Graph Embedding Models for Community Detection

Yinan Chen*, Zhuanming Gao*, Dong Li+
South China University of Technology, Guangzhou 510006, China

***Abstract*—Graph embedding models, also known as network representation models, have been tried to be applied to community detection tasks. However, most existing graph embedding models are not specially designed for community detection tasks and thus may be incapable of revealing the community structures in networks well. To fill this gap, this paper proposes two novel graph embedding models, GEMod and GEMap, which are specially designed for community detection. The proposed methods try to optimize the modified modularity and two-level coding length while learning the nodes embedding, so that the learned nodes embedding can be better applied to detect community structures in networks. Experimental results show that the algorithms proposed are superior or comparable to other community detection algorithms based on graph embedding models. Besides, the nodes embedding generated by GEMod and GEMap are generally more compact and separable, which means that they are more suitable for clustering tasks.**

Keywords—community detection, graph embedding, clustering

I. INTRODUCTION

Many complex systems exist in the form of networks or can be modeled as networks, such as social networks, scientists collaboration networks, epidemic spreading networks and protein interaction networks. Community detection is an important task in the field of network analysis, which aims to reveal the community structures in networks. A community is generally defined as a group of nodes which are closely connected internally, while the connections between different community nodes are sparse.

The graph embedding task attempts to represent network nodes with low-dimensional continuous vectors and simultaneously capture the structural information of the network. Graph embedding can provide effective input for downstream machine learning tasks, such as node classification [1], link prediction [2] and graph visualization [3]. With the gradual maturity of graph embedding, some scