Hindawi
Complexity
Volume 2020, Article ID 7096230, 16 pages
https://doi.org/10.1155/2020/7096230

Research Article

Constructing Real-Life Benchmarks for Community Detection by Rewiring Edges

Jing Xiao, Hong-Fei Ren, and Xiao-Ke Xu

College of Information and Communication Engineering, Dalian Minzu University, Dalian 116600, China

Correspondence should be addressed to Xiao-Ke Xu; xuxiaoke@foxmail.com

Received 19 September 2019; Revised 31 January 2020; Accepted 9 March 2020; Published 7 April 2020

Academic Editor: Ana Meštrović

In order to make the performance evaluation of community detection algorithms more accurate and deepen our analysis of community structures and functional characteristics of real-life networks, a new benchmark constructing method is designed from the perspective of directly rewiring edges in a real-life network instead of building a model. Based on the method, two kinds of novel benchmarks with special functions are proposed. The first kind can accurately approximate the microscale and mesoscale structural characteristics of the original network, providing ideal proxies for real-life networks and helping to realize performance analysis of community detection algorithms when a real network varies characteristics at multiple scales. The second kind is able to independently vary the community intensity in each generated benchmark and make the robustness evaluation of community detection algorithms more accurate. Experimental results prove the effectiveness and superiority of our proposed method. It enables more real-life networks to be used to construct benchmarks and helps to deepen our analysis of community structures and functional characteristics of real-life networks.

1. Introduction

Community structure is one of the most important characteristics of complex networks, which has been commonly found in social networks, biological networks, traffic networks, etc [1–3]. Community detection provides an effective tool to gain insights into the nontrivial internal organization of networks, allowing us to unearth in-depth network information that may not be obtained from direct observation [1–3], e.g., functional and dynamic characteristics [2, 3]. In recent years, we have experienced an increasing demand for detecting communities from a wide range of science domains, such as online targeted advertising [4], social crisis response [5, 6], disease prevention, and medical diagnosis [7].

Generally speaking, an accurate detection of community structures is the basis of analysing the deep information hidden in real-life networks, so the high-precision of community detection algorithms is vitally important. In the past decades, a large number of community detection algorithms based on different knowledge backgrounds have been proposed to improve the accuracy and stability of detection results [1–3, 8–13]. However, the problem remains in our opinion severely limited for two reasons. First, there are no universal protocols on the fundamental ingredients, such as the accurately mathematical definition of community structures [1–3, 14, 15]. Second, there are not enough benchmark networks available with high-precision and diverse functional features, and as a result the crucial issue of estimating the performance of algorithms is still open [14, 15].

Benchmark networks have a significant influence on evaluating the quality of community detection algorithms, and it is also a meaningful tool for us to deeply analyse community structural characteristics of real-life networks. In the existing community detection studies, both real-life and synthetic networks are widely used as standard benchmarks [1, 3, 9]. Real-life networks with precisely known community structures (i.e., the ground truth) are considered to be ideal benchmarks for algorithm tests [1, 3, 9]. However, unfortunately, owing to the actual community structure is often difficult to be obtained and

many features (e.g., degree distribution, transitivity, and size of communities) are not possible to be controlled, the number of real-life networks used as benchmarks in scientific research studies is very limited [3, 15–20]. Therefore, most of the performance evaluation for detection algorithms is conducted on synthetic benchmarks [3, 15–20], among which the built-in community structures are determined during the process of network construction and can be adjusted according to different performance evaluation requirements.

Currently, the most commonly used artificial synthetic benchmarks mainly include the Girvan–Newman (GN) [21] and Lancichinetti–Fortunato–Radicchi (LFR) [14]. GN is generally considered not to be a good proxy for real-life networks because of the same degree of all nodes and the same size of all communities [1, 3, 14, 20, 21]. The abovementioned problems have been addressed by LFR, which is constructed based on a configuration model and characterized by a heterogeneous distribution of node degree and community size [1, 3, 14, 20, 21]. In recent years, several variants of the standard LFR have been introduced to construct more complex and realistic benchmarks [15, 22–27]. Specially, Lancichinetti and Fortunato designed Hierarchical LFR benchmarks (HLFR) for directed and weighted networks with overlapping communities [15]. Orman and Labatut proposed a modified LFR network based on a preferential attachment model, which considers more realistic properties in real-life networks such as degree correlation and transitivity [23]. Elhadi and Agam provided a new LFR network that can be used to analyse and assess the performance of structures and attributes clustering methods [24]. In [25], a new kind of hierarchical LFR network (RB-LFR) was developed by Yang et al. to generate hierarchical community structures that appear in social, biological, and technical systems. Le et al. devised a Generalized LFR (GLFR) network with heterogeneous community mixing fractions, avoiding a fixed fraction of intercommunity links for every community in the same network [26]. Muscoloni and Cannistraci proposed a nonuniform Popularity-Similarity-Optimization model (nPSO) to generate realistic complex networks with communities in the hyperbolic space [27].

Though many novel artificial synthetic benchmarks have been devised, it is still hard for them to satisfy the increasing demand of algorithm performance evaluation in terms of accuracy and functionality. First, there are still great differences between available artificial benchmarks and real-life networks in structural characteristics, which may seriously affect the performance of algorithms. Taking LFR benchmarks for example, during the process of network construction, structural characteristic parameters are set according to some fixed priority [14]. Therefore, each parameter cannot be adjusted independently, and low-priority microscale characteristics are often overlooked or sacrificed. The results of algorithm evaluation obtained on these benchmarks may not provide us accurate and reliable references for studying real-life networks [1, 3]. Second, the existing real-life benchmarks cannot be directly used to estimate the robustness of an algorithm. As the number of each real-life network is only one and the community intensity within the network cannot be adjusted, it is impossible to know how the detection accuracy of the algorithm changes when the network community structure gradually blurs. Therefore, in the existing research studies, the robustness evaluation of algorithms still needs to rely on synthetic benchmarks, such as GN and LFR, which can be constructed as a series of networks with gradually varied community intensity [12, 13, 28–31]. However, the modification of mesoscale structures (i.e., community) in existing synthetic benchmarks, often changes with microscale characteristics, which affects the accuracy of robustness assessment.

To address the issue, in this study we design a new benchmark construction method by the framework of directly rewiring edges of real-life networks, which has not received much attention in studies of community detection. Based on the rewiring-edge method, two kinds of novel benchmarks with special functions are proposed as follows.

First, a set of synthetic benchmarks can be constructed based on a single real-life network of any kind, which can accurately and stably approximate the microscale and mesoscale structural characteristics of the original network. The new benchmarks provide desired proxies for real-life networks and thus making the evaluation of community detection algorithms more accurate. Multiscale structure characteristic analysis of a real-life network can also be realized by employing random edge rewiring null models with different orders in the construction of benchmarks.

Second, a series of synthetic benchmarks with gradually varied community intensity can be constructed based on a single real-life network. Specially, the community intensity in each benchmark can be adjusted independently without interfering with microscale characteristics of the original real network. The new benchmarks enable more real-life networks to be adopted to evaluate the robustness of community detection algorithms with higher precision.

From the abovementioned research studies, more kinds of benchmarks with higher accuracy and functional diversity can be obtained, providing us more accurate and reliable test criteria for evaluating community detection algorithms. In addition, the proposed benchmarks also help to deepen our understanding and analysis of the structure and functional characteristics of real-life networks.

2. The Proposed Benchmarks by Rewiring Edges of a Real-Life Network

2.1. Benchmarks that Accurately Approximate a Real-Life Network. In experimental tests of community detection algorithms based on artificial synthetic benchmarks, because the structure does not accurately reflect the real properties of nodes and communities found in real-life networks [14, 15, 23–27], the accuracy and reliability of test results are affected. In order to obtain benchmarks with structural characteristics closer to reality, a new class of benchmarks is designed by directly rewiring edges of the real-life network according to a constrained rewiring algorithm, which can

accurately approximate the original network in terms of micro- and mesoscale characteristics simultaneously.

The random edge rewiring null model used in the benchmark construction refers to the randomized synthetic network generated based on the original real-life network, by disconnecting all the feasible edges in the network and reconnecting new edges randomly. The null model has properties P_d (termed dK-series), where $d = 0, 1 \ldots, 4$ of P_d corresponds to $k = 0, 1, \ldots, 4$ of dK-series [32–34]. The null model with different orders is able to maintain structural characteristics of the original real-life network at different levels. In addition, null models with different orders are interrelated, i.e., $0k \supseteq 1k \supseteq 2k \cdots (n-1)k \supseteq nk$, and any nk null model contains the characteristics of $(n-1)k$ null model. Specifically, $1k$ null model ensures that the degree distribution of nodes remain unchanged; $2k$ null model maintains the joint degree distribution of nodes; $3k$ null model retains the same clustering coefficient of all the nodes [32–34].

