (9).
 6. Feed G_A^M into ECG to obtain final community structure \hat{C}
 7. Iterate over every community and detect the most influential node using Shapely Centrality Value.
-

- Step 3: Coalition games based on Shapely algorithm** SV is the average marginal cost contribution across all potential coalitions of a function value.

Step 3: Coalition games based on Shapely algorithm

The Shapley algorithm is applied carefully and tries to find the top-k nodes that might be the most prominent ones.

Specifically, we concentrated on five underlying network-defined coalition games that vary in degree and centrality of the network. Each game has a certain characteristic function $v(C)$ which represents how prominent a particular node is to a given coalition C . For more insight into the working of these games and their underlying mathematics, refer to [1, 22]. The game descriptions are as follows:

- (i) **Game 1:** In this game, we consider all the permutations of all the nodes that are immediately reachable, by one hop to the node $n_i \in N(G)$. Let each random permutation be denoted by P_i , the neighbours of node n_i , in the graph $G(N, E)$ be denoted by $n_i.neighbours$ and the degree of node n_i , be denoted by $n_i.degree$. Algorithm 2 describes the procedure involved in SV calculation.
- (ii) **Game 2:** In many real-life social scenarios, often taking into account nodes that are directly attached to each other is not enough. A rumour source, will more often than not, affects farther nodes.
For the purpose of taking relationships with farther nodes into account, and generalising the game, we introduce a value, p , depicting the number of agents that the node is adjacent to in a coalition. In this game, a node is considered ‘influenced’ if at least p of its neighbours are influenced. We divide the analysis using this game into 2 parts, first, where the degree of the node is less than p and second where the degree is more than p . Algorithm 3 describes the procedure involved in SV calculation.
- (iii) **Game 3.** In this game, we introduce the concept of weighted graph networks. This game is an extension of game 1; it uses the Dijkstra Algorithm to compute the distance between 2 nodes. The cutoff value, d , is the maximum permissible distance of a node from any member in a given coalition. The extended degree is defined as the size of the set of all nodes that are at-most distance ‘ d ’ away from the node n_i . Algorithm 4 describes the procedure involved for SV calculation.
- (iv) **Game 4.** This is a generalization of game 3. Here we work with the assumption that a node closer to a coalition will have a greater effect on it than some other node farther away, even if both nodes satisfy the cut-off criteria as in game 3. For this purpose, we introduce a positive-valued decreasing function $f(x)$. $f(d)$ refers to the function which has a directly proportional effect on SV of the coalition which is ‘ d ’ units away from a node. The marginal contribution of each node n_i through node $n_j \neq n_i$, for each coalition C_i gives SV, as shown in Algorithm 5.
- (v) **Game 5.** This is a generalization of game 2 in case of weighted networks. Here we have defined a cut-off value (n_i) for each $n_i \in N(G)$. For every coalition C , $d(n_j)$,

$C) = \sum_{n_j \in n_i \text{ neighbours}} W(n_i, n_j)$, where $W(n_i, n_j)$ represents the edge's weight between nodes n_i and n_j (0, if no edge exists).

A node n_i marginally contributes node $n_j \in n_i$ neighbours to the value of coalition C_i if and only if $n_j \notin C_i$ and $d(n_j) - W(n_i, n_j) \leq W(C_i, n_j) < d(n_j, C)$. Algorithm 6 describes the procedure for calculating the SVs.

Step 4: Estimating centrality measures After working on the five coalition games, we introduced multiple centrality measures to determine the network's most powerful node with the highest scope or effect. To generate an elaborate comparison, various network centrality measures such as EC, BC, CC, have been used.

6 Illustration with example

A synthetic network is constructed to illustrate the working of our proposed EDGly algorithm, where nodes in communities are denoted with different colours and the edges represent the connection between communities.

A hypergraph, is formed in the first step as shown in Fig. 1. After partitioning the largest connected component into modules in Step 2, two modules can be identified and a new edge set is constructed to derive a clique from each module, A clique is a graph that has an edge connecting every pair of vertices—the new edges are shown as the dashed lines. A rewired network is obtained by combining the current edge set into the original network by reorienting the network in Step 3. Finally, ECG algorithm is used to partition the rewired network into communities.

After the community structure of the graph is obtained via ECG algorithm in Step 4, the game-theoretic Shapely centrality value is used to detect the most influential node from each community. Accumulation of these nodes from all communities in the graph gives the set of most influential nodes of the network.

Table 1 depicts the results of using EDGly on the synthetic dataset. Three communities are detected and further influential nodes are obtained by applying the Shapely centrality using Games 1–5. The three influential nodes obtained from game 1 and 4 are nodes 1, 9 and 12. Similarly, for game 2, the nodes are 3, 5 and 11. For game 3, the nodes are 1, 6 and 12 while for game 5, the nodes are 3, 9 and 11. We get the set of influential nodes by accumulating the most influential node from each community.

