

# SCIENTIFIC REPORTS

OPEN

## Cut Based Method for Comparing Complex Networks

Qun Liu<sup>1</sup> , Zhishan Dong<sup>1</sup> & En Wang<sup>2</sup>

Received: 8 September 2017

Accepted: 6 February 2018

Published online: 23 March 2018

Revealing the underlying similarity of various complex networks has become both a popular and interdisciplinary topic, with a plethora of relevant application domains. The essence of the similarity here is that network features of the same network type are highly similar, while the features of different kinds of networks present low similarity. In this paper, we introduce and explore a new method for comparing various complex networks based on the cut distance. We show correspondence between the cut distance and the similarity of two networks. This correspondence allows us to consider a broad range of complex networks and explicitly compare various networks with high accuracy. Various machine learning technologies such as genetic algorithms, nearest neighbor classification, and model selection are employed during the comparison process. Our cut method is shown to be suited for comparisons of undirected networks and directed networks, as well as weighted networks. In the model selection process, the results demonstrate that our approach outperforms other state-of-the-art methods with respect to accuracy.

Networks, in particular complex networks, appear as domain structures in a wide range of domains<sup>1–5</sup>, including computer science, biology and sociology<sup>6–9</sup>, *etc.* In most real-world cases, networks are directed, in the sense that the edge between two nodes presents not only the connectivity but also the directionality. For example, in a power grid network<sup>10</sup>, electricity flows directly from one node to the other; in a gene regulatory network<sup>11</sup>, one gene transports the transcription of the other gene, which is obviously directed.

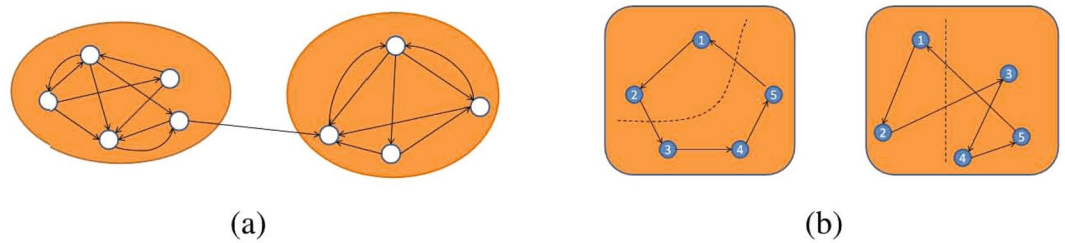
A central topic in real-world network analysis is the network clustering problem. Because clustering is widely used for studying the structure of various networks, the network clustering problem has attracted much research devoted to statistical and computational issues. Technically, there are two different domains in the network clustering problem. The first domain aims to detect communities in a single network according to a group of regulations (illustrated in Fig. 1(a)), for example, edge density or topology structure, so it is also called the community detection problem; see<sup>12–15</sup> and the citations therein for more details in this area. On the other hand, the goal of the second domain which is the focus of this paper is to compare a set of diverse networks (illustrated in Fig. 1(b)), regarding them as individual objects<sup>16</sup>. However, the network comparing problem is for the case of undirected networks, and the research on directed networks is still in its infancy, although it is evident that comparing directed networks has many significant applications in different research regimes.

It is common sense that the problem of clustering in directed networks is more challenging than in undirected networks. A common way to deal with directed networks is to simply ignore the edge directionality, in other words, regard the directed networks as undirected and then apply the algorithms for the undirected settings. However, in many situations, this simplistic approach is not satisfactory since the underlying edge directionality needs to be considered (e.g., the directional structure in a citation network between scientific publications). To maintain the information of directionality, in<sup>17</sup>, the author also converts the directed network into an undirected network, but the information about edge directionality is retained via weights on the edges.

The second category of methods is based on spectral graph theory. It is common sense that the adjacency matrix and the Laplacian matrix contain tremendous structural information about graphs<sup>18</sup>. There are many related works and surveys that refer to community detection in directed networks on the basis of spectral methods<sup>19–21</sup>, unfortunately, there are no related works that compare directed networks in the literature.

Among the recent developments in the successful graph comparison algorithms, kernel-based methods<sup>22</sup> from the machine learning community are the most valuable. The attractiveness of kernel based methods is due to the fact that they can compare networks in polynomial time. In recent years, various graph kernels, which focus on various types of substructures, have been developed, such as subtrees<sup>23</sup>, cycles<sup>24</sup>, random walks<sup>25</sup>, and shortest

<sup>1</sup>Mathematics School and Institute, Jilin University, Changchun, 130012, China. <sup>2</sup>Department of Computer Science and Technology, Jilin University, Changchun, 130012, China. Correspondence and requests for materials should be addressed to E.W. (email: [wangen@jlu.edu.cn](mailto:wangen@jlu.edu.cn))



**Figure 1.** (a) Density-based community detection in one directed network. (b) Comparing two directed networks. The dot line represents the cut which separates the node set  $V = \{1, 2, 3, 4, 5\}$  into two parts,  $S = \{1, 2\}$  and  $S^c = \{3, 4, 5\}$ ; then,  $e_G(S, S^c) = e_{G'}(S, S^c) = 2$ . Run all the subsets of  $V$ ,  $d_{\square}(G, G') = 0$ .

paths<sup>26</sup>. The other line of existing research has concentrated on graphs in specified structures, such as trees<sup>27</sup> and strings<sup>28</sup>. Among these algorithms, one can simply and naturally apply the kernel based methods utilized in undirected graphs to directed graphs, just as<sup>29</sup> did. Unfortunately, there are two unsolved problems in kernel based methods: The first is the problem of choosing the appropriate kernels that can capture network similarity: this is still far from being answered; the other obstacle is that computing any kernel function that is capable of fully recognizing the structure of networks (using subgraph-isomorphism) has been proven to be NP-hard<sup>30</sup>, making it impossible to find efficient kernels.

Apart from the aforementioned methodologies, some research achieves its goals by optimizing various clustering objective functions. A large number of approaches that regard the undirected graph clustering problem as an optimal problem have been presented in recent years, such as modularity<sup>31</sup> and normalized cut<sup>32</sup>. Recently, some works have tried to extend these measures for directed networks, such as directed vision of modularity<sup>33</sup> and objective function of weighted cuts in directed networks<sup>34</sup>. In this paper, we develop a framework for comparing directed networks via optimizing an objective function called cut distance<sup>35,36</sup>.

The starting point in this paper is similar to that of<sup>34</sup>. In<sup>34</sup>, the authors utilized the cut-based method to solve the community detection problem in one directed network; our focus here is to provide a network comparison approach based on the cut distance. In this framework, we will explore and verify this novel network comparison approach via machine learning technologies such as genetic algorithms and model selection. Our cut-based approach to comparing various artificial networks, as well as chemical molecule networks and wild female African elephant dominance networks in the real world, proved to be well suited for comparing undirected networks, directed networks and weighted networks. The corresponding simulations of the model selection process show that our approach works well in comparing directed networks with much higher accuracy than other state-of-the-art methods.