Constructing this class of real-life benchmarks proceeds through the steps shown in Algorithm 1. For a real network G_{origin}, if it has n nodes and m edges, we assume that the number of edges in the original real-life network that successfully performed the rewiring operation is m_{rewired}, and the current and maximum number of executions of the random edge rewiring operation are n_{try} and n_{maxtry}, respectively. The value of n_{maxtry} is set to be $5m$. Let δ be the threshold value of significant errors between two different community divisions. After parameter settings, in Step 2, the community partition of G_{origin} is detected by a typical and efficient community detection algorithm (e.g., CNM [35]) and will be utilized in the following Steps 3 and 4. Then, in Step 3.1, based on the initial community partition, the random rewiring process for all the edges of G_{origin} is operated according to the random rewiring null model with a specific order ranging from $1k$ to $3k$ [32–34], and then a random rewiring network G_{rewired} can be obtained. At the same time, the corresponding microscale characteristic of the original real-life network G_{origin} can be preserved in G_{rewired}. The operation of edge rewiring stops when the number of successfully rewired edges m_{rewired} reaches $2m$ (m is the number of edges in the original network). Experimental experience has shown that when the number of rewired edges G_{rewired} reaches $2m$, the main structural characteristics of G_{rewired} no longer changes strongly as the number of rewired edges increases, indicating that the original network has been sufficiently randomized. However, because each edge rewiring operation may fail, if m_{rewired} cannot reach $2m$, in order to ensure that the network structure of the generated benchmark is random enough, the edge rewiring operation will be carried out until the total number of executions reaches $5m$, i.e., $n_{\text{try}} = n_{\text{maxtry}}$. After the edge rewiring operation in Step 3.1, based on the initial community partition, we recalculate the modularity Q and the fuzzy coefficient μ (i.e., the ratio of the intercommunity edges and total edges) of the G_{rewired} in Step 3.2. In Step 4, repeat the same random edge rewiring operation in Step 3.1 to adjust the mesoscale structural characteristics of G_{rewired}. Since the community partition remains unchanged, the mesoscale structural characteristic values of G_{rewired} will fluctuate slightly up and down [33]. By utilizing this fluctuation property, we can get a modified G_{rewired} that is as close as possible to the G_{origin} in mesoscale structure characteristics. The adjustment operation stops if the accuracy error of G_{rewired} and G_{origin} in terms of community intensity is less than or equal to the threshold value δ (i.e., $|Q(G_{\text{rewired}}) - Q(G_{\text{origin}})| \leq \delta$ or $|\mu(G_{\text{rewired}}) - \mu(G_{\text{origin}})| \leq \delta$). If the abovementioned condition cannot be met, the modified G_{rewired} with Q and μ values closest to the G_{origin} will be selected as the output. Finally, in Step 5, the network G_{rewired} after adjustment is the output as the final generated benchmark $G_{\text{benchmark}}$, which accurately approximates G_{origin} in terms of mesoscale characteristic as well as the microscale characteristic with a specific order.

About the benchmark constructing method in Algorithm 1, there are three points that need further explanation as follows. (1) In the random edge rewiring operation, an intracommunity edge should still be located in the same community after rewiring, while an intercommunity edge should be located between the original two communities after rewiring. For example, if two intracommunity edges l_{intra1} and l_{intra2} are selected, both of them are disconnected and another two new intracommunity edges l'_{intra1} and l'_{intra2} are constructed within the same community. However, if the two selected edges are intercommunity edges l_{inter1} and l_{inter2}, another two new intercommunity edges l'_{inter1} and l'_{intra2} will be connected between the same two communities. (2) Any reconnected edges should meet the specific microscale characteristic restricted by the rewiring operations. For example, if $1k$ null model is selected to construct the benchmarks, all the reconnected edges should not destroy the degree distribution of the original network. If the two newly generated edges cannot meet the condition, they will not be retained in the benchmark and the current rewiring operation is considered a failure. Then, the algorithm will try to randomly rewire another two edges which meets the constraint until the termination conditions are satisfied. (3) There is no strict requirement for the selection of community detection algorithm in Algorithm 1. From the perspective of preserving the mesoscale characteristic of the original real-life network, the selection of different community detection algorithms has little impact on the performance of the generated benchmarks. Any kinds of community detection algorithms with relatively higher performance on the target real-life network can be selected. However, in order to make the community partition represent the ground truth of the original real-life network as accurately as possible and thus ensuring the quality of the generated benchmarks, it is necessary to choose the community detection algorithm with relatively good performance on the target real-life network. Moreover, when the precisions of several algorithms are almost the same, the algorithm with low time complexity is preferred to speed up the benchmark construction.

The computational complexity of Algorithm 1 mainly depends on Steps 2-3. The time complexity of the parameter settings in Step 1 can be negligible. In addition, the complexity of Step 4 should be negligible compared to Steps 2-3 because there are usually very few edges to adjust, especially

Input: A real-life network G_{origin}
Output: Synthetic benchmark $G_{benchmark}$
(1) **Parameter Settings**
Set $m_{rewired}$ to represent the current total number of edges that are successfully rewired. Set n_{try} and n_{maxtry} ($n_{maxtry} = 5m$) to represent the current execution times and maximum execution times of the random edge rewiring operation respectively. The threshold value of significant error between community structures is set to be δ.
(2) Discover the community partition of the original real-life network G_{origin}.
(3) **Initial Benchmark Construction**
3.1 Random edge rewiring operation
while $m_{rewired} < 2m$ **and** $n_{try} < n_{maxtry}$ **do**
Randomly select two edges l_{intra1} and l_{intra2} within a random community of G_{origin} or two edges l_{inter1} and l_{inter2} between two random communities of G_{origin}.
$n_{try} = n_{try} + 1$
If l_{intra1} and l_{intra2} are selected **do**
Disconnect edges l_{intra1} and l_{intra2}, and construct two new intracommunity edges l'_{intra1} and l'_{intra2} that meet the topological characteristic requirements of the selected $1k$–$3k$ null model within the same community.
If the above operation is successful **then**
$m_{rewired} = m_{rewired} + 2$
else
continue
end if
else if l_{inter1} and l_{inter2} are selected **do**
Disconnect edges l_{inter1} and l_{inter2}, and construct two new intercommunity edges l'_{inter1} and l'_{inter2} that meet the topological characteristic requirements of the selected $1k$–$3k$ null model between the same two communities.
If the above operation is successful **then**
$m_{rewired} = m_{rewired} + 2$
else
continue
end if
end if
end while
The random rewiring network $G_{rewired}$ is generated.
3.2 Mesoscale Structure Characteristic Calculation
Calculate the modularity Q and the fuzzy coefficient μ of the $G_{rewired}$ obtained in Step 3.1.
(4) **Microadjustment of Mesoscale Structure Characteristics**
while$|Q(G_{rewired}) - Q(G_{origin})| > \delta$**and**$|\mu(G_{rewired}) - \mu(G_{origin})| > \delta$**do**
Repeat the random edge rewiring operation in Step 3.1 by using the same order null model.
Adjust the network structure of $G_{rewired}$ slightly to approximate the mesoscale characteristic of G_{origin} as accurately as possible.
end while
(5) Output the $G_{rewired}$ as the generated benchmark $G_{benchmark}$.

Algorithm 1: Constructing the benchmarks that accurately approximate a real-life network.

in the $3k$ benchmarks. In Step 2, the computational complexity is closely related to the community detection algorithm adopted, such as $O(n\log^2 n)$ for CNM [35]. Step 3 consists of two substeps with a different time complexity. In the first substep of random edge rewiring, because the microscale structural characteristics calculation involves only a small number of local nodes around the newly reconnected edges, the complexity can be roughly estimated as $O(m)$. In the second substep, the mesoscale structural characteristics can be calculated in time $O(n+m)$ [1]. Therefore, the total time complexity of Algorithm 1 can be estimated as $O(n\log^2 n) + O(m) + O(n+m)$, which can be further simplified to $O(n\log^2 n) + O(n+m)$.

The performance of the abovementioned benchmark is tested. Four real-life networks with different types and sizes are employed as the original real-life network, respectively, including a social network (i.e., Karate [36]), a biological network (i.e., an anatomical connection network of the macaque cortex [37, 38]), an engineering network (i.e., PowerGrid [39]), and a communication network (i.e., PGP [40]). Among the classical datasets, PowerGrid and PGP are relatively large, where the PowerGrid network has 4941 nodes and 6594 edges and the PGP network has 10680 nodes and 24316 edges. In the construction of the proposed $1k$–$3k$ benchmarks, the community structure detected by the CNM algorithm [35] is adopted as the input for Algorithm 1. The traditional GN and LFR, and two kinds of up-to-date benchmark construction methods (HLFR and nPSO), are adopted for performance comparison. On each real-life network, each kind of benchmark construction method is conducted to generate 100 benchmarks, respectively, adopting the optimal parameter settings obtained from the original real-life network to approximate the actual structural characteristics as accurately as

possible. Structural characteristics of the benchmarks generated by all the methods on each real-life network are summarized in Tables 1–4, respectively. All of the benchmarks are compared on the basis of microscale (average degree $<k>$, assortativity coefficient r, and clustering coefficient c), mesoscale (the number of community Cn and modularity Q), and macroscale (average shortest path length l) characteristics. In Tables 1–4, the values of structural characteristics presented in the first row correspond to each original real-life network. The values shown in the other rows correspond to seven kinds of generated benchmarks, where each data represents the average of the characteristic values calculated on the 100 benchmarks and the data in parentheses represents the corresponding standard deviation. In addition, the average values of modularity Q and community number Cn of all kinds of generated benchmarks are calculated based on the community structures discovered by the same community detection algorithm (i.e., CNM). For the $1k$–$3k$ benchmarks, it is recommended to use the same community detection algorithm for the performance evaluation as that used in the construction. It can be more accurate to reflect the similarity between the obtained benchmarks and the original real-life network in terms of the community structure characteristics.

As we can see from Table 1, the structural characteristics of the generated GN, LFR, HLFR, and nPSO benchmarks are significantly different from those of the original Karate network. GN maintains the same number of nodes and communities. LFR maintains the same number of nodes and modularity value and shows a different performance to the original Karate network in the other structural characteristics. HLFR and nPSO can also maintain the same number of nodes and communities, and they are closer to the original Karate network in terms of clustering coefficient and modularity. In the process of constructing the abovementioned four kinds of benchmarks, although we set the control parameters strictly according to the characteristics of the original Karate network, the generated benchmarks are not guaranteed to have the same number of edges. The $1k$–$3k$ benchmarks generated by Algorithm 1 show significant performance advantages in almost all the structural characteristics listed in Table 1, and they are able to accurately approximate most of the structural characteristics of the original Karate network. Furthermore, by increasing the order of null models, the structural characteristics of constructed benchmarks are more and more similar to those of the original network. Specially, when the order is greater than or equal to $2k$, the generated benchmarks can accurately approximate all the microscale and mesoscale characteristics, providing desired proxies for the original network. In addition to Table 1, the results in Figure 1 also prove the advantages of the proposed $1k$–$3k$ benchmarks in accurately approximating the degree distribution of the original Karate network.

Experiment results in Tables 2–4 further prove the performance advantage of the proposed $1k$–$3k$ benchmarks in approaching the structural characteristics of real-life networks with different types and sizes. In addition, the proposed benchmarks have advantages of network diversification, that is, based on a single real-life network, multiple synthetic benchmarks that are not identical but have the same high-precision characteristics can be constructed. It helps us to obtain more accurate and reliable evaluation results for any community detection algorithm. Moreover, by comparing the benchmarks with different orders, multiscale structural characteristics of the original network can be further analysed.

In order to further clarify the difference between the generated benchmarks and the original real network in terms of network structure, we calculated the average ratio of the number of shuffled edges in the generated $1k$–$3k$ benchmarks on all the real-life networks in Table 5. As can be seen from Table 5, in the $1k$ benchmarks generated based on all the real-life networks, the number of shuffled edges is always the highest, and the average shuffled-edge ratio in the large-scale network PowerGrid reaches 89.28%. Furthermore, in each real-life network, as the order of the null model increases, the average number of shuffled edges in the corresponding benchmarks decreases. It means that the diversity of generated benchmarks is decreasing because fewer edges can be rewired, making the network structures similar to the original network. However, even for the $3k$ benchmarks, there is still a certain percentage of edges that have been be rewired. Especially, in the PowerGrid network, still more than 62% edges have been reconnected. The abovementioned experimental results show that the proposed benchmarks can actually lead to significant changes in the original real-life networks, providing diversified test networks for performance evaluation of community detection algorithms.

2.2. Benchmarks with Varying Community Intensity. Real-life benchmarks can be utilized to evaluate the accuracy of community detection algorithms. However, the robustness of algorithms can hardly be tested on real-life benchmarks directly because we do not know exactly how well the algorithms will perform if the community intensity of the original network changes. Therefore, in most previous studies, evaluating the robustness of a community detection algorithm can only be carried out on artificial synthetic benchmarks (e.g., GN and LFR) [8–13, 28–31], whose structural characteristics differ significantly from real-life networks [14, 15, 23–27].

To solve the abovementioned problems, a new class of benchmarks with independently modified community intensity is constructed based on a real-life network. A set of synthetic benchmarks with gradually modified community intensity can be achieved by directly rewiring edges. Specially, in all of these benchmarks, the community intensity can be both increased and decreased, and at the same time, other main structural characteristics of the original real-life network can be preserved as much as possible.

Constructing the benchmarks proceeds through the steps as follows. For a real-life network of any kind, we first divide it into multiple communities by employing a typical community detection algorithm (e.g., Infomap [41]). Then,

Table 1: Structural characteristics of the original Karate network and seven kinds of generated benchmarks. Each data represents the average value of the characteristic calculated on 100 generated benchmarks and the data in parentheses represents the corresponding standard deviation. Note that the best results closest to the original karate network on each characteristic are highlighted in bold.

Networks	n	m	<k>	r	c	Cn	Q	l
Karate	**34**	**78**	**4.59**	**-0.48**	**0.57**	**4**	**0.42**	**2.41**
GN	**34**	89	5.24 (±0.18)	−0.16 (±0.07)	0.12 (±0.03)	**4 (±0.00)**	0.24 (±0.01)	2.20 (±0.03)
LFR	**34**	79	4.65 (±0.38)	−0.29 (±0.08)	0.44 (±0.11)	3 (±0.49)	**0.42 (±0.05)**	2.42 (±0.08)
HLFR	**34**	66	3.91 (±0.62)	−0.22 (±0.11)	0.35 (±0.16)	**4 (±1.17)**	0.49 (±0.06)	2.85 (±0.19)
nPSO	**34**	65	3.82 (±0.01)	−0.15 (±0.02)	0.43 (±0.11)	**4 (±0.80)**	0.52 (±0.02)	2.73 (±0.05)
1k benchmark	**34**	**78**	**4.59 (±0.00)**	−0.47 (±0.01)	**0.57 (±0.03)**	**4 (±0.00)**	**0.42 (±0.00)**	2.14 (±0.03)
2k benchmark	**34**	**78**	**4.59 (±0.00)**	**−0.48 (±0.00)**	**0.57 (±0.01)**	**4 (±0.00)**	**0.42 (±0.00)**	2.39 (±0.01)
3k benchmark	**34**	**78**	**4.59 (±0.00)**	**−0.48 (±0.00)**	**0.57 (±0.00)**	**4 (±0.00)**	**0.42 (±0.00)**	**2.41 (±0.01)**

Note. n: number of nodes; m: number of edges; <k>: average degree; r: assortativity coefficient; c: clustering coefficient; Cn: community number; Q: modularity; and l: average shortest path length. Parameters of GN benchmarks: n: 34; <k>: 4.59; Cn: 4; and Z_{out}: 2. Parameters of LFR benchmarks: n: 34; <k>: 4.59; max$k$: 17; mu: 0.20; min$c$: 5; and max$c$: 12. Parameters of HLFR benchmarks: n: 34; <k>: 4.59; max$k$: 17; mu_1: 0.09; min$c$: 5; max$c$: 6; min$C$: 11; max$C$: 12; and mu_2: 0.56. Parameters of nPSO benchmarks: n: 34; <k>: 4.59; Cn: 4; T: 0.1; gamma: 3; and plot_flag: 1. Parameters of 1k–3k benchmarks: n: 34; m: 78; and δ: 0.01.

Table 2: Structural characteristics of the original anatomical connection network of the macaque cortex and seven kinds of generated benchmarks. Each data represents the average value of the characteristic calculated on 100 generated benchmarks and the data in parentheses represents the corresponding standard deviation. Note that the best results closest to the original karate network on each characteristic are highlighted in bold.

Networks	n	m	<k>	r	c	Cn	Q	l
Anatomical connection	**71**	**438**	**12.34**	**0.09**	**0.50**	**4**	**0.39**	**2.24**
GN	**71**	538	15.02 (±0.16)	−0.14 (±0.04)	0.22 (±0.01)	**4 (±0.00)**	0.23 (±0.01)	1.82 (±0.01)
LFR	**71**	427	12.05 (±0.42)	0.01 (±0.13)	0.42 (±0.02)	**4 (±0.75)**	0.45 (±0.02)	2.06 (±0.05)
HLFR	**71**	405	11.43 (±0.43)	−0.13 (±0.01)	0.39 (±0.02)	2 (±0.40)	0.50 (±0.05)	2.12 (±0.04)
nPSO	**71**	405	11.41 (±0.00)	0.08 (±0.03)	0.56 (±0.01)	**4 (±0.00)**	0.46 (±0.01)	2.12 (±0.02)
1k benchmark	**71**	**438**	**12.34 (±0.00)**	0.04 (±0.02)	0.46 (±0.01)	**4 (±0.00)**	0.38 (±0.01)	2.17 (±0.01)
2k benchmark	**71**	**438**	**12.34 (±0.16)**	**0.09 (±0.00)**	**0.50 (±0.01)**	**4 (±0.00)**	**0.39 (±0.01)**	2.23 (±0.01)
3k benchmark	**71**	**438**	**12.34 (±0.16)**	**0.09 (±0.00)**	**0.50 (±0.00)**	**4 (±0.00)**	**0.39 (±0.00)**	**2.24 (±0.01)**

Note. n: number of nodes; m: number of edges; <k>: average degree; r: assortativity coefficient; c: clustering coefficient; Cn: community number; Q: modularity; and l: average shortest path length. Parameters of GN benchmarks: n: 71; <k>: 12.34; Cn: 4; and Z_{out}: 6. Parameters of LFR benchmarks: n: 71; <k>: 12.34; max$k$: 28; mu: 0.27; min$c$: 7; and max$c$: 28. Parameters of HLFR benchmarks: n: 71; <k>: 12.34; max$k$: 28; mu_1: 0.12; min$c$: 7; max$c$: 16; min$C$: 20; max$C$: 28; and mu_2: 0.59. Parameters of nPSO benchmarks: n: 71; <k>: 12.34; Cn: 4; T: 0.1; gamma: 3; and plot_flag: 1. Parameters of 1k–3k benchmarks: n: 71; m: 438; and δ: 0.01.

Table 3: Structural characteristics of the original PowerGrid network and seven kinds of generated benchmarks. Each data represents the average value of the characteristic calculated on 100 generated benchmarks and the data in parentheses represents the corresponding standard deviation. Note that the best results closest to the original karate network on each characteristic are highlighted in bold.

Networks	n	m	<k>	r	c	Cn	Q	l
PowerGrid	**4941**	**6594**	**2.67**	**0.003**	**0.08**	**40**	**0.94**	**18.99**
GN	**4941**	12989	5.26 (±0.01)	−0.02 (±0.01)	0.01 (±0.01)	**40 (±0.00)**	0.44 (±0.01)	5.43 (±0.01)
LFR	**4941**	5501	2.23 (±0.01)	−0.08 (±0.01)	0.04 (±0.01)	72 (±15.85)	0.97 (±0.01)	14.46 (±0.01)
HLFR	**4941**	4827	1.95 (±0.02)	0.04 (±0.01)	0.01 (±0.01)	74 (±18.00)	0.97 (±0.01)	15.21 (±0.01)
nPSO	**4941**	4940	2.00 (±0.01)	−0.05 (±0.01)	0.00 (±0.00)	70 (±3.98)	0.97 (±0.01)	9.80 (±0.21)
1k benchmark	**4941**	**6594**	**2.67 (±0.00)**	0.004 (±0.01)	0.02 (±0.01)	**40 (±0.00)**	0.93 (±0.01)	18.31 (±0.48)
2k benchmark	**4941**	**6594**	**2.67 (±0.00)**	**0.003 (±0.00)**	0.04 (±0.01)	**40 (±0.00)**	0.93 (±0.01)	18.51 (±0.40)
3k benchmark	**4941**	**6594**	**2.67 (±0.00)**	**0.003 (±0.00)**	**0.08 (±0.00)**	**40 (±0.00)**	**0.94 (±0.01)**	**18.99 (±0.07)**

Note. n: number of nodes; m: number of edges; <k>: average degree; r: assortativity coefficient; c: clustering coefficient; Cn: community number; Q: modularity; and l: average shortest path length. Parameters of GN benchmarks: n: 4941; <k>: 2.67; Cn: 40; and Z_{out}: 0.09. Parameters of LFR benchmarks: n: 4941; <k>: 2.67; max$k$: 19; mu: 0.0343; min$c$: 18; and max$c$: 207. Parameters of HLFR benchmarks: n: 4941; <k>: 2.67; max$k$: 11; mu_1: 0.006; min$c$: 18; max$c$: 129; min$C$: 131; max$C$: 207; and mu_2: 0.62. Parameters of nPSO benchmarks: n: 4941; <k>: 2.67; Cn: 40; T: 0.1; gamma: 3; and plot_flag: 1. Parameters of 1k–3k benchmarks: n: 4941; m: 6594; and δ: 0.01.

with the original microscale structures (e.g., degree sequence) unmodified, the intracommunity and intercommunity edges are randomly reconnected to adjust the significance of community structures. For example, for a simple network with two communities shown in Figure 2(a), in order to enhance community structures, a pair of intercommunity edges $A1$–$B1$ and $A5$–$B3$ are randomly selected to disconnect. At the same time, the corresponding two intracommunity edges $A1$–$A5$ and $B1$–$B3$ are built with the degrees of four nodes unchanged. As we can see from the

Table 4: Structural characteristics of the original PGP network and seven kinds of generated benchmarks. Each data represents the average value of the characteristic calculated on 100 generated benchmarks and the data in parentheses represents the corresponding standard deviation. Note that the best results closest to the original karate network on each characteristic are highlighted in bold.

Networks	n	m	<k>	r	c	Cn	Q	l
PGP	**10680**	**24315**	**4.55**	**0.24**	**0.27**	**110**	**0.88**	**7.49**
GN	**10680**	23468	4.39 (±0.01)	0.01 (±0.02)	0.01 (±0.01)	**110 (±0.00)**	0.74 (±0.01)	7.93 (±0.01)
LFR	**10680**	19035	3.56 (±0.10)	−0.29 (±0.02)	0.16 (±0.01)	85 (±3.96)	0.91 (±0.01)	7.20 (±0.04)
HLFR	**10680**	32485	6.08 (±0.10)	−0.11 (±0.02)	0.14 (±0.01)	36 (±1.50)	0.93 (±0.01)	5.64 (±0.09)
nPSO	**10680**	21357	4.00 (±0.00)	0.01 (±0.05)	0.38 (±0.01)	57 (±3.72)	0.95 (±0.01)	6.72 (±0.14)
1k benchmark	**10680**	**24315**	**4.55 (±0.00)**	0.15 (±0.01)	0.09 (±0.02)	**110 (±0.00)**	0.86 (±0.01)	6.79 (±0.01)
2k benchmark	**10680**	**24315**	**4.55 (±0.00)**	**0.24 (±0.00)**	0.14 (±0.03)	**110 (±0.00)**	0.87 (±0.01)	7.22 (±0.01)
3k benchmark	**10680**	**24315**	**4.55 (±0.00)**	**0.24 (±0.00)**	**0.27 (±0.00)**	**110 (±0.00)**	**0.88 (±0.01)**	**7.47 (±0.01)**

Note. n: number of nodes; m: number of edges; <k>: average degree; r: assortativity coefficient; c: clustering coefficient; Cn: community number; Q: modularity; and l: average shortest path length. Parameters of GN benchmarks: n: 10680; <k>: 4.55; Cn: 110; and Z_{out}: 1. Parameters of LFR benchmarks: n: 10680; <k>: 4.55; max$k$: 194; mu: 0.08; min$c$: 6; and max$c$: 664. Parameters of HLFR benchmarks: n: 10680; <k>: 4.55; max$k$: 72; mu_1: 0.04; min$c$: 6; max$c$: 97; min$C$: 107; max$C$: 664; and mu_2: 0.79. Parameters of nPSO benchmarks: n: 10680; <k>: 4.55; Cn: 110; T: 0.1; gamma: 3; and plot_flag: 1. Parameters of 1k–3k benchmarks: n: 10680; m: 24315; and δ: 0.01.

Table 5: The average ratio of the number of actually changed edges in the generated 1k–3k benchmarks on four real-life networks with different types and sizes.

Networks	Karate (%)	Anatomical connection (%)	PowerGrid (%)	PGP (%)
1k benchmarks	21.79	28.77	89.28	61.11
2k benchmarks	14.10	16.21	90.14	16.84
3k benchmarks	7.69	1.83	62.09	5.01

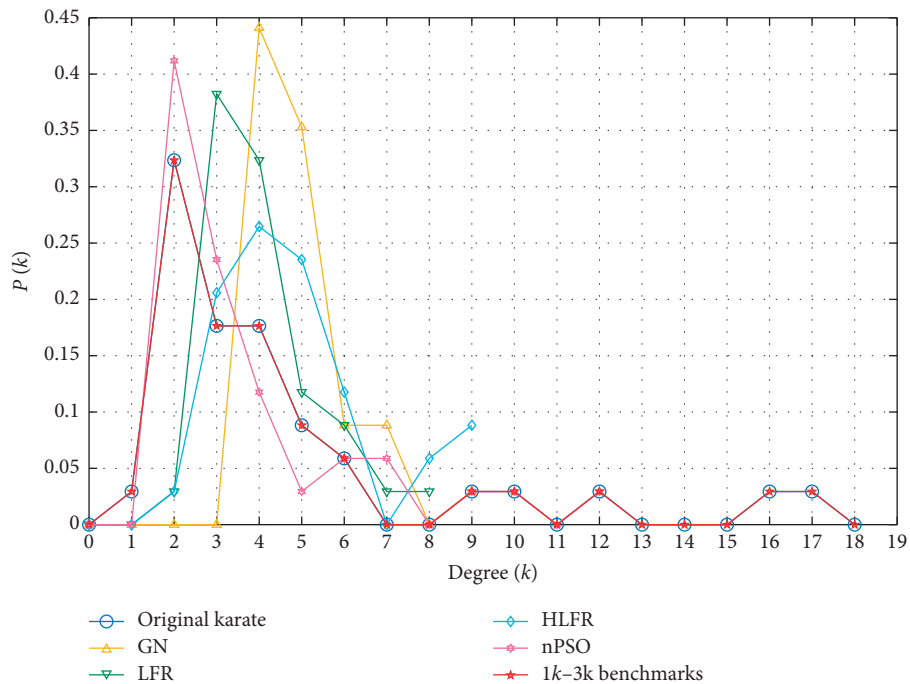

Figure 1: Degree distribution of the original Karate network and seven kinds of generated benchmarks.

new benchmark after reconnection shown in Figure 2(b), the community density has been obviously enhanced. Repeating the abovementioned reconnection operation until no existing edges can be rewired, we obtain a set of synthetic benchmarks with community structures gradually strengthened, and at the same time maximally maintaining the microscale characteristic of the original network. On the contrary, we can also get a series of benchmarks with gradually weakened community structures by employing the method of edge rewiring, as shown in Figure 3.

About the second kind of benchmark construction method, there are some points that need further explanation as follows. (1) The abovementioned benchmark constructing method is only applicable to the unweighted and undirected

networks; the influence of the weight information and directional characteristics of edges on the benchmark construction are not considered. (2) There is no strict requirement for the selection of community detection algorithm in the first step. From the perspective of providing a reference standard for constructing a series of benchmarks with gradually changed community intensity, the selection of different community detection algorithms has little impact on the generated benchmarks. The functionality of the generated benchmarks (i.e., evaluating the robustness of different community detection algorithms) will not be affected. Anyway, in order to keep the mesoscale characteristic of the original real-life network as accurately as possible, we try to choose the community detection algorithms with relatively higher performance on real-life networks. Moreover, when the precisions of several algorithms are almost the same, the algorithm with low time complexity is preferred to speed up the benchmark construction.

The computational complexity of the abovementioned method mainly depends on the two steps in the benchmark construction. In the first step, the computational complexity is determined by the community detection algorithm adopted, such as $O(n(n+m))$ for Infomap. In the second step, there are two different kinds of random edge rewiring operations with a different computational complexity. Specifically, for the operation to strengthen the community structure, the complexity is no more than the total number of edges between any two communities. For the operation to weaken the community structure, the complexity is no more than the total number of edges within all communities. Therefore, the computational complexity of the second step is less than $O(m)$. To sum up, the computational complexity of the benchmark construction method can be roughly estimated as $O(n(n+m)) + O(m)$ when using Infomap, which can be further simplified to $O(n(n+m))$.

The performance of the new benchmark construction method is tested on two kinds of real-life networks with different types and sizes, including a social network (i.e., American college football [21]) and a biological network (i.e., an anatomical connection network of the macaque cortex [37, 38]). For the Football network, by using the initial community partition detected by DECD [42] and the $1k$ random edge rewiring null model, 230 benchmarks with different community structures are generated which have gradually modified community intensity. Among all the benchmarks, the community intensity of the first 30 benchmarks is stronger than that of the original Football network, while the subsequent 199 benchmarks have weaker community intensity. In addition, the $1k$ microscale characteristic of all the 230 benchmarks (i.e., degree distribution) is the same as the original network. The generated benchmarks are utilized to evaluate the accuracy and robustness of several typical community detection algorithms, including KClique [43], CNM [35], GN [44], LPA [45], LENC [46], Infomap [41], and DECD [42]. Modularity (Q) [47] and Normalized Mutual Information (NMI) [48] are used to evaluate the quality of community detection algorithms. Statistical results are presented in Figures 4 and 5, where each data represents an average optimal value of Q and NMI obtained by each algorithm after 10 independent runs on each benchmark.

Figure 4 exhibits the accuracy variation of detection results achieved by each algorithm as the ambiguity of community structure increases. The position where the x-coordinate is zero represents the detection results on the original network. As we can see, taking this position as the baseline, continuously enhancing the community intensity (the value of x-coordinate ranges from -30 to 0) has no significant impact on detection results because the modularity values obtained by all the algorithms remain basically unchanged. On the contrary, continuously weakening the community intensity (the value of x-coordinate ranges from 0 to 200), the accuracy of all the algorithms significantly decreases, among which KClique, LPA, and Infomap show the most dramatic performance degradation, and all of them cannot get meaningful community divisions when the number of reconnected edges exceeds 100. By contrast, CNM, DECD, LENC, and GN show a continuous and relatively stable downward trend in accuracy. When the community structure is fuzzy to a certain extent (the x-coordinate is greater than 120), the detection accuracy of the four algorithms is basically unchanged at a certain level. According to the abovementioned experimental results, we can see that compared with KClique, LPA, and Infomap algorithms, CNM, DECD, LENC, and GN show the stronger robustness because the accuracy of their detection performance is less affected by modifying the community structures.