7 Evaluations and results

Section 7.2 explains SIR Model [37] used to compare our model with the prominent existing centrality algorithms to identify the network's most powerful node. To evaluate and compare the performance of EDGly, we have used various benchmark network centrality measures such as EC, BC, CC; these centrality measures describe the influence of nodes on the basis of the structure of the network, and are helpful to

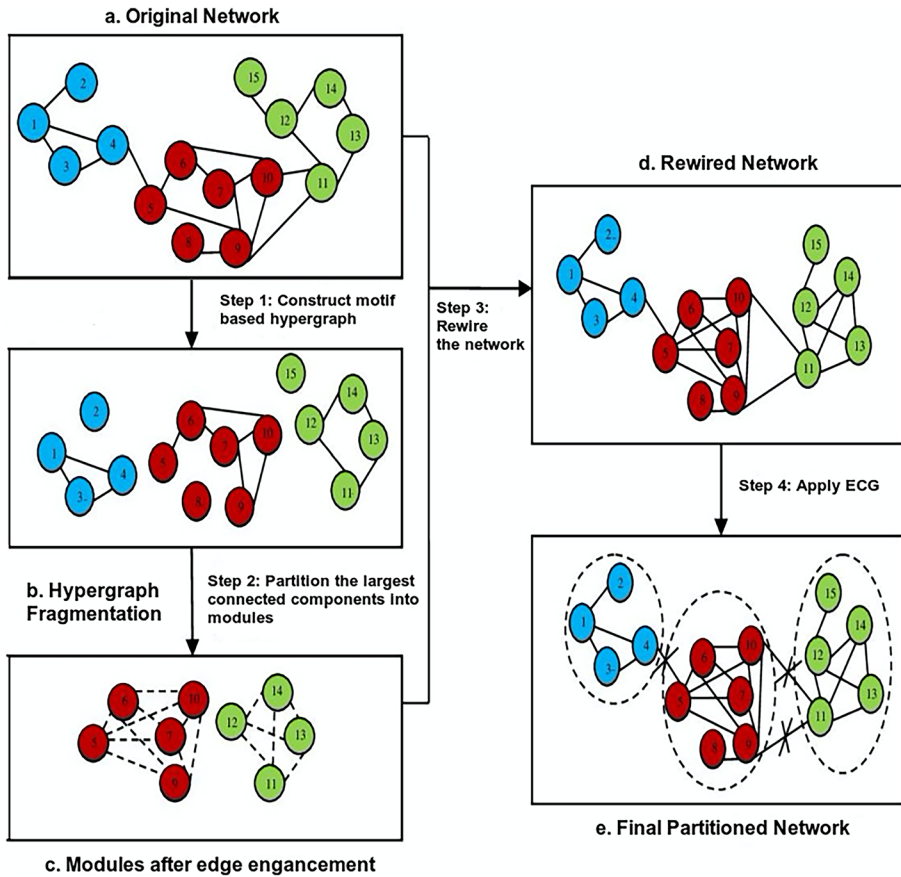


Fig. 1 Algorithmic flow for EDGly

depict an accurate effect of the impact of community structure on the influence of the nodes in a graph. We have further used the SIR model. We have compared the algorithms on the basis of their spreading abilities that can be measured by final infected scale $F(t_c)$ value, that is obtained from SIR simulation and widely used to measure the spreading ability of nodes.

To generate an elaborate comparison, Sect. 7.3 explains the evaluations performed. Sect. 7.1 shows the detailed comparison of results of our proposed EDGly algorithm with those of the Shapely centrality measure and of Shapely centrality with EdMot-Louvain for community detection.

Table 1 Result of EDGly on synthetic dataset

Community	Games				
	G1	G2	G3	G4	G5
1	1	3	1	1	3
2	9	5	6	9	9
3	12	11	12	12	11

7.1 SIR model

According to this model, the dynamics of rumour spreading can be studied. There are three types of nodes in a typical rumour propagation model: (i) susceptible nodes capable of infection, (ii) infected nodes capable of further spreading the rumour, and (iii) recovered nodes that are healed and no longer capable of infection. The most influential nodes identified by any of the algorithms are considered to be the initial sources of rumour propagation. These nodes are marked as infected and all the remaining nodes are marked as susceptible.

We used the USAIR97 dataset [6] to find the most significant airports in an air-travel network. We assumed the initially infected nodes to be most influential airport per graph community. As each edge holds the attribute for the number of flights from one airport to another, we were able to determine the coverage of all locations via the initial influential airport at the end of the simulation.

In the Facebook dataset [25], we aimed to depict a simulation of rumour spreading. The most influential nodes identified by any of the algorithms were considered to be the initial sources of rumour propagation. These nodes were marked as infected and all the remaining nodes were marked as susceptible. At the end of the simulation, we were able to understand the influence of each person predicted by our algorithm. A similar understanding can be used for targeted advertising. The versatility of the model can be gauged by analyzing results on datasets from various domains.

Each iteration is considered to be one time-step, where each of the infected nodes can spread the rumour to its susceptible neighbours with a probability β . Also, each currently infected node becomes recovered with a probability γ , i.e., this node will not spread rumours in the further iterations.

7.2 Evaluations

For the purpose of our experiments, we set $\beta = 0.8$ and $\gamma = 0.2$. The iterations go on till there are no more new infected nodes. The term $R_0 = \beta/\gamma$ is known as the reproduction rate and is extremely crucial to the spreading process. If $R_0 > 1$, then rumour will spread otherwise it will die out soon. Spreading potential is assessed at each time-step 't' by the infected $F(t)$ scale [11]. The value of $F(t)$ is obtained by using the formula given by Eq. (6).

$$F(t) = \frac{n_{i(t)} + n_{r(t)}}{n} \quad (6)$$

Here, $n_{i(t)}$ is the number of infected nodes at time t, $n_{r(t)}$ is the number of recovered nodes at time t, and n is the total number of nodes. Larger the value of $F(t)$, more the influencing capacity of our initial infected set.

After a few iterations, when the rumour spread starts to slow down, we reach a point of saturation where no more new nodes get infected. This is considered to be a stable state and is represented by $F(t_c)$, as shown in Eq. (7).

$$F(t_c) = \frac{n_{r(t_c)}}{n} \quad (7)$$

Table 2 Comparison of infected scale at stable state in various benchmark algorithms for USAIR97 dataset

Method	Initial spreading nodes							$F(t_c)$
	1	2	3	4	5	6	7	
GAME 1								
EC	26	187	182	169	322	147	258	0.421
BC	144	116	46	67	183	112	219	0.370
CC	110	47	197	67	104	322	248	0.337
SHAPELY	182	255	230	201	8	166	67	0.319
SHAPELY	8	65	182	118	201	258	261	0.352
+EdMot								
EDGly	195	71	100	152	261	118	35	0.472
GAME 2								
EC	26	187	182	169	322	147	258	0.334
BC	144	116	46	67	183	112	219	0.391
CC	110	47	197	67	104	322	248	0.397
SHAPELY	166	152	182	230	7	147	112	0.346
SHAPELY	4	74	140	93	312	302	272	0.415
+EdMot								
EDGly	195	130	100	215	119	145	35	0.439
GAME 3								
EC	26	187	182	169	322	147	258	0.373
BC	144	116	46	67	183	112	219	0.352
CC	110	47	197	67	104	322	248	0.312
SHAPELY	201	248	313	67	144	182	261	0.412
SHAPELY	8	59	182	118	313	256	230	0.418
+EdMot								
EDGly	195	130	100	256	67	118	35	0.454
GAME 4								
EC	26	187	182	169	322	147	258	0.388
BC	144	116	46	67	183	112	219	0.388
CC	110	47	197	67	104	322	248	0.406
SHAPELY	152	201	255	230	8	67	166	0.382
SHAPELY	8	65	182	118	201	258	261	0.313
+EdMot								
EDGly	195	71	100	152	261	118	139	0.412
GAME 5								
EC	26	187	182	169	322	147	258	0.445
BC	144	116	46	67	183	112	219	0.430
CC	110	47	197	67	104	322	248	0.382
SHAPELY	248	255	47	293	67	166	233	0.313
SHAPELY	8	65	67	112	313	109	221	0.346
+EdMot								
EDGly	195	71	100	248	261	118	35	0.490

Table 3 Comparison of infected scale at stable state in various benchmark algorithms for Facebook dataset