## Results

To illustrate that our framework has strongly mathematical guarantees, in this section, we provide the basic terminology notations that are frequently used in this paper. Although these notations show some differences between dense graphs and sparse graphs, they have no effect on our task, so we only use the terminology for dense graphs; there is no difference if you prefer to use those of sparse settings.

**Graph theory.** A graph  $G = (V(G), E(G))$  consists of a set of nodes  $V(G)$  and a set of edges  $E(G)$  that connect pairs of nodes. In a directed graph  $G = (V(G), E(G))$ , every edge  $(i, j) \in E(G)$  represents a connection from node  $i$  to  $j$ . An undirected graph can be seen as a directed graph where if edge  $(i, j) \in E(G)$ , then edge  $(j, i) \in E(G)$  as well. We also need some functions to describe the neighbors of a node  $V$  in a graph  $G$ :  $\delta_+(v) = \{(v, u) \in E(G)\}$  and  $\delta_-(v) = \{(u, v) \in E(G)\}$ . Here,  $\delta_+(v)$  and  $\delta_-(v)$  are called the outdegree and indegree of a node  $v$ , respectively. Furthermore, the maximal outdegree and indegree are denoted by  $\Delta^+ = \max\{|\delta_+(v)|, v \in V(G)\}$  and  $\Delta^- = \max\{|\delta_-(v)|, v \in V(G)\}$ , respectively.

In a weighted graph  $G$ , for each subset  $S, T \subset V(G)$ , define

$$e_G(S, T) = \sum_{i \in S, j \in T} \beta_{ij}(G), \quad (1)$$

where  $\beta_{ij}(G)$  is the associated edge weight for edge  $ij$ ; i.e., when  $G$  is unweighted,  $e_G(S, T)$  is the number of edges with one end in  $S$  and the other in  $T$  (illustrated in Fig. 1(b)).

Homomorphism density is a crucial definition in our framework:

**Definition 1:** Let  $F$  be simple graphs, and define  $\text{hom}(F, G)$  as the number of homomorphisms from  $F$  to  $G$ ; i.e., the number of adjacency preserving maps  $V(F) \rightarrow V(G)$ , and the homomorphism density of  $F$  in  $G$  is

$$t(F, G) = \frac{1}{|V(G)|^{|V(F)|}} \text{hom}(F, G). \quad (2)$$

Here,  $\text{hom}(F, G)$  can be computed via

D	(50,0.2)	(100,0.2)	(100,0.2)	(50,0.5)	(100,0.5)	(100,0.5)	(50,0.8)	(100,0.8)	(100,0.8)
(50,0.2)	0	0.0125	0.0138	0.2268	0.2173	0.2266	0.4412	0.4589	0.4600
(100,0.2)	0.0125	0	0.0102	0.2189	0.2172	0.2334	0.4511	0.4643	0.4757
(100,0.2)	0.0138	0.0102	0	0.2406	0.2374	0.2308	0.4713	0.4760	0.4678
(50,0.5)	0.2268	0.2189	0.2406	0	0.0172	0.0156	0.2548	0.2240	0.2305
(100,0.5)	0.2173	0.2172	0.2374	0.0172	0	0.0147	0.2419	0.2395	0.2390
(100,0.5)	0.2266	0.2334	0.2308	0.0156	0.0147	0	0.2360	0.2250	0.2266
(50,0.8)	0.4412	0.4511	0.4713	0.2548	0.2419	0.2360	0	0.0135	0.0127
(100,0.8)	0.4589	0.4643	0.4760	0.2240	0.2395	0.2250	0.0135	0	0.0101
(100,0.8)	0.4600	0.4757	0.4678	0.2305	0.2390	0.2266	0.0127	0.0101	0

**Table 1.** The cut distance matrix for directed Erdős-Rényi graphs with different number of nodes and parameters.

$$\text{hom}(F, G) = \sum_{\phi: V(F) \rightarrow V(G)} \prod_{i, j \in E(F)} \beta_{\phi(i), \phi(j)}(G), \quad (3)$$

where the sum runs over all maps from  $V(F)$  to  $V(G)$ .

**Cut distance.** In this paper, our framework is based on the cut distance (or rectangle distance) proposed by Frieze and Kannan<sup>37</sup>; we strongly recommend the celebrated works<sup>35–39</sup> if readers are interested in mathematical settings. The following is the explicit mathematical definition of the cut distance. First, we illustrate the cut distance between two graphs with the same set of nodes  $V$ ; the schematic illustration for unweighted graphs can be seen in Fig. 1(b).

**Definition 2:** For two graphs  $G$  and  $G'$  with the same set of nodes  $V$ ,

$$d_{\square}(G, G') = \max_{S \subseteq V} \frac{1}{|V|^2} |e_G(S, S^c) - e_{G'}(S, S^c)|, \quad (4)$$

where  $S^c = V \setminus S$ .

According to<sup>35</sup>, to calculate the cut distance between two labeled graphs  $G$  and  $G'$  on  $n$  and  $n'$  nodes, one can compute the cut distance between two new “blow-up graphs”  $G[n']$  and  $G'[n]$ . The  $k$ -fold blow-up of a graph  $G$  is the graph  $G[k]$  obtained from  $G$  by replacing each node by  $k$  independent nodes, and connecting two new nodes if and only if their originals were connected. If  $G$  is weighted, we define  $G[k]$  to be the graph on  $nk$  nodes labeled by pairs  $iu, i \in V(G), u = 1, \dots, k$ , with edge weights  $\beta_{iu, jv}(G[k]) = \beta_{ij}(G)$ . Note that when  $n'/n$  is an integer, it is sufficient to blow-up graph  $G$  to  $G[n'/n]$ .

The relationship between the cut distance and homomorphism density is from the following core theorem:<sup>35</sup>

**Theorem:** Let  $G$  and  $G'$  be two graphs; if  $F$  is a simple graph with  $m$  edges, then

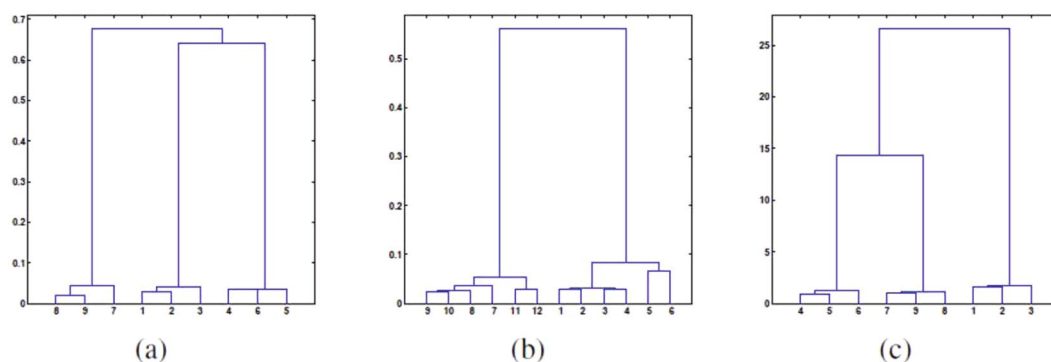
$$|t(F, G) - t(F, G')| \leq 4|E(F)|d_{\square}(G, G'). \quad (5)$$

Heuristically, if the cut distance between two graphs is small enough, then the homomorphism densities of these two graphs must have similar values for arbitrary small “structure”  $F$ ; in other words, these two graphs must have very similar structures. This is the theoretical basis for our framework.