In Figure 5, the accuracy variation of community detection results achieved by all of the seven algorithms are further evaluated by the NMI index. The community partition detected by the DECD on the original real network is used as the ground truth to calculate NMI. As can be seen from Figure 5, as the ambiguity of community structure increases, the NMI values obtained by all the algorithms also show a trend of gradual decline, indicating that the accuracy of all the algorithms gradually decreases. Through further careful observation, it is found that when the community intensity increases gradually, the NMI values of the seven community detection algorithms are basically unchanged, indicating that all the algorithms have good stability when the community structure is obvious. However, when the community intensity is gradually weakened, the NMI values of GN, DECD, CNM, and LENC algorithms decline gently, while the NMI values of KClique, LPA, and Infomap algorithms decline sharply, so their robustness is relatively poor. Among the seven kinds of algorithms, except GN algorithm, the robustness of most algorithms under NMI is basically the same as that under Q. The robustness and accuracy of GN under NMI is obviously better than that under Q. This may be because the NMI index is sensitive to the number of clusters of the detected partition and may attain larger values when the number of clusters is large [1]. In our experiments, we find that the detection results of GN contain many communities with a small scale. The abovementioned experimental results show that the generated benchmarks can effectively evaluate the

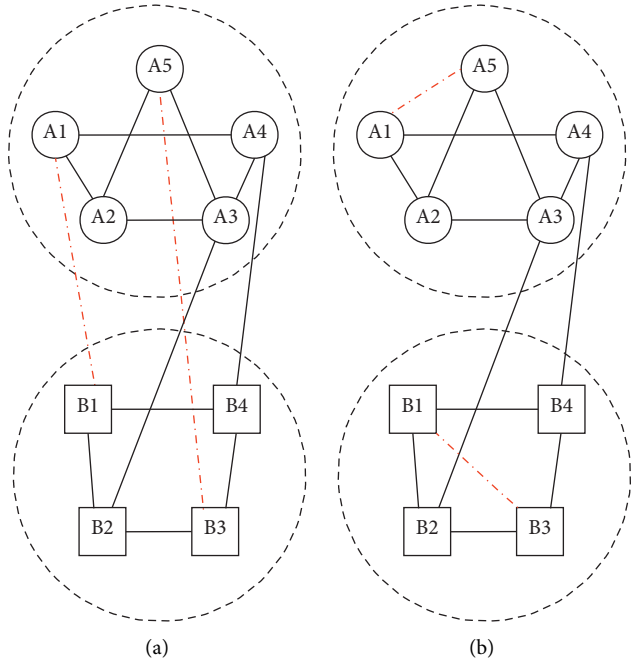

Figure 2: The operation of edge rewiring for strengthening community structures.

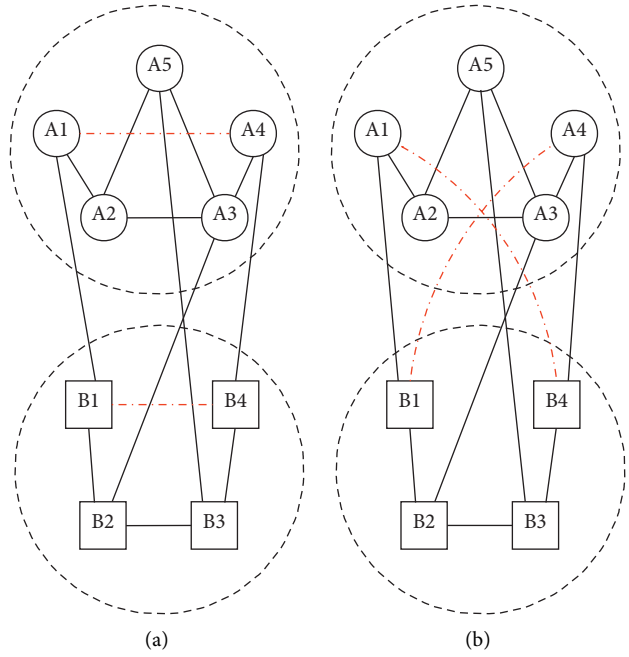

Figure 3: The operation of edges rewiring for weakening community structures.

robustness of each algorithm when different performance indexes are used. The diversified evaluation results under different indexes (Q and NMI) also make the robustness evaluation of community detection algorithms more comprehensive and reliable.

In order to further prove that when the community intensity of benchmarks gradually modifies from strong to weak, other main structural characteristics of the original real-life network can be preserved as much as possible. The performance variation of the benchmarks with different orders in four typical micro- and macrocharacteristics are plotted in Figure 6. In order to facilitate performance comparison, based on the same original Football network, two sets of benchmarks are constructed by using the random edge rewiring model with different orders. The benchmarks generated based on $1k$ null model is represented by red line, and the benchmarks generated based on $2k$ null model is represented by blue line. The blue lines stop when the

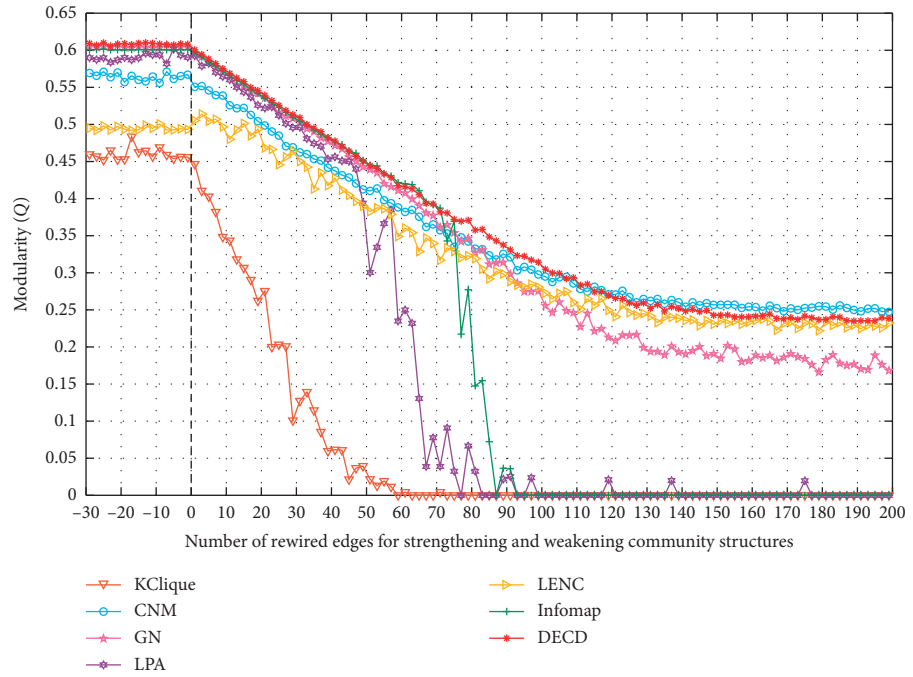

Figure 4: The average optimal value of Q obtained by typical community detection algorithms on a set of benchmarks with gradually modified community intensity, which are generated based on the American College Football (ACF) network. Each data point is obtained based on averaging 10 independent runs of each algorithm on each benchmark. The negative and positive values of the abscissa represent the number of rewired edges when the community structure is strengthened and weakened, respectively.

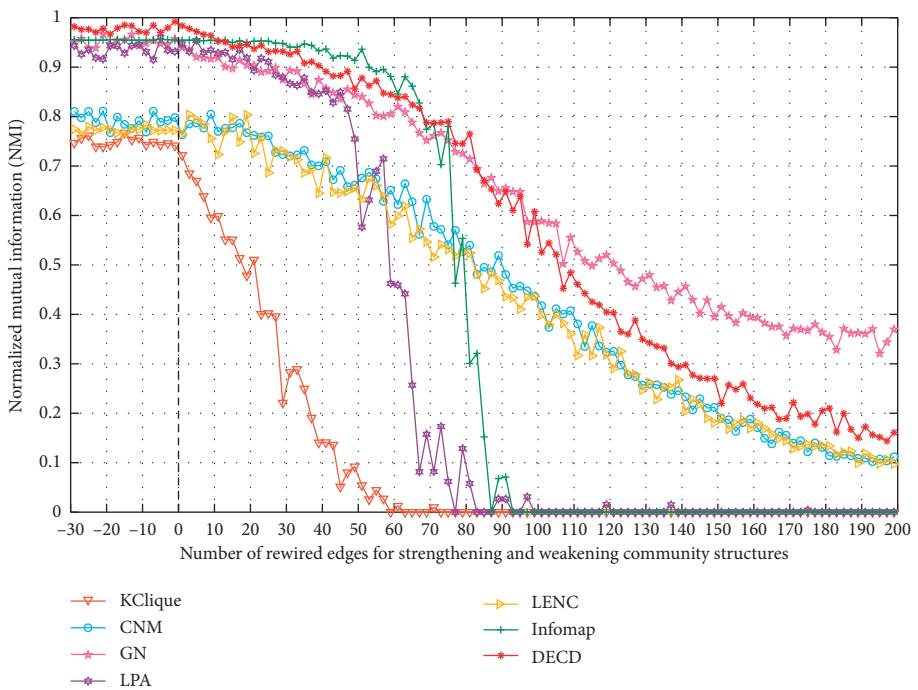

Figure 5: The average optimal value of *NMI* obtained by typical community detection algorithms on a set of benchmarks with gradually modified community intensity, which are generated based on the American College Football (ACF) network. Each data point is obtained based on averaging 10 independent runs of each algorithm on each benchmark. The negative and positive values of the abscissa represent the number of rewired edges when the community structure is strengthened and weakened, respectively.