Method	Initial spreading nodes							$F(t_c)$
	1	2	3	4	5	6	7	
GAME 1								
EC	13	68	59	870	101	186	443	0.634
BC	68	8	54	150	650	173	175	0.636
CC	75	570	150	60	650	29	176	0.620
SHAPELY	75	194	82	12	67	208	8	0.638
SHAPELY	8	59	298	121	164	10	212	0.635
+EdMot								
EDGly	164	7	19	75	10	13	59	0.648
GAME 2								
EC	13	68	59	870	101	186	443	0.599
BC	68	8	54	150	650	173	175	0.615
CC	75	570	150	60	650	29	176	0.641
SHAPELY	164	75	82	12	91	18	19	0.625
SHAPELY	67	13	384	331	405	500	212	0.604
+EdMot								
EDGly	437	600	488	316	873	435	718	0.655
GAME 3								
EC	13	68	59	870	101	186	443	0.622
BC	68	8	54	150	650	173	175	0.599
CC	75	570	150	60	650	29	176	0.610
SHAPELY	13	82	164	18	59	208	75	0.590
SHAPELY	67	2	482	441	164	10	212	0.636
+EdMot								
EDGly	151	117	225	149	453	22	91	0.650
GAME 4								
EC	13	68	59	870	101	186	443	0.586
BC	68	8	54	150	650	173	175	0.627
CC	75	570	150	60	650	29	176	0.645
SHAPELY	75	67	82	12	194	164	208	0.619
SHAPELY + EdMot	67	59	298	121	164	10	212	0.635
EDGly	164	7	19	75	10	13	91	0.656
GAME 5								
EC	13	68	59	870	101	186	443	0.647
BC	68	8	54	150	650	173	175	0.641
CC	75	570	150	60	650	29	176	0.599
SHAPELY	13	18	82	290	59	75	164	0.647
SHAPELY	67	75	107	650	164	247	212	0.604
+EdMot								
EDGly	707	174	19	17	841	13	356	0.666

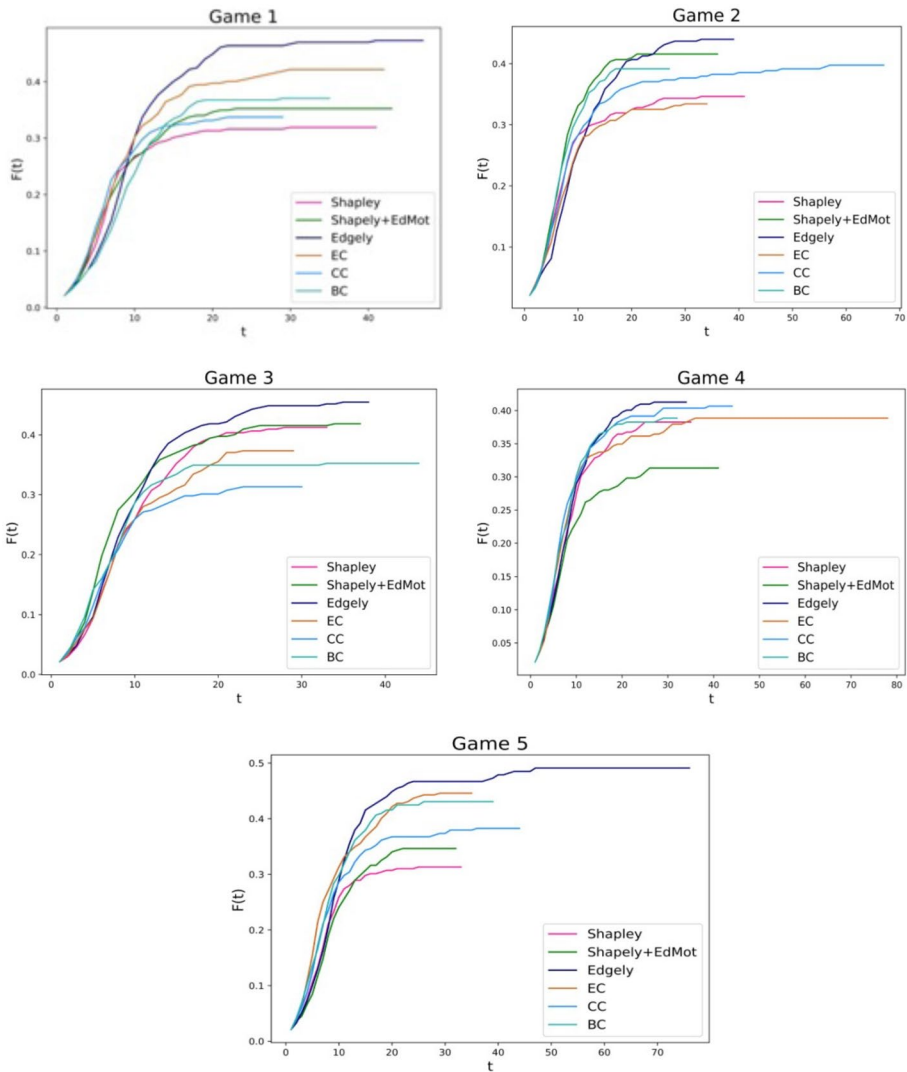


Fig. 2 Spreading speeds and influence of initially infected nodes of the various methods employed for USAIR97 dataset

7.3 Results

Tables 2 and 3 display the $F(t_c)$ results obtained in the SIR simulations for each of the five coalition games. Out of the seven communities identified, the most influential node from each community is identified via the EDGly algorithm. The final infected scale, produced by these spreading nodes are then compared for each algorithm. It can be seen that the

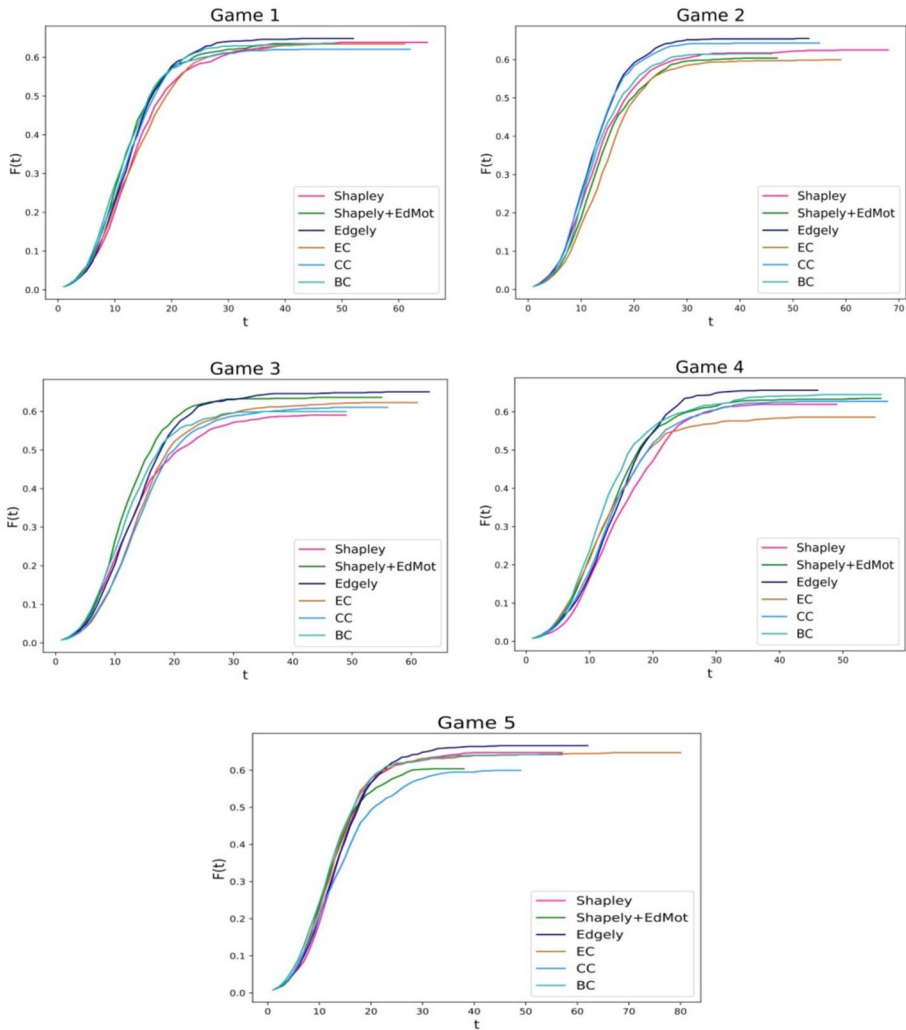


Fig. 3 Spreading speeds and influence of initially infected nodes of the various methods employed for Facebook dataset

nodes identified by the EDGLy algorithm constantly result in a higher final infected scale $F(t_c)$ and hence a better coverage of the network.

It was observed that the change in the number of communities identified in the graph, does not show significant variation in the infected scale, hence we have picked the number of communities as 7.

In Figs. 2 and 3, each graph shows the experimental results of the infected scales by time, $F(t)/t$ received by the SIR Model, on the five different coalition games, using USAIR97 and Facebook datasets respectively. The results are obtained by taking the average of the results of 100 independent runs of SIR simulations. Larger $F(t)$ implies a

larger scale of control in a network with the same spreading time t , which in turn shows a faster spreading rate. It can be seen that the nodes identified by the EDGly algorithm result in a significantly higher final infected scale $F(t_c)$ —depicted by navy blue—and thus, EDGly performs significantly better than the centrality measures EC, BC and CC, marked by colours—brown, light blue and sea green.

Further, it is seen that EDGly also consistently performs better than other algorithms. The addition of community detection using the EdMot—Louvain algorithm results in a significant jump in the infected scale, further the replacement of Louvain, by the ECG algorithm drastically improves the model. Thus, it can be seen from the figures that EDGly spreads faster than other benchmark methods in all games in almost all scenarios and over varied data.

8 Conclusion

We aimed at organizing the nodes into community structures and employing SV-based centrality algorithm for finding the most influential nodes from each community. EdMot approach, in tandem with ECG algorithm, was used to detect communities in a graph because of its unique quality of considering not only lower-order structures but also higher-order community structures. Finally, the SIR model was used to evaluate the capability and scope of spreading ability of each influential node identified during the process.