**Cut distance measures conventional artificial networks.** In this subsection, two illustrative examples are utilized to show that the cut distance is very well suited for comparing conventional artificial networks—Erdős-Rényi graphs<sup>40</sup> with different parameters. Intuitively, two undirected Erdős-Rényi graphs with the same connection probability  $p$  will have similar topology structures, and the scale of the nodes of these two graphs contribute little to their dissimilarity. In our experiment, we utilize  $ER(n, p)$  to represent an undirected Erdős-Rényi instance with  $n$  nodes, and each unordered pair  $(i, j), i, j \in [n]$  is presented by an edge with connection probability  $p$ , independently of all the other edges. With respect to the directed case  $ER(n, p_1, p_2)$ , each pair  $(i, j)$ , if  $i < j$ , is presented by a directed edge from  $i$  to  $j$  with probability  $p_1$ , and a directed edge from  $j$  to  $i$  with probability  $p_2$ . Unlike the undirected case, two directed Erdős-Rényi graphs, for example,  $ER(n, p_1, p_2)$  and  $ER(n, p_2, p_1)$ , will exhibit dissimilarity due to the directionality within edges.

In Table 1, nine different independent undirected Erdős-Rényi graphs with different graphic parameters are generated: For convenience, we utilize the notation  $(n, p)$  to represent an undirected Erdős-Rényi instance with the number of nodes  $n$  and connection probability  $p$ . As seen from Table 1 and the corresponding cluster dendrogram in Fig. 2(a), different instances with the same connection probability  $p$  are well clustered, and the scale of the network contributes little to the clustering process.

In the directed case, we generated 12 independent directed Erdős-Rényi graphs with different parameters. Similarly, we use notation  $(n, p_1, p_2)$  to represent a directed Erdős-Rényi instance  $ER(n, p_1, p_2)$ . From Table 2 and the associated dendrogram plot in Fig. 2(b), it is plausible to cluster these 12 directed graphs into 4 groups: (1, 2,



**Figure 2.** Dendrogram plots for various networks. The labels in the x axis represent the corresponding instances in Tables 1–3 respectively. (a) represents the clustering process for undirected Erdős-Rényi graphs. (b) represents the clustering process for directed Erdős-Rényi graphs. (c) represents the clustering process for directed weighted Erdős-Rényi graphs.

3, 4), (5, 6), (7, 8, 9, 10), (11, 12). These results show that the cut-distance is a suitable measure to cluster directed graphs, and this simple implementation is also in accordance with our prediction.

In the following part, we demonstrate that the cut distance is capable of comparing weighted networks. We design the experiment as follows: First, we generate 9 directed Erdős-Rényi graphs with  $n = 100$ ,  $p_1 = 0.5$ ,  $p_2 = 0.5$ , and then in each generated graph, we add each edge weight independently according to a uniform distribution. We denote  $ER([0, a])$  a weighted graph within which each edge weight is assigned from a uniform distribution  $U[0, a]$ . As shown in Table 3 and the corresponding dendrogram plot in Fig. 2(c), the cut distance between two graphs in which edge weights are assigned from the same uniform distribution is much smaller than that from different distributions. This result demonstrates that our cut distance framework performs well in comparisons of directed weighted networks.

**Experiments on real networks.** *Chemical molecules networks.* We perform experiments on two different well-known undirected real network datasets, namely, MUTAG and PTC<sup>41</sup>. The MUTAG and PTC data sets are used to predict the toxicity of chemical molecules based on a comparison of their three-dimensional structure. The average number of nodes and edges per graph in the MUTAG data set are 17.72 and 38.76, respectively, and 26.70 and 52.06 in the PTC data set. We employ cut distance to measure the similarity of the molecules from these two data sets. In our implementation, we choose the NO 1, 2, 8, and 13 samples from the MUTAG data set and the NO 1, 3, 69, and 81 samples from the PTC data set.

A network sample is supposed to be similar to other samples from the same data set; for example,  $MUTAG_1$  is expected to be more similar to  $MUTAG_2$  than to  $PTC_1$ . Table 4 exhibits the pairwise cut distances for different MUTAG and PTC network samples. Table 4 and the corresponding dendrogram Fig. 3(a) show that cut distance measurements are consistent with the expectation: Cut distance calculates dissimilarities in such a way that networks of the same type are considered more similar to each other than to networks of different types.

*Wild female African elephant dominance networks.* In this part, 10 wild female African elephant dominance networks are compared. Each dominance network represents the dominance-subordinate relationship between adult females within 5 families each in Amboseli and Tarangire. The original data were collected by Archie *et al.*<sup>42</sup> in the form of a dominance matrix: Each row in the matrix represents one individual in the family, and the value of the intersection of each row (the aggressor) and column (the loser) shows the number of agonistic interactions won by the aggressor against the loser. In this work, we will use Appleby's<sup>43</sup> criteria for re-scoring in the dominance matrix. The criteria are widely used in predicting the linear dominance hierarchies in animal social networks<sup>44</sup>. Individuals that win in each dyad receive a score of 1 in their row at the column position of the subordinate, and losers receive a score of 0. If both individuals win equally or the win-lose relationship is unknown, each receives a score of 0.5 in its row-column position. After re-scoring, we are facing 10 directed weighted networks, where each network represents the dominance hierarchy structure within the wild female African elephant family. It is reasonable to believe that different elephant families in different districts (Amboseli and Tarangire) would have different dominance hierarchy structures, even if they are of the same specie and sexuality. In our experiment, we choose 5 families from Amboseli (AA, PC, EA, EB, OA), and 5 families from Tarangire (T, D, P, PH, SI); the corresponding data are available in Fig. 2 of Archie's work<sup>42</sup>. As shown in Table 5 and the associated dendrogram plot in Fig. 3(b), three clusters appears: (AA, PC, EA, EB, OA), (T, PH), (D, P, SI). Since the cut distances among T, D, P, PH are very small, i.e., the largest distance among these four families is 0.029, it is plausible to infer that the second cluster (T, PH) and the third cluster (D, P, SI) should be merged into one large cluster. It turns out that these results validate our prediction in addition to showing that our cut distance framework is well suited for comparing real directed weighted networks.

**Compare to other baseline methods.** In this subsection, we evaluate the performance of a model selection process on the basis of cut distance and compare it with other baseline methods in terms of prediction accuracy. Our baseline comparators are classic kernel-based methods<sup>29,30,45</sup>, here, we investigate four kinds of kernels:

D	(50,0.2,0.6)	(50,0.2,0.6)	(100,0.2,0.6)	(100,0.2,0.6)	(50,0.6,0.2)	(50,0.6,0.2)	(50,0.5,0.8)	(50,0.5,0.8)	(100,0.5,0.8)	(100,0.5,0.8)	(50,0.8,0.5)	(50,0.8,0.5)
(50,0.2,0.6)	0	0.0141	0.0157	0.0147	0.0392	0.0417	0.1967	0.1902	0.1951	0.1993	0.2090	0.1910
(50,0.2,0.6)	0.0141	0	0.0166	0.0154	0.0431	0.0446	0.1906	0.1956	0.1954	0.1975	0.2093	0.2088
(100,0.2,0.6)	0.0157	0.0166	0	0.0119	0.0467	0.0437	0.1779	0.1917	0.1938	0.2019	0.1997	0.1961
(100,0.2,0.6)	0.0147	0.0202	0.0142	0	0.0467	0.0489	0.1924	0.1966	0.1997	0.1971	0.1868	0.1941
(50,0.6,0.2)	0.0392	0.0431	0.0467	0.0452	0	0.0220	0.1930	0.2152	0.2032	0.2139	0.2134	0.2135
(50,0.6,0.2)	0.0417	0.0446	0.0437	0.0411	0.0220	0	0.1755	0.1815	0.1934	0.1910	0.1926	0.1848
(50,0.5,0.8)	0.1967	0.1906	0.1779	0.1924	0.1930	0.1755	0	0.0177	0.0163	0.0170	0.0319	0.0295
(50,0.5,0.8)	0.1902	0.1956	0.1917	0.1966	0.2152	0.1815	0.0177	0	0.0123	0.0162	0.0314	0.0294
(100,0.5,0.8)	0.1951	0.1954	0.1938	0.1997	0.2032	0.1934	0.0163	0.0123	0	0.0127	0.0296	0.0284
(100,0.5,0.8)	0.1993	0.1975	0.2019	0.1971	0.2139	0.1910	0.0170	0.0162	0.0127	0	0.0269	0.0307
(50,0.8,0.5)	0.2090	0.2093	0.1997	0.1868	0.2134	0.1926	0.0319	0.0314	0.0296	0.0269	0	0.0138
(50,0.8,0.5)	0.1910	0.2088	0.1961	0.1941	0.2135	0.1848	0.0295	0.0294	0.0284	0.0307	0.0138	0

**Table 2.** The cut distance matrix for directed Erdős-Rényi graphs with different number of nodes and parameters.

D	ER([0,100])	ER([0,100])	ER([0,100])	ER([0,50])	ER([0,50])	ER([0,50])	ER([0,25])	ER([0,25])	ER([0,25])
ER([0,100])	0	0.9216	0.9355	9.0712	9.1051	9.9368	14.4134	14.6252	14.3390
ER([0,100])	0.9216	0	0.9193	8.7676	9.2203	9.1469	14.3175	14.9862	14.0681
ER([0,100])	0.9355	0.9193	0	9.5209	9.7795	10.1581	14.2001	14.8115	15.0536
ER([0,50])	9.0712	8.7676	9.5209	0	0.4237	0.4238	5.0383	5.0746	5.2899
ER([0,50])	9.1051	9.2203	9.7795	0.4237	0	0.5373	5.0749	4.7845	4.9209
ER([0,50])	9.9368	9.1469	10.1581	0.4238	0.5373	0	5.2681	4.7529	5.0814
ER([0,25])	14.4134	14.3175	14.2001	5.0383	5.0749	5.2681	0	0.2260	0.2425
ER([0,25])	14.6252	14.9862	14.8115	5.0746	4.7845	4.7529	0.2260	0	0.2204
ER([0,25])	14.3390	14.0681	15.0536	5.2899	4.9209	5.0814	0.2425	0.2204	0

**Table 3.** The cut distance matrix for directed weighted Erdős-Rényi graphs with different edge weights.

D	MUTAG <sub>1</sub>	MUTAG <sub>2</sub>	MUTAG <sub>8</sub>	MUTAG <sub>13</sub>	PTC <sub>1</sub>	PTC <sub>3</sub>	PTC <sub>69</sub>	PTC <sub>81</sub>
MUTAG <sub>1</sub>	0	0.0027	0.0017	0.0015	0.0441	0.0405	0.0446	0.0482
MUTAG <sub>2</sub>	0.0027	0	0.0029	0.0028	0.0433	0.0395	0.0449	0.0482
MUTAG <sub>8</sub>	0.0017	0.0029	0	0.0016	0.0436	0.0415	0.0443	0.0457
MUTAG <sub>13</sub>	0.0015	0.0028	0.0016	0	0.0436	0.0407	0.0447	0.0465
PTC <sub>1</sub>	0.0441	0.0433	0.0436	0.0436	0	0.0044	0.0031	0.0036
PTC <sub>3</sub>	0.0405	0.0395	0.0415	0.0407	0.0044	0	0.0051	0.0077
PTC <sub>69</sub>	0.0446	0.0449	0.0443	0.0447	0.0031	0.0051	0	0.0038
PTC <sub>81</sub>	0.0482	0.0482	0.0457	0.0465	0.0036	0.0077	0.0038	0

**Table 4.** The cut distance matrix for the MUTAG and PTC data sets.

size 3 undirected graphlet kernels, size 3 directed graphlet kernels, size 4 undirected graphlet kernels, and random walks kernels. Figure 4(b) shows the accuracy of the proposed model selection process based on the cut distance along with the baseline methods. As Fig. 4(b) shows, our model selection process based on the cut distance outperforms traditional kernel-based methods with respect to accuracy.

## Methods

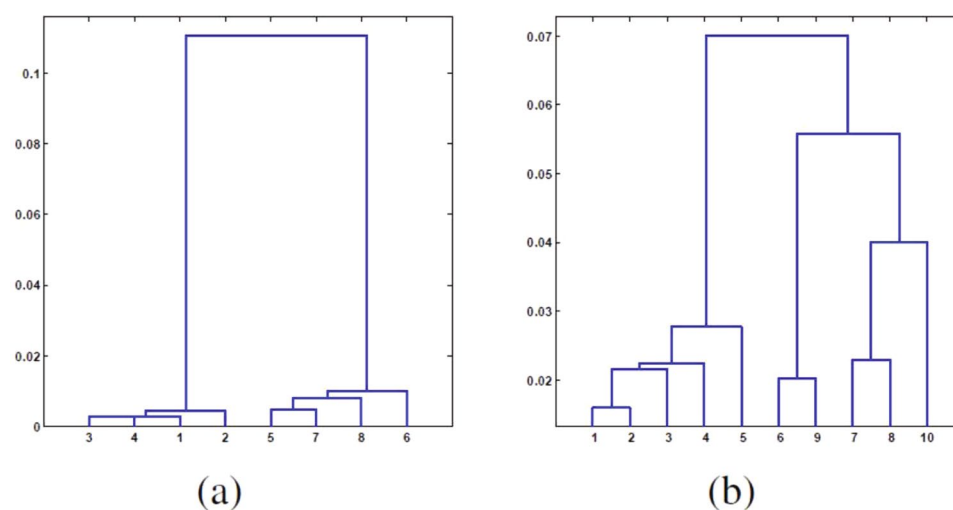
**Genetic algorithm.** Calculating the explicit cut distance between two graphs is nontrivial, especially for large graphs, since in principle the problem of optimizing over all possible subsets is NP-hard. In this framework, we utilize genetic algorithm<sup>46</sup> to solve this optimization problem.

A genetic algorithm is based on an imitation of the natural selection process; every candidate solution is represented by a chromosome which can be mutated and altered. Typically, solutions are represented by binary strings. In this framework, candidate solutions are different node sets; we transform node sets into a binary string as follows: First, we assume that graphs  $G$  and  $G'$  have the same node set  $V$ , for each node  $i \in V$ ,

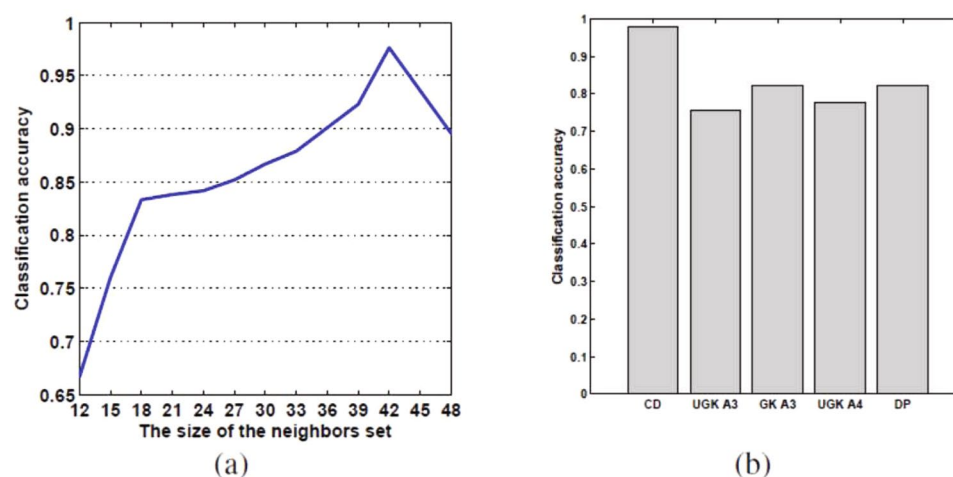
$$\omega(i) = \begin{cases} 1, & i \in S, \\ 0, & \text{otherwise.} \end{cases} \quad (6)$$

Distance	AA	PC	EA	EB	OA	T	D	P	PH	SI
AA	0	0.0103	0.0141	0.0102	0.0154	0.0546	0.0399	0.0411	0.0617	0.0300
PC	0.0103	0	0.0095	0.0122	0.0143	0.0549	0.0407	0.0395	0.0582	0.0307
EA	0.0141	0.0095	0	0.0142	0.0208	0.0547	0.0481	0.0513	0.0593	0.0361
EB	0.0102	0.0122	0.0142	0	0.0236	0.0624	0.0486	0.0482	0.0596	0.0373
OA	0.0154	0.0143	0.0208	0.0236	0	0.0525	0.0489	0.0417	0.0589	0.0313
T	0.0546	0.0549	0.0547	0.0624	0.0525	0	0.0290	0.0288	0.0107	0.0381
D	0.0399	0.0407	0.0481	0.0486	0.0489	0.0290	0	0.0145	0.0282	0.0165
P	0.0411	0.0395	0.0513	0.0482	0.0417	0.0288	0.0145	0	0.0228	0.0195
PH	0.0617	0.0582	0.0593	0.0596	0.0589	0.0107	0.0282	0.0228	0	0.0417
SI	0.0300	0.0307	0.0361	0.0373	0.0313	0.0381	0.0165	0.0195	0.0417	0

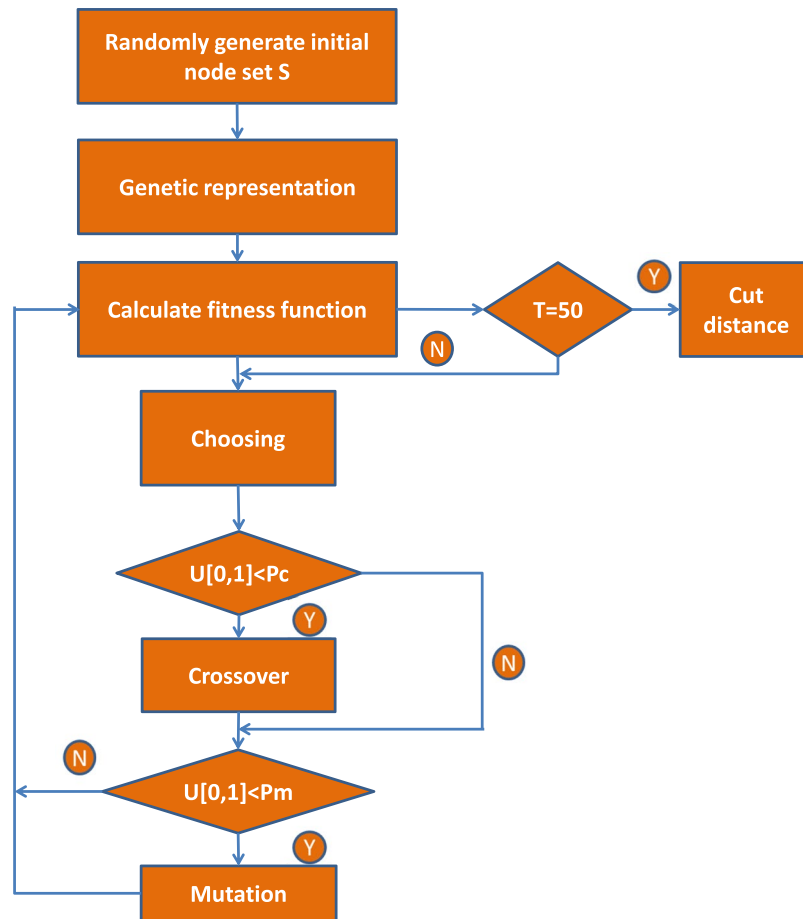
**Table 5.** The cut distance matrix for 10 wild female African elephant dominance networks.



**Figure 3.** Dendrogram plot for various real networks. The labels in the x axis represent the corresponding instances in Tables 4 and 5. (a) represents the clustering process for the MUTAG and PTC networks. (b) represents the clustering process for 10 wild female African elephant dominance networks.



**Figure 4.** (a) The effect of the neighbors set size on the accuracy of the model selection process on the basis of cut distance. (b) Classification accuracy based on various comparison methods. CD is the method based on cut distance. UGK Ak denotes the graphlet kernel method via the use of size k undirected graphlets. GK Ak denotes the graphlet kernel method via the use of size k directed graphlets. DP represents the directed product kernel method.



**Figure 5.** The genetic algorithm for cut distance. Here,  $U[0;1]$  represents a uniform distribution of interval  $[0;1]$ .

so we obtain a length  $n$  vector that works as the chromosome. In the process of the genetic algorithm, as in the natural selection process in the real world, only those excellent chromosomes remain over the long-time evolution, so we can obtain an optimizing result that approximates the explicit cut distance after the iterations. The related procedure of the genetic algorithm for cut distance can be seen in Fig. 5. In our framework, its fitness would then be given by the corresponding sum—which we are trying to maximize and the set  $S$  finally evolves, which results in the cut distance defined in Eq. (4).

For the case where  $G$  and  $G'$  have a different number of nodes, by constructing two new blow-up graphs  $G_{new}$ ,  $G'_{new}$  with the same node set, the cut distance can then be obtained by computing the distance between these two new blow-up graphs.

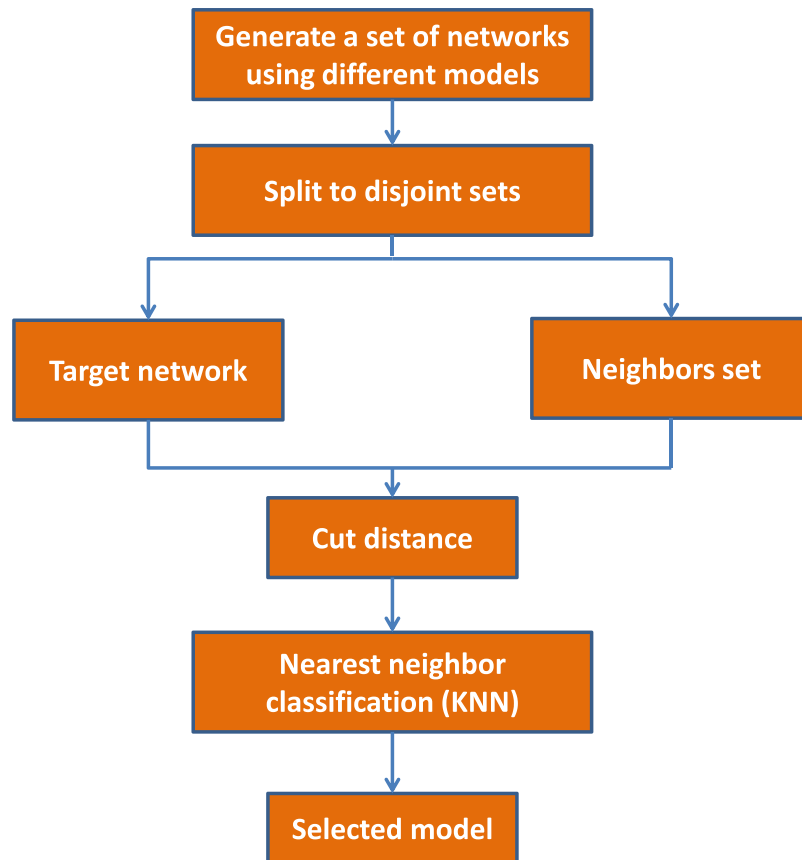
**Model selection.** The Model selection process is based on a network distance metric that can separate networks among different kinds of models. In this subsection, we give a description of the model selection process.

The corresponding model selection process can be seen in Fig. 6. In the first step, we generate a set of networks using various network models; each network instance is labeled by its corresponding model; *i.e.*, if we generate two types of networks, the instances of the first type are labeled 1, and the other instances are labeled 2. In the second step, choose one network instance as the target network; the other instances are regarded as the neighbors set; then, the target network is classified by a majority vote of its neighbors, with the target network being assigned to the class most common among its  $k$  nearest neighbors. If  $k = 1$ , the target network is simply assigned to the class of that single nearest neighbor. Run the second step until all the instances are classified. In essence, this procedure is the classic  $k$  nearest neighbor classification algorithm (KNN). The corresponding classification accuracy on the basis of a distance  $d$  is

$$\text{accuracy}(d) = \mathbb{P}(\text{knnclass}(i) = \text{trueclass}(i); i \in [n]), \quad (7)$$

and here,  $n$  is the number of generated networks.

**Implementation details.** Recall that in the first step of the genetic algorithm, each subset  $S$  is represented via a length  $n$  chromosome. In the following steps of the genetic algorithm, a new generation of the population is produced by using the crossover and mutation operations. The excellent (more fitting) genes from the parent



**Figure 6.** The proposed methodology for model selection.

chromosomes are inherited by descendants through crossover operation with probability  $P_c = 0.7$ . To maintain diversity, the mutation operation is utilized by changing each gene in a chromosome into an adverse gene (i.e.,  $1 \rightarrow 0$  or  $0 \rightarrow 1$ ) with probability  $P_m = 0.16$ . The maximum of iterations for the genetic algorithm is set to  $T = 50$ , and then the local optimal solution for the cut distance is the maximum of the 50 iterations.

In our implementation, the model selection is performed based on the KNN algorithm with  $k = 1$ ; in other words, each target network is assigned to the class of that nearest neighbor. We design the model selection process with 420 instances; that is, 420 different artificial networks with randomly chosen parameters are synthesized by means of the three described models (140 instances per model). Our evaluation includes 10 rounds. In each round, 10% of these generated instances are randomly averagedly selected; i.e., 14 instances for each model. These networks are used either in the neighbor set or as the target network (see Fig. 6). Each round includes 42 iterations; in each iteration, one instance out of the 42 directed artificial networks is chosen as the target network, and the remaining instances are utilized as the neighbors set. Accuracy is calculated as its average score in different rounds and iterations. In practice, as Fig. 4(a) shows, the accuracy of this model selection process increases with the size of the neighbors set and reaches a maximum value with 41 networks in the neighbors set. The experiment shows that having more than 41 networks in the neighbors results in no improvement in the accuracy of the model selection.

## Discussion

In this paper, we develop a framework for comparing networks on the basis of cut distance. This framework provides sufficient mathematical guarantees for comparing networks with high accuracy. In practice, we find our approach is suited for comparing undirected networks and works well on directed networks. Furthermore, on the basis of cut distance, we design a model selection process to classify various directed networks. More precisely, the major contributions include:

- Develop a novel network comparison framework on the basis of cut distance.
- Utilize genetic algorithm to evaluate the cut distance between two arbitrary networks.
- Compare various complex networks on the basis of cut distance.
- Design a model selection process to classify various directed networks.

Since in our framework, the cut distance between two graphs is evaluated by the genetic algorithm, the performance of our framework depends on the efficiency of the genetic algorithm. However, as is widely known, as the search space increases, the efficiency of the genetic algorithm decreases. As our experience shows, our framework

works well for comparing networks around thousands of nodes, but it is not capable of completing the task of comparing very large networks, such as networks with millions of nodes. To address this problem, one may try using other fast optimal algorithms to reevaluate the cut distance.

As noted in our research, the cut distance we investigate is only capable of comparing unlabeled networks. To break this ceiling, in future work, we will develop a network comparison method based on a new cut distance. First, we have a look at the definition:

**Definition** Given two graphs  $G_1$  and  $G_2$ , then the new cut distance is

$$\delta_{\square}(G_1, G_2) = \min_{G \cong G_2} d_{\square}(G_1, G). \quad (8)$$

Here we write  $G \cong G'$  if  $G$  and  $G'$  are isomorphic, i.e., if  $G'$  can be obtained from  $G$  by a relabeling of its nodes.

The reason we want to utilize this new cut distance can be summarized as follows:

- The new cut distance  $\delta_{\square}$  can be utilized in comparing unlabeled networks, while the cut distance  $d_{\square}$  can only deal with labeled networks.
- In<sup>35</sup>, the authors developed a rigorous mathematical theory based on the cut distance  $\delta_{\square}$ , and we can have better mathematical guarantees than those of  $d_{\square}$ .

The main obstacle is to evaluate the cut distance  $\delta_{\square}$ ; as far as we know, the graph isomorphic problem is NP-hard in computer science. Even if it is NP-hard, we strongly believe one may achieve this goal by optimizing various clustering objective functions.

**Artificial networks data description.** The utilized network generation models and the synthesized graphs of the artificial networks data set are described in the following:

*Directed Erdős-Rényi model.* This model generates random graphs with two specified density  $p_1$  and  $p_2$ . In our implementation,  $p_1$  is 0.08, and the other density parameter is chosen randomly from the range  $0.02 \leq p_2 \leq 0.1$ .

*Directed preferential attachment model.* In this model, we begin with an initial directed graph  $G(t_0)$  with  $t_0$  edges. For  $t \geq t_0$ , we generate directed graph  $G(t)$  according to the following three rules<sup>47</sup>.

- (A) With probability  $\alpha$ , add a new node  $v$  together with  $m_1$  edges from  $v$  to  $m_1$  existing nodes which are chosen according to  $D_i^{(in)}(t) + \delta_{in}$ .
- (B) With probability  $\beta$ ,  $m_2$  new edges are added between the existing nodes  $v_1, \dots, v_{m_2}$  and  $w_1, \dots, w_{m_2}$ , where  $v_1, \dots, v_{m_2}$  and  $w_1, \dots, w_{m_2}$  are chosen independently,  $v_i$  is selected according to  $D_i^{(in)}(t) + \delta_{in}$ , and  $w_i$  is selected according to  $D_i^{(out)}(t) + \delta_{out}$ .
- (C) With probability  $\gamma$ , we add a new node  $w$  and  $m_3$  edges from  $m_3$  existing nodes to  $w$ , where the chosen nodes are selected according to  $D_i^{(out)}(t) + \delta_{out}$ .

In our implementation, we fix  $\alpha = 0.3$ ,  $\beta = 0.5$  and  $\gamma = 0.2$ ;  $\delta_{in}$  and  $\delta_{out}$  are fixed as 1 and 2, respectively; the edge parameters  $m_1$  and  $m_2$  are fixed as 1 and 2, respectively; the range of edge parameter  $m_3$  in our evaluations is  $1 \leq m_3 \leq 5$ .

*Directed configuration model.* This model is configured by two degree sequences—in-degree and out-degree. In our implementation, each element of the in-degree and out-degree sequence is chosen randomly from the set  $\{2, 3, 4, 5, 6\}$  with identical probability  $1/5$ <sup>48</sup>.

**Baselines.** For comparison, we briefly introduce the traditional kernel-based algorithms.

*Graphlet kernels.* Literally, graphlets are the subgraphs with  $k$  nodes. Let  $\mathbb{G} = \{\text{graphlet}(1), \dots, \text{graphlet}(N_k)\}$  be the set of size- $k$  graphlets and  $G$  be a graph of size  $n$ . Define a vector  $f_G$  of length  $N_k$  whose  $i$ -th component corresponds to the frequency of occurrence of  $\text{graphlet}(i)$  in  $G$ .  $f_G$  is called the  $k$ -spectrum of  $G$ . In order to account for differences in size of the graphs, we normalize the counts to probability vectors<sup>45</sup>:

$$D_G = \frac{1}{|\text{all graphlets in } G|} f_G. \quad (9)$$

Given two graphs  $G$  and  $G'$  of size  $n \geq k$ , the graphlet kernel  $k_g$  is then defined as

$$k_g(G, G') = D_G^T D_{G'}. \quad (10)$$

In our experimental setting, we compute all undirected size-3, size-4 graphlet kernels and directed size-3 graphlet kernel<sup>29</sup>.

*Direct product kernel.* Let  $G_1(V_1, E_1), G_2(V_2, E_2)$  be two graphs; let  $A_{\times}$  denote the adjacency matrix of their direct product  $G_1 \times G_2$ ; here,  $G_1 \times G_2$  is a graph with vertex set<sup>30</sup>:

$$V_{\times} = \{(v_i, v'_r): v_i \in V_1, v'_r \in V_2\}, \quad (11)$$

and edge set

$$E_{\times} = \{((v_i, v'_r), (v_j, v'_s)): (v_i, v_j) \in E_1 \wedge (v'_r, v'_s) \in E_2\}. \quad (12)$$

In other words,  $G_1 \times G_2$  is a graph over pairs of vertices from  $G_1$  and  $G_2$ , and two vertices in  $G_1 \times G_2$  are neighbors if and only if the corresponding vertices in  $G_1$  and  $G_2$  are both neighbors. With a sequence of weights  $\lambda = \lambda_0, \lambda_1, \dots$  ( $\lambda_i \in \mathbb{R}$ ;  $\lambda_i \geq 0$  for all  $i \in \mathbb{N}$ ) the direct product kernel is defined as

$$k_{\times}(G_1, G_2) = \sum_{i,j=1}^{|V_{\times}|} \left[ \sum_{n=0}^{\infty} \lambda_n A_{\times}^n \right]_{ij} \quad (13)$$

if the limit exists.

In our implementation, to make sure the kernel Eq. (13) exists, we let  $\{\lambda_i\}$  be a geometric series, that is  $\lambda_i = \gamma^i$ ; here,  $\gamma = \frac{1}{a+1}$ , where  $a = \min\{\Delta^+(G_1 \times G_2), \Delta^-(G_1 \times G_2)\}$ .