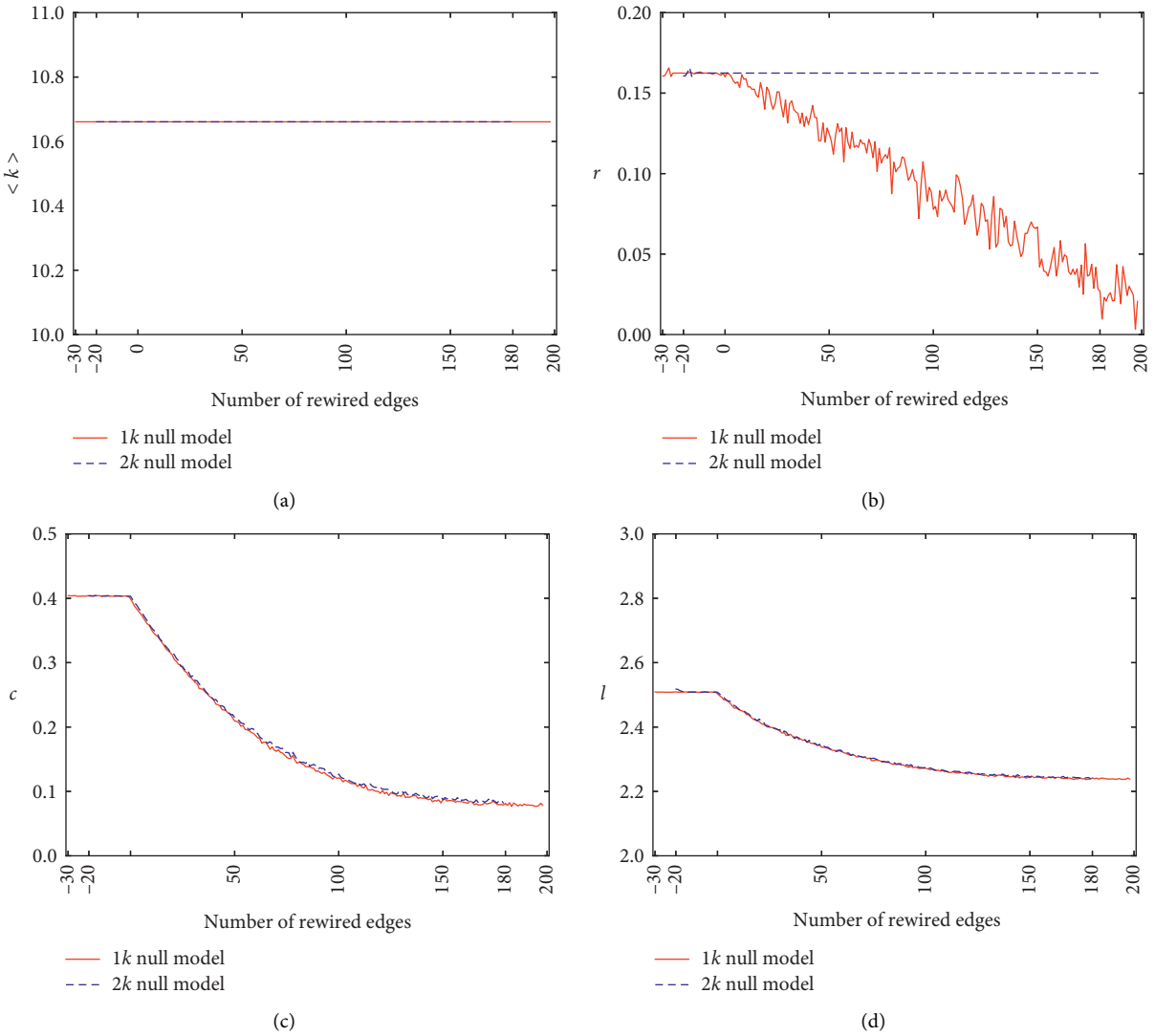

Figure 6: Structural characteristic of the benchmarks generated based on the football network, including average degree <k>, assortativity coefficient r, clustering coefficient c, and average shortest path length l. Red lines correspond to the benchmarks generated based on the 1k null model, and blue lines correspond to the benchmarks generated based on the 2k null model. The negative and positive values of the abscissa represent the number of rewired edges when the community structure is strengthened and weakened, respectively.

number of rewired edges achieves 180 because the number of benchmarks generated based on the 2k null model is not as large as that of the 1k null model.

Experimental results in Figure 6 show that, in the 1k benchmarks, only one microscale characteristic of the original Football network (i.e., average degree <k>) can be kept strictly within each benchmark network. In contrast, in the 2k benchmarks, both of the average degree <k> and the assortativity coefficient r of the original Football network can be strictly maintained. In the two sets of benchmarks, although the 3k microscale characteristic (i.e., clustering coefficient c) and the average shortest path length l change as the number of rewired edges increases, the range of their variation is limited. When more structure characteristics of the original real network remain unchanged, the robustness evaluation of community detection algorithms suffers less interference, thus ensuring the accuracy and reliability of evaluation results. However, the 1k benchmark has greater diversity than high-order models, which helps to completely reflect the variation tendency of the algorithm performance when the community intensity varies from strong to weak.

Secondly, on the biological brain network (i.e., an anatomical connection network of the macaque cortex), by using the initial community partition detected by DECD [42] and the 1k random edge rewiring null model, 170 benchmarks with different community structures are generated which have gradually modified community intensity. Among all the real-life benchmarks, the community intensity of the first 20 benchmarks is stronger than that of the original Football network, while the subsequent 149 benchmarks have weaker community intensity. Robustness of seven well-known community detection algorithms, including GN [44], LPA [45], LENC [46], Infomap [41], Louvain [49], Walktrap [50], and DECD [42], are tested on

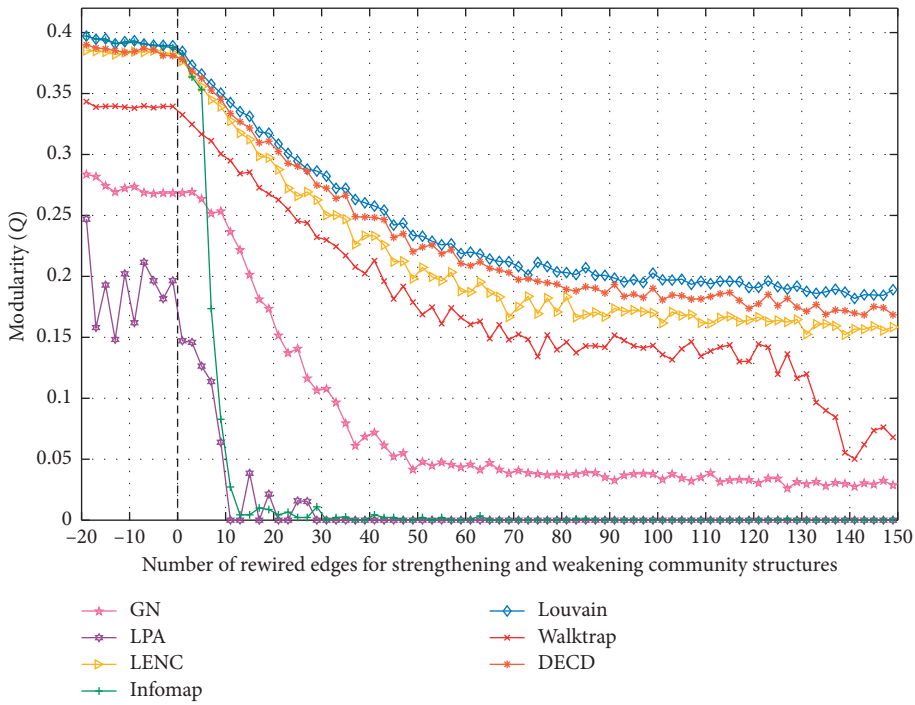

Figure 7: The average optimal value of Q obtained by typical community detection algorithms on a set of benchmarks with gradually modified community intensity, which are generated based on the anatomical connection network of macaque cortex. Each data point is obtained based on averaging 10 independent runs of each algorithm on each benchmark. The negative and positive values of the abscissa represent the number of rewired edges when the community structure is strengthened and weakened, respectively.

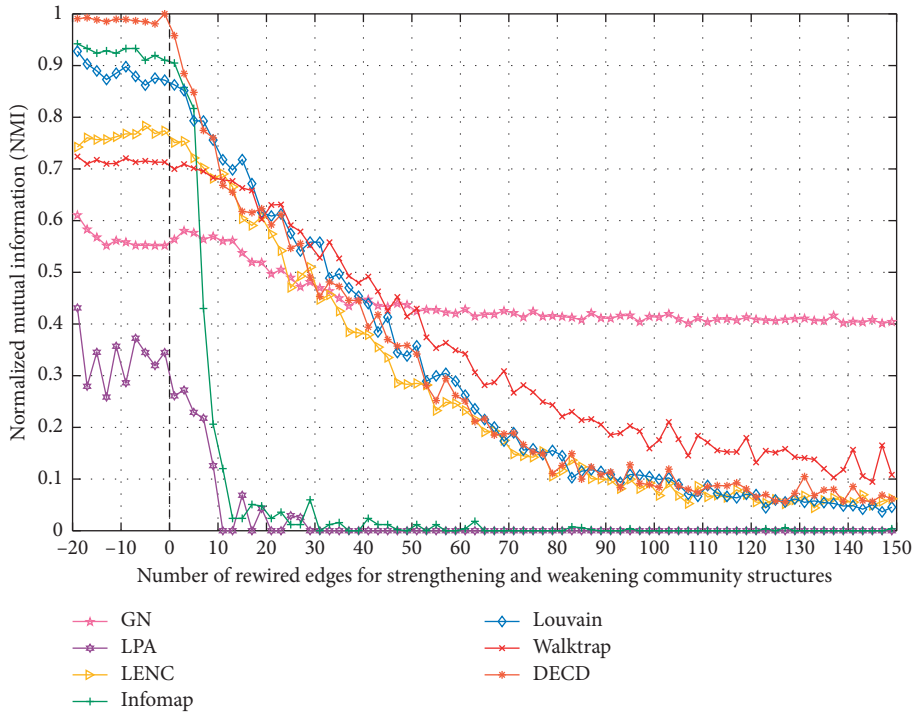

Figure 8: The average optimal value of *NMI* obtained by typical community detection algorithms on a set of benchmarks with gradually modified community intensity, which are generated based on the anatomical connection network of macaque cortex. Each data point is obtained based on averaging 10 independent runs of each algorithm on each benchmark. The negative and positive values of the abscissa represent the number of rewired edges when the community structure is strengthened and weakened, respectively.