To evaluate our algorithm on various real-world scenarios, we examined five different game situations, thereby taking into consideration various approaches to determine the most influential nodes in a given dataset. This helped us to gain a deeper and holistic understanding of the game-theoretical algorithm.

It is observed that our proposed algorithm, EDGly algorithm, not only compares well to other commonly-accepted centrality measures, but also drastically improves results by the addition of the task of community detection, to the idea of using coalitional game theory. This is in line with our intuitive prediction, that the application of solely using coalitional game theory on an interconnected network, where each node acts as a player, and maximizes his cumulative yield over all permutations, creates a certain level of restriction in all the possible permutations.

With the application of community detection, we were able to analyze the network on a more individualistic level along with combining the positive effects of coalitions in game theory. This approach thus creates a fine balance between the two approaches and performs better than state of the art approaches.

The model has shown significant success as the most prominent nodes are successfully identified for both the datasets used. We plan to extend the idea of finding the most powerful node in social networks to numerous other similar applications for future work, such as the Internet, or urban networks, and involving a given node in disease dynamics. This will help us understand our algorithm's efficiency and accuracy in multiple applications in the real world.

Further, various optimisation techniques on the SV algorithm, for example, Fuzzy Logic will be implemented for mining much larger social networks and to improve accuracy and other relevant metrics of the project. Fuzzy-based implementation will solve various complexities and limitations that we're currently encountering.

Appendix

Algorithm 2: SVs for Game 1

Input: An unweighted graph $G(N, E)$

Output: SVs of all nodes in G

Initialise: $\forall n_i \in N(G)$ set $SV[n_i] \leftarrow 0$

```

for each  $n_i \in N(G)$  do
     $SV[n_i] \leftarrow 1/(1 + n_i.degree)$ 
    for each  $u_i \in n_i.neighbours$  do
         $SV[n_i] \leftarrow SV[n_i] + \frac{1}{(1 + u_i.degree)}$ 
    end for
end for
return SV

```

Algorithm 3: SVs for Game 2

Input: An unweighted graph $G(N, E)$ and a positive integer p

Output: SVs of all nodes in G

Initialise: $\forall n_i \in N(G)$ set $SV[n_i] \leftarrow 0$

```

for each  $n_i \in N(G)$  do
     $SV[n_i] \leftarrow \min(1, \frac{p}{(1 + n_i.degree)})$ 
    for each  $u_i \in n_i.neighbours$  do
         $SV[n_i] \leftarrow SV[n_i] + \max(0, \frac{u_i.degree - p + 1}{u_i.degree * (1 + u_i.degree)})$ 
    end for
end for
return SV

```

Algorithm 4: SVs for Game 3

Input: A weighted graph $G(N, E, W)$ and a positive cut-off value d

Output: SVs of all nodes in G

Initialise: $\forall n_i \in N(G)$ set $SV[n_i] \leftarrow 0$

```

for each  $n_i \in N(G)$  do
    Distance_Vector  $D \leftarrow \text{Dijkstra}(n_i)$ 
    extended_neighbours  $\leftarrow$  empty 2D array
    extended_degree[ $n_i$ ]  $\leftarrow 0$ 

    for each  $u_i \in N(G)$  such that  $u_i \neq n_i$  do
        if  $D[u_i] \leq d$  then
            extended_neighbours[ $n_i$ ].add( $u_i$ )
            extended_degree[ $n_i$ ]++
        end if
    end for
end for

for each  $n_i \in N(G)$  do
     $SV[n_i] \leftarrow \frac{1}{1 + \text{extended\_degree}[n_i]}$ 

    for each  $u_i \in \text{extended\_neighbours}[n_i]$  do


---


         $SV[n_i] \leftarrow SV[n_i] + \frac{1}{(1 + \text{extended\_degree}[u_i])}$ 
    end for
end for
return  $SV$ 

```

Algorithm 5: SVs for Game 4**Input:** A weighted graph $G(N,E,W)$ and function $f: \mathbb{R}^+ \rightarrow \mathbb{R}^+$ **Output:** SVs of all nodes in G **Initialise:** $\forall n_i \in N(G)$ set $SV[n_i] \leftarrow 0$ **for each** $n_i \in N(G)$ **do** [Distance D , Nodes w] \leftarrow Dijkstra(n_i) sum $\leftarrow 0$, index $\leftarrow |N|-1$, prev_dist $\leftarrow -1$, prevSV $\leftarrow -1$ **while** index > 0 **do** **if** $D(\text{index}) == \text{prev_dist}$ **then**

— currSV = prevSV

else — currSV = $\frac{f(D(\text{index}))}{1+\text{index}}$ - sum **end if** SV[w(index)] \leftarrow currSV +

SV[w(index)]

 sum \leftarrow sum + $\frac{f(D(\text{index}))}{1+\text{index}}$ prev_dist = $D(\text{index})$, prevSV = currSV index \leftarrow index - 1 **end while** SV[n_i] \leftarrow SV[n_i] + $f(0)$ - sum**end for****return** SV

Algorithm 6: SVs for Game 5

Input: A weighted graph $G(N, E, W)$ and cut-offs $W_{cutoff}(n_i)$ for each $n_i \in N(G)$

Output: SVs of all nodes in G

Initialise: $\forall n_i \in N(G)$ set $SV[n_i] \leftarrow 0$

for each $n_i \in N(G)$ **do**

 compute and store α_i and β_i

end for

for each $n_i \in N(G)$ **do**

for each m in 0 to $n_i.degree$ **do**

 compute $\mu \leftarrow \mu(X_m^{n_i}), \sigma \leftarrow \sigma(X_m^{n_i})$

 compute $p \leftarrow \Pr \{ \mathcal{N}(\mu, \sigma^2) < W_{cutoff}(n_i) \}$

$SV[n_i] \leftarrow SV[n_i] + \frac{p}{1 + n_i.degree}$

end for

for each $v_j \in n_i.neighbours$ **do**

$p \leftarrow 0$

for each m in 0 to $n_i.degree$ **do**

 compute $\mu \leftarrow \mu(X_m^{n_i}), \sigma \leftarrow \sigma(X_m^{n_i})$

 compute $z \leftarrow Z_m^{n_i}$

$p \leftarrow p + \frac{z * (v_i.degree - m)}{v_j.degree * (v_j.degree + 1)}$

end for

$SV[n_i] \leftarrow SV[n_i] + p$

end for

end for return SV

References

1. Aadithya KV, Ravindran B, Michalak TP, Jennings NR (2010) Efficient computation of the shapley value for centrality in networks. In: Saberi A (ed) Internet and Network Economics, WINE, Lecture Notes in Computer Science, vol 6484
2. Al-Garadi MA, Varathan KD, Ravana SD (2017) Identification of influential spreaders in online social networks using interaction weighted k-core decomposition method. Phys A 468:278–288
3. Amati G, Angelini S, Gambosi G, Rossi G, Vocca P (2019) Influential users in Twitter: detection and evolution analysis. Multimed Tools Appl 78:3395–3407
4. Bhuiyan BA (2018) An overview of game theory and some applications. Philos Progress 59(1–2):111–128
5. Blondel VD, Guillaume JL, Lambiotte R, Lefebvre E (2008) Fast unfolding of communities in large networks. J Stat Mech. <https://doi.org/10.1088/1742-5468/2008/10/P10008>
6. Bureau of Transportation Statistics. <https://transtats.bts.gov/>
7. Chen W, Teng SH (2017) Interplay between social influence and network centrality: a comparative study on shapley centrality and single-node-influence centrality. In: WWW '17: Proceedings of the 26th international conference on World Wide Web, pp 967–976
8. Domingos P, Richardson M (2001) Mining the network value of customers. In: KDD '01: Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and Data mining, pp 57–66