## References

- Barabási, A. & Albert, R. Emergence of scaling in random networks. *Science* **286**, 509–512 (1999).
- Watts, D. & Strogatz, S. Collective dynamics of 'small-world' networks. *Nature* **393**, 440–442 (1998).
- Newman, M. The structure and function of complex networks. *SIAM Review* **167**–256 (2003).
- Vidal, C. M. M. & Barabási, A. Interactome networks and human disease. *Cell* **144**, 986–998 (2011).
- Fortunato, S. Community detection in graphs. *Phys. Rep.* **486**, 75–174 (2010).
- Barabási, A. L. The new science of networks. *Amer. J. Phys.* **71**, 409–410 (2004).
- Dorogovtsev, S. N. & Mendes, J. F. F. Evolution of networks. *Adv. Phys.* **51**, 1079–1187 (2002).
- Dorogovtsev, S. N. & Mendes, J. F. F. *Evolution of networks: from biological nets to the internet and WWW*. (Oxford Univ. Press 2004).
- Boccaletti, S., Latora, V., Moreno, Y., Chavez, M. & Hwang, D. U. Complex networks: structure and dynamics. *Phys. Rep.* **424**(4–5), 175–308 (2006).
- Pagani, G. A. & Aiello, M. The power grid as a complex network: a survey. *Phys. Rep.* **392**, 2688–2700 (2013).
- Davidson, E. H. The regulatory genome: gene regulatory networks in development and evolution. (Elsevier/Academic Press 2006).
- Fragkiskos, M. D. & Vazirgiannis, M. Clustering and community detection in directed networks: A survey. *Phys. Rep.* **533**, 95–142 (2013).
- Van Dongen, S. M. *Graph Clustering by Flow Simulation*. Ph.D. thesis, University of Utrecht, The Netherlands (2000).
- Brandes, U., Gaertler, M. & Wagner, D. Experiments on graph clustering algorithms. *11th Annual European Symposium on Algorithms*. **2832**, 568–579 (2003).
- Kim, Y., Son, S. W. & Jeong, H. Finding communities in directed networks. *Phys. Rev. E* **81**, 016103 (2010).
- Aggarwal, C. C. & Wang, H. *A survey of clustering algorithms for graph data*. (Springer Science+Business Media 2010).
- Satuluri, V. & Parthasarathy, S. Symmetrizations for clustering directed graphs. In *Proceedings of the 14th International Conference on Extending Database Technology, EDBT*. **11**, 343–354 (2011).
- Nascimento, M. C. V. & Carvalho, A. C. P. L. F. Spectral methods for graph clustering—a survey. *European J. Oper. Res.* **211** (2011).
- Zhou, D., Hunag, J. & Schölkopf, B. Learning from labeled and unlabeled data on a directed graph. In *Proceedings of the 22nd International Conference on Machine Learning, ICML*. **05**, 1036–1043 (2005).
- Chung, F. Laplacians and the cheeger inequality for directed graphs. *Ann. Comb.* **9**, 1–19 (2005).
- Luxburg, U. A tutorial on spectral clustering. *Stat. Comput.* **17**(4), 395–416 (2007).
- Schölkopf, B. & Smola, A. J. *Learning with kernels*. (MIT Press 2002).
- Ramon, J. & Gärtner, T. Expressivity versus efficiency of graph kernels. In *First International Workshop on Mining Graphs, Trees and Sequences* (2003).
- Horvath, T., Gärtner, T. & Wrobel, S. Cyclic pattern kernels for predictive graph mining. *KDD*. 158–167 (2004).
- Kashima, H. & Inokuchi, A. Kernels for graph classification. In *ICDM Workshop on Active Mining* (2002).
- Borgwardt, K. M. E. A. Protein function prediction via graph kernels. *Bioinformatics*. **21**, i47–i56 (2005).
- Collins, M. & Duffy, N. *Convolution kernels for natural language*, vol. 14 (Advances in Neural Information Processing Systems, MIT Press 2010).
- Lodhi, H., Saunders, C., Taylor, J. S., Cristianini, N. & Watkins, C. Text classification using string kernels. *Journal of Machine Learning Research*. **2** (2002).
- Aparicio, D., Ribeiro, P. & Silva, F. Network comparison using directed graphlets. ArXiv: 1511.01964.
- Gärtner, T., Flach, P. & Wrobel, S. On graph kernels: Hardness results and efficient alternatives. *Learning Theory and Kernel Machines, Lecture Notes in Computer Science*. **2777**, 129–143 (2003).
- Newman, M. Modularity and community structure in networks. *Proc. Natl. Acad. Sci.* **103**(23), 8577–8582 (2006).
- Shi, J. & Malik, J. Normalized cuts and image segmentation. *IEEE Trans. Pattern Anal. Mach. Intell.* **22**(8), 888–905 (2000).
- Leicht, E. A. & Newman, M. Community structure in directed networks. *Phys. Rev. Lett.* **100**, 118703 (2008).
- Meilă, M. & Pentney, W. Clustering by weighted cuts in directed graphs. In *Proceedings of the 2007 SIAM International Conference on Data Mining, SDM07*, 135–144 (2007).
- Borgs, C., Chayes, J. T., Lovász, L., Sós, V. T. & Vesztegombi, K. Convergent sequences of dense graphs i: Subgraph frequencies, metric properties and testing. *Adv. Math.* **219**, 1801–1851 (2008).
- Borgs, C., Chayes, J. T., Lovász, L., Sós, V. T. & Vesztegombi, K. Convergent sequences of dense graphs ii. multiway cuts and statistical physics. *Ann. Math.* **176**(1), 151–219 (2012).
- Frieze, A. & Kannan, R. Quick approximation to matrices and applications. *Combinatorica*. **19**, 175–220 (1999).
- Bollobás, B. & Riordan, O. Sparse graphs: metrics and random models. *Random Struct. Alg.* **39**(1), 1–38 (2011).
- Lovász, L. & Szegedy, B. Limits of dense graph sequences. *Journal of Combinatorial theory series B*. **96**, 933–957 (2006).
- Erdős, P. & Rényi, A. On the evolution of random graphs. *Publ. Math. Inst. Hungar. Acad. Sci.* **5**, 17–61 (1960).
- Vishwanathan, S. V. N., Schraudolph, N. N., Kondor, R. & Borgwardt, K. M. Graph kernels. *Journal of Machine Learning Research*. **11**, 1201–1242 (2010).
- Archie, E. A., Morrison, T. A., Foley, C. A. H., Moss, C. J. & Alberts, S. C. Dominance rank relationships among wild female african elephants, *loxodonta africana*. *Animal Behaviour* **71**, 117–127 (2006).
- Appleby, M. C. The probability of linearity in hierarchies. *Animal Behaviour* **31**, 600–608 (1983).
- Shizuka, D. & McDonald, D. B. A social network perspective on measurements of dominance hierarchies. *Animal Behaviour* **83**, 925–934 (2012).

45. Shervashidze, N., Vishwanathan, S. V. N., Petri, T., Mehlhorn, K. & Borgwardt, K. M. Efficient graphlet kernels for large graph comparison. *AISTATS*. 5, 488–495 (2009).
46. Goldberg, D. E. *Genetic algorithms in search, optimization and machine learning* (Addison-Wesley Longman Publishing 1989).
47. Bollobás, B., Borgs, C., Chayes, J. & Riordan, O. Directed scale-free graphs. In *Proceedings of the Fourteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, 132–139 (Baltimore 2003).
48. Chen, N. & Olvera-Cravioto, M. Directed random graphs with given degree distributions. *Stoch. Syst.* 3(1), 147–186 (2013).

## Acknowledgements

This work is supported by the National Natural Science Foundation of China (No 11371169, 11671168).

## Author Contributions

Q.L. and Z.D. conceived and designed the experiments. Q.L. performed the experiments. Q.L. and E.W. analysed the data and improved the methods. Q.L. and E.W. wrote the manuscript. All authors reviewed the manuscript.

## Additional Information

**Competing Interests:** The authors declare no competing interests.

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



**Open Access** This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons license, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons license, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons license and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this license, visit <http://creativecommons.org/licenses/by/4.0/>.

© The Author(s) 2018