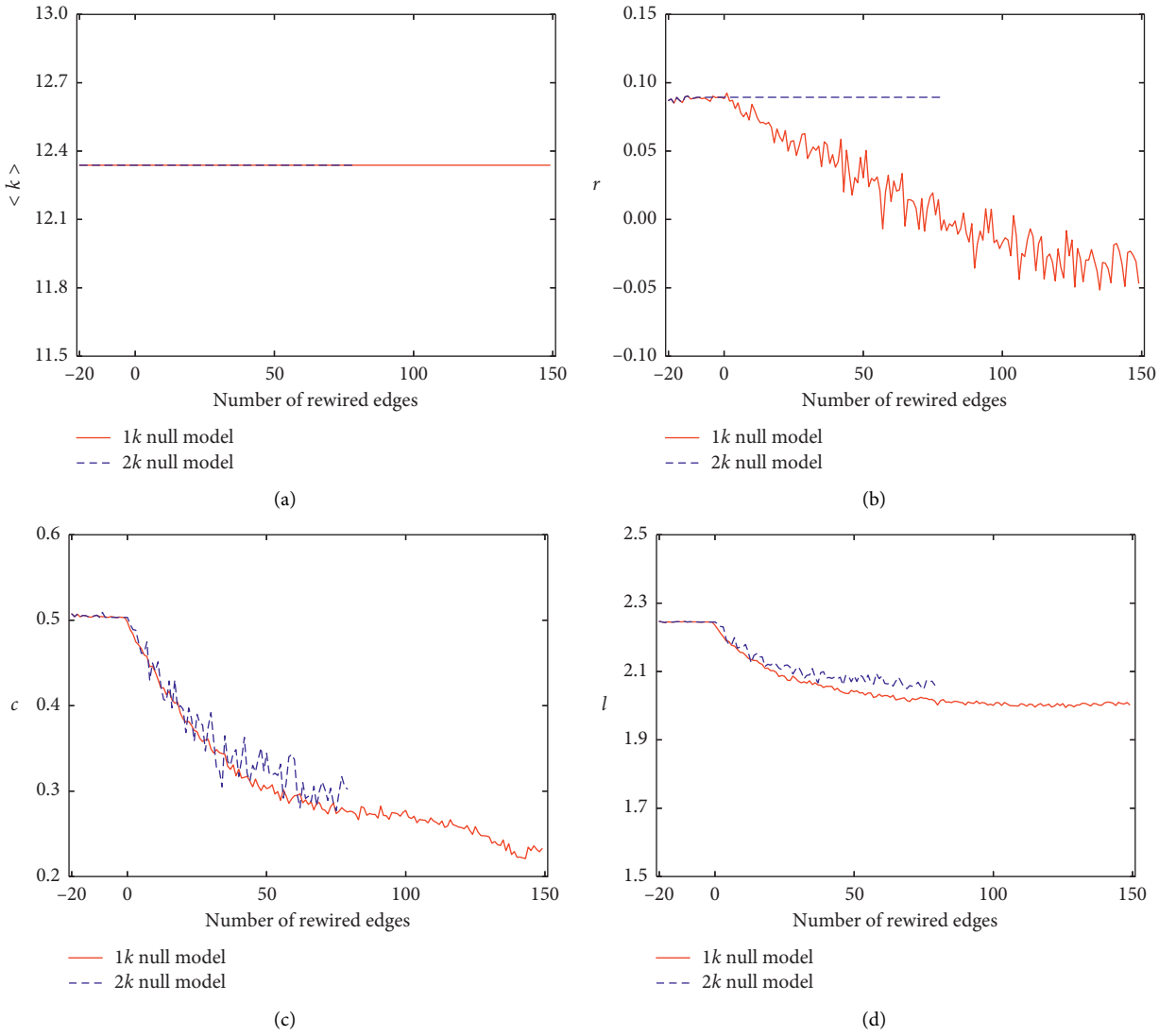

FIGURE 9: Structural characteristic of the benchmarks generated based on the anatomical connection network of the macaque cortex, including average degree <k>, assortativity coefficient r, clustering coefficient c, and average shortest path length l. Red lines correspond to the benchmarks generated based on the 1k null model, and blue lines correspond to the benchmarks generated based on the 2k null model. The negative and positive values of the abscissa represent the number of rewired edges when the community structure is strengthened and weakened, respectively.

the benchmarks. The performance of all the algorithms evaluated by the indexes of Q and NMI are shown in Figures 7 and 8, respectively.

The experimental results in Figure 7 show that when the community structure is clear, the detection results of Q for most algorithms are stable, while only LPA algorithm fluctuates greatly. However, when the community structure is gradually blurred, the performance of Infomap and LPA declines almost linearly. When the number of rewired edges is greater than 10, both of them can hardly get effective detection results. In contrast, the algorithms of Louvain, DECD, LENC, and Walktrap show strong stability. Although their precision decreases with the gradual weakening of community structures, the performance changes relatively gently. Compared with the values of Q, the evaluation results under the NMI index are significantly different. The community partition detected by DECD on the original real network is used as the ground truth to calculate NMI. As shown in Figure 8, the value of NMI obtained by GN is not high, but basically remains unchanged when the community structure becomes weak. Algorithms of Walktrap, Louvain, DECD, and LENC show strong robustness under Q index, but their NMI values drop significantly when the community structure is weakened. In addition, the precision of Walktrap becomes better than that of Louvain and LENC when the number of rewired edges is greater than 30. The experimental results in Figures 7 and 8 show that on the benchmarks generated based on the biological brain network, the robustness of different kinds of community detection algorithms can also be effectively evaluated.

In addition, in Figure 9, we also plot the performance variation of the generated benchmarks in four typical micro-

and macrocharacteristics and investigate the influence of the variations in mesoscale characteristic (i.e., community intensity) on the other main structural characteristics. Based on the original anatomical connection network, two sets of benchmarks are generated based on the 1k and 2k null models, respectively. In Figure 9, they are represented by red and blue lines, respectively. As shown in Figure 9, the blue lines stop when the number of rewired edges achieves 80 because the number of benchmarks generated based on the 2k null model is not as large as the number of benchmarks generated based on the 1k null model.

Experimental results in Figure 9 show that the 1k benchmarks only preserve the original average degree <k>, but the 2k benchmarks can further maintain the assortativity coefficient r. In the two sets of benchmarks, although the 3k microscale characteristic (i.e., clustering coefficient c) and the average shortest path length l changes as the number of rewired edges increases, the range of their variations is also limited. If the benchmarks are generated by using a null model with a higher order (e.g., 3k null model), they are able to preserve more structure characteristics of the original real-life network, but the diversity of benchmarks will be reduced at the same time. By contrast, the 1k-2k benchmarks are more suitable to be used for evaluating robustness of community detection algorithms.

The abovementioned experiments suggest that the proposed second kind of benchmarks can not only effectively evaluate the accuracy of community detection algorithms but also directly detect their robustness. In addition, because the microscale characteristic of the original real-life network can be maintained in all the constructed benchmarks as much as possible, the detection performance is mainly affected by the variation of community structures, so evaluating the robustness of community detection algorithms suffers less interference and can be more accurate.

Moreover, existing real-life benchmarks of community detection are mostly social networks because the actual community division of these networks is usually known. The prior community information is not required in the benchmark construction process, so more types of real-life networks, such as biological networks, engineering networks, and communication networks, can be used to build benchmarks for community detection. As a result, the performance evaluation of algorithms can be more comprehensive and reliable, and our analysis of community structural characteristics for different types of real networks will also be more in-depth.

3. Conclusions

In summary, we present a benchmark constructing method based on edge rewiring of real-life networks, based on which two kinds of special functions are designed. The new proposed benchmarks can not only make the performance evaluation of community detection algorithms more accurate and reliable but also help to deepen our analysis of community structures and functional characteristics of real-life networks.

Among the newly designed two kinds of benchmarks, the first kind can accurately approximate the microscale and mesoscale structural characteristics of the original network, and thus making the algorithm evaluation more accurate and realizing performance analysis of a real network in different microscale orders. The second kind is able to independently modify the community intensity in each generated benchmark, without interfering with microscale characteristic of the original network, and thus evaluating the robustness of detection algorithms can be more accurate.

The proposed methods of constructing benchmarks can be applied to any kind of real-life networks because no prior community structure information is required. Therefore, more types of real-life networks can be used to construct benchmarks, making the performance evaluation of community detection algorithms more comprehensive and reliable. At the same time, the community structural characteristics of different types of real-life networks will also be further studied. In the future work, we will further explore whether they can be extended to build real-life benchmarks in community detection of weighted [47], directed [51], and signed networks [52].

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request, and the codes of constructing benchmarks can be found on https://github.com/Lonelycat1/Community-nullmodel.

Conflicts of Interest

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

Acknowledgments

This work was supported by the National Natural Science Foundation of China (61773091 and 61603073), Key Research and Development Plan of Liaoning Province (2018104016), LiaoNing Revitalization Talents Program (XLYC1807106), and Program for the Outstanding Innovative Teams of Higher Learning Institutions of Liaoning (LR2016070).

References

[1] S. Fortunato and D. Hric, "Community detection in networks: a user guide," *Physics Reports*, vol. 659, pp. 1–44, 2016.

[2] B. S. Khan and M. A. Niazi, "Network community detection: a review and visual survey," 2017, https://arxiv.org/pdf/1708.00977.

[3] M. A. Javed, M. S. Younis, S. Latif, J. Qadir, and A. Baig, "Community detection in networks: a multidisciplinary review," *Journal of Network and Computer Applications*, vol. 108, pp. 87–111, 2018.

[4] S. A. Ríos and I. F. Videla–Cavieres, "Generating groups of products using graph mining techniques," *Procedia Computer Science*, vol. 35, pp. 730–738, 2014.

[5] A. Ali, J. Qadir, R. U. Rasool et al., "Big data for development: applications and techniques," *Big Data Analytics*, vol. 1, no. 1, p. 2, 2016.