9. Freeman LC (1978–1979) Centrality in social networks conceptual clarification. *Soc Netw* 1(3):215–239
10. Gao S, Ma J, Chen Z, Wang G, Xing C (2014) Ranking the spreading ability of nodes in complex networks based on local structure. *Phys A* 403:130–147
11. Guo C, Chen X, Chen D, Gao H, Yang L, Ma J (2020) Influential nodes identification in complex networks via information entropy. *Entropy* 22:242
12. Huang L, Wang C, Chao H (2018) A harmonic motif modularity approach for multi-layer network community detection. In: *IEEE International Conference on Data Mining (ICDM)*, pp 1043–1048
13. Irfan MT, Ortiz LE (2011) A game theoretic approach to influence in networks. In: *Proceedings of the twenty-fifth AAAI conference on artificial intelligence (AAAI'11)*, AAAI Press, pp 688–694
14. Jiang L, Zhao X, Ge B, Xiao W, Ruan Y (2019) An efficient algorithm for mining a set of influential spreaders in complex networks. *Phys A* 516:58–65
15. Li PZ, Cai YX, Wang CD, Liang MJ, Zheng YQ (2019) Higher-order brain network analysis for auditory disease. *Neural Process Lett* 49:1–19
16. Li PZ, Huang L, Wang CD, Huang D, Lai JH (2018) Community detection using attribute homogeneous motif. *IEEE Access* 6:47707–47716
17. Li PZ, Huang L, Wang CD, Lai JH (2019) EdMot: An edge enhancement approach for motif-aware community detection. In: *Proceedings of the 25th ACM SIGKDD international conference on knowledge discovery and data mining (KDD '19)*, Association for Computer Machinery, New York, pp 479–487
18. Liu F, Li M (2019) A game theory-based network spreading model: based on game experiments. *Int J Mach Learn Cybern* 10:1449–1457
19. Liu HL, Ma C, Xiang BB, Tang M, Zhang HF (2018) Identifying multiple influential spreaders based on generalized closeness centrality. *Phys A* 492:2237–2248
20. Li Ma, Ma C, Zhang HF, Wang BH (2016) Identifying influential spreaders in complex networks based on gravity formula. *Phys A* 451:205–212
21. Mao C, Xiao W (2018) A comprehensive algorithm for evaluating node influences in social networks based on preference analysis and random walk. *Complexity* 40:1–16
22. Michalak TP, Aadithya KV, Ravindran B, Jennings NR, Szczepanski PL (2013) Efficient computation of the shapley value for game-theoretic network centrality. *J Artif Int Res* 46(1):607–650
23. Ni C, Sugimoto CR, Jiang J (2013) Degree, closeness, and betweenness: application of group centrality measurements to explore macro-disciplinary evolution diachronically
24. Nie T, Guo Z, Zhao K, Lu ZM (2016) Using mapping entropy to identify node centrality in complex networks. *Phys A* 453(2016):290–297
25. Opsahl T (2013) Triadic closure in two-mode networks: redefining the global and local clustering coefficients. *Soc Netw* 35(2):159–167. <https://doi.org/10.1016/j.socnet.2011.07.001>
26. Poulin V, Théberge F (2018) Ensemble clustering for graphs. arXiv:1809.05578
27. Radicchi F, Castellano C, Cecconi F, Loreto V, Parisi D (2004) Defining and identifying communities in networks. *PNAS* 101(9):2658–2663
28. Shetty J, Adibi J (2005) Discovering important nodes through graph entropy: the case of Enron email database. In: *LinkKDD '05: Proceedings of the 3rd international workshop on Link discovery*, pp 74–81
29. Stephenson K, Zelen M (1989) Rethinking centrality: methods and examples. *Soc Netw* 11(1):1–37
30. Tan CW, Yu PD, Zheng L, Lai CK, Zhang W, Fu HL (2016) Optimal detection of influential spreaders in online social networks. In: *Proceeding of annual conference on information sciences and systems*, pp 145–150
31. Tarkowski MK, Michalak TP, Rahwan T, Wooldridge M (2017) Game-theoretic network centrality: a review. arXiv preprint arXiv:1801.00218
32. Wei H, Pan Z, Hu G, Zhang L, Yang H, Li X, Zhou X (2018) Identifying influential nodes based on network representation learning in complex networks. *PLoS ONE* 13(7):e0200091
33. Yadati N, Narayanam R (2011) Game theoretic models for social network analysis. In: *Proceedings of the 20th international conference companion on World Wide Web—WWW' 11*, pp 291–292
34. Yang Z, Algesheimer R, Tessone CJ (2016) A comparative analysis of community detection algorithms on artificial networks. *Sci Rep* 6:30750
35. Zhan J, Gurung S, Parsa SPK (2017) Identification of top-k nodes in large networks using Katz centrality. *J Big Data* 4:16
36. Zhang X, Zhu J, Wang Q, Zhao H (2013) Identifying influential nodes in complex network with community structures. *Knowl-Based Syst* 42:74–84
37. Zhao L, Cui H, Qiu X, Wang X, Wang J (2013) SIR Rumour spreading model in the new media age. *Phys A* 392(4):995–1003

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Minni Jain obtained her M. Tech. degree in Information Security and B. Tech. degree in Information Technology. She is currently working as an Assistant Professor in Delhi Technological University, Delhi, India. Her major research interests include Natural Language Processing, Sentiment Analysis, Information Security, Neural Networks and Fuzzy Logic

Aman Jaswani obtained his B. Tech. degree in Software Engineering from Delhi Technological University, Delhi, India. He has worked on numerous projects in the field of Machine Learning and Natural Language Processing. His areas of research include Game Theory, Machine Learning, Natural Language Processing and Data Science

Ankita Mehra obtained her B. Tech. degree in Software Engineering from Delhi Technological University, Delhi, India. She is a final year student. Her areas of research include Game Theory, Machine Learning, Deep Learning, Computer Vision and Natural Language Processing

Laqshay Mudgal obtained his B. Tech. degree in Software Engineering from Delhi Technological University, Delhi, India. His major research interests include Artificial Intelligence, Machine Learning, Deep Learning, Natural Language Processing, and Data Science