[6] J. Qadir, A. Ali, R. U. Rasool et al., "Crisis analytics: big data-driven crisis response," *Journal of International Humanitarian Action*, vol. 1, no. 1, pp. 1–12, 2016.

[7] W. Liu and L. Chen, "Community detection in disease-gene network based on principal component analysis," *Tsinghua Science and Technology*, vol. 18, no. 5, pp. 454–461, 2013.

[8] X. Zhang, C. Wang, Y. Su, L. Pan, and H.-F. Zhang, "A fast overlapping community detection algorithm based on weak cliques for large-scale networks," *IEEE Transactions on Computational Social Systems*, vol. 4, no. 4, pp. 218–230, 2017.

[9] C. Pizzuti, "Evolutionary computation for community detection in networks: a review," *IEEE Transactions on Evolutionary Computation*, vol. 22, no. 3, pp. 464–483, 2018.

[10] W. Zhang, R. Zhang, R. Shang, J. Li, and L. Jiao, "Application of natural computation inspired method in community detection," *Physica A: Statistical Mechanics and Its Applications*, vol. 515, pp. 130–150, 2019.

[11] T. Chakraborty, A. Dalmia, A. Mukherjee et al., "Metrics for community analysis: a survey," *ACM Computing Surveys*, vol. 50, no. 4, p. 54, 2016.

[12] X. Zhou, K. Yang, Y. Xie, C. Yang, and T. Huang, "A novel modularity-based discrete state transition algorithm for community detection in networks," *Neurocomputing*, vol. 334, pp. 89–99, 2019.

[13] H. S. Pattanayak, A. L. Sangal, and H. K. Verma, "Community detection in social networks based on fire propagation," *Swarm and Evolutionary Computation*, vol. 44, pp. 31–48, 2019.

[14] A. Lancichinetti, S. Fortunato, and F. Radicchi, "Benchmark graphs for testing community detection algorithms," *Physical Review E*, vol. 78, no. 2, Article ID 046110, 2008.

[15] A. Lancichinetti and S. Fortunato, "Benchmarks for testing community detection algorithms on directed and weighted graphs with overlapping communities," *Physical Review E*, vol. 80, no. 1, Article ID 016118, 2009.

[16] G. K. Orman, V. Labatut, and H. Cherifi, "Comparative evaluation of community detection algorithms: a topological approach," *Journal of Agricultural & Food Chemistry*, vol. 46, no. 3, pp. 820–824, 2012.

[17] J. Xie, S. Kelley, and B. K. Szymanski, "Overlapping community detection in networks: the state-of-the-art and comparative study," *ACM Computing Surveys*, vol. 45, no. 4, pp. 1–35, 2011.

[18] S. Harenberg, G. Bello, L. Gjeltema et al., "Community detection in large-scale networks: a survey and empirical evaluation," *Wiley Interdisciplinary Reviews: Computational Statistics*, vol. 6, no. 6, pp. 426–439, 2014.

[19] A. Said, R. A. Abbasi, O. Maqbool, A. Daud, and N. R. Aljohani, "CC-GA: a clustering coefficient based genetic algorithm for detecting communities in social networks," *Applied Soft Computing*, vol. 63, no. 1274, pp. 59–70, 2018.

[20] Z. Yang, R. Algesheimer, and C. J. Tessone, "A comparative analysis of community detection algorithms on artificial networks," *Scientific Reports*, vol. 6, no. 1, Article ID 30750, 2016.

[21] M. Girvan and M. E. J. Newman, "Community structure in social and biological networks," *Proceedings of The National Academy of Sciences*, vol. 99, no. 12, pp. 7821–7826, 2002.

[22] P. G. Sun and X. Sun, "Complete graph model for community detection," *Physica A: Statistical Mechanics and Its Applications*, vol. 471, pp. 88–97, 2017.

[23] G. K. Orman and V. Labatut, "The effect of network realism on community detection algorithms," in *Proceedings of the International Conference on Advances in Social Networks Analysis and Mining (ASONAM 2010)*, pp. 301–305, Odense, Denmark, 2010.

[24] H. Elhadi and G. Agam, "Structure and attributes community detection benchmark and A novel selection method," in *Proceedings of the 2013 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining, Niagara*, ACM Press, Ontario, Canada, pp. 1474–1476, 2013.

[25] Z. Yang, J. I. Perotti, and C. J. Tessone, "Hierarchical benchmark graphs for testing community detection algorithms," *Physical Review E*, vol. 96, no. 5, Article ID 052311, 2017.

[26] B. D. Le, H. X. Nguyen, H. Shen et al., "GLFR: A generalized LFR benchmark for testing community detection algorithms," in *Proceedings of the International Conference on Computer Communication and Networks (ICCCN)*, July-August 2017.

[27] A. Muscoloni and C. V. Cannistraci, "A nonuniform popularity-similarity optimization (nPSO) model to efficiently generate realistic complex networks with communities," *New Journal of Physics*, vol. 20, no. 5, Article ID 052002, 2018.

[28] S. He, G. Jia, Z. Zhu et al., "Cooperative Co-evolutionary module identification with application to cancer disease module discovery," *IEEE Transactions on Evolutionary Computation*, vol. 20, no. 6, pp. 874–891, 2016.

[29] J. Xiao, Y.-J. Zhang, and X.-K. Xu, "Convergence improvement of differential evolution for community detection in complex networks," *Physica A: Statistical Mechanics and Its Applications*, vol. 503, pp. 762–779, 2018.

[30] J. Xiao, C. Wang, and X.-K. Xu, "Community detection based on symbiotic organisms search and neighborhood information," *IEEE Transactions on Computational Social Systems*, vol. 6, no. 6, pp. 1257–1272, 2019.

[31] Y. Atay, I. Koc, I. Babaoglu, and H. Kodaz, "Community detection from biological and social networks: a comparative analysis of metaheuristic algorithms," *Applied Soft Computing*, vol. 50, pp. 194–211, 2017.

[32] P. Holme, "Network reachability of real-world contact sequences," *Physical Review E*, vol. 71, no. 2, Article ID 046119, 2005.

[33] W. K. Cui, K. K. Shang, Y. J. Zhang et al., "Constructing null networks for community detection in complex networks," *The European Physical Journal B*, vol. 91, no. 7, p. 145, 2018.

[34] P. Mahadevan, D. Krioukov, K. Fall, and A. Vahdat, "Systematic topology analysis and generation using degree correlations," in *Proceedings of the 2006 conference on Applications, technologies, architectures, and protocols for computer communications (SIGCOMM'06)*, pp. 135–146, ACM Press, New York, NY, USA, 2006.

[35] A. Clauset, M. E. Newman, and C. Moore, "Finding community structure in very large networks," *Physical Review E*, vol. 70, no. 2, Article ID 066111, 2004.

[36] W. W. Zachary, "An information flow model for conflict and fission in small groups," *Journal of Anthropological Research*, vol. 33, no. 4, pp. 452–473, 1977.

[37] M. Rubinov and O. Sporns, "Complex network measures of brain connectivity: uses and interpretations," *NeuroImage*, vol. 52, no. 3, pp. 1059–1069, 2010.

[38] D. Gustavo, S. Mario, and J. Viktor, "How anatomy shapes dynamics: a semi-analytical study of the brain at rest by a simple spin model," *Frontiers in Computational Neuroscience*, vol. 6, p. 68, 2012.

[39] D. J. Watts and S. H. Strogatz, "Collective dynamics of "small-world" networks," *Nature*, vol. 393, no. 6684, pp. 440–442, 1998.

[40] M. Boguñá, R. Pastor-Satorras, A. Díaz-Guilera, and A. Arenas, "Models of social networks based on social distance attachment," *Physical Review E*, vol. 70, no. 5, Article ID 056122, 2004.

[41] M. Rosvall and C. T. Bergstrom, "Maps of random walks on complex networks reveal community structure," *Proceedings of The National Academy of Sciences*, vol. 105, no. 4, pp. 1118–1123, 2008.

[42] G. Jia, Z. Cai, M. Musolesi, and Y. Wang, *Community Detection in Social and Biological Networks Using Differential Evolution*, Springer, Berlin, Germany, 2012.

[43] G. Palla, I. Derényi, I. Farkas, and T. Vicsek, "Uncovering the overlapping community structure of complex networks in nature and society," *Nature*, vol. 435, no. 7043, pp. 814–818, 2005.

[44] M. E. Newman and M. Girvan, "Finding and evaluating community structure in networks," *Physical Review E*, vol. 69, no. 2, Article ID 026113, 2004.

[45] U. N. Raghavan, R. Albert, and S. Kumara, "Near linear time algorithm to detect community structures in large-scale networks," *Physical Review E*, vol. 76, no. 2, Article ID 036106, 2007.

[46] M. E. J. Newman, "Finding community structure in networks using the eigenvectors of matrices," *Physical Review E*, vol. 74, no. 3, Article ID 036104, 2006.

[47] M. E. J. Newman, "Analysis of weighted networks," *Physical Review E*, vol. 70, no. 5, Article ID 056131, 2004.

[48] L. Danon, A. Dazguilera, J. Duch et al., "Comparing community structure identification," *Journal of Statistical Mechanics-Theory and Experiment*, vol. 2005, no. 9, Article ID P09008, 2005.

[49] V. D. Blondel, J. L. Guillaume, R. Lambiotte et al., "Fast unfolding of communities in large networks," *Journal of Statistical Mechanics-Theory and Experiment*, vol. 2008, no. 10, Article ID P10008, 2008.

[50] P. Pons and M. Latapy, "Computing communities in large networks using random walks," *Journal of Graph Algorithms and Applications*, vol. 10, no. 2, pp. 191–218, 2006.

[51] B. Tillman, A. Markopoulou, C. T. Butts et al., "Construction of directed 2K graphs," in *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, ACM Press, New York, NY, USA, pp. 1115–1124, 2017.

[52] A. Amelio and C. Pizzuti, "An evolutionary and local refinement approach for community detection in signed networks," *International Journal of Artificial Intelligence Tools*, vol. 25, no. 4, Article ID 1650021, 2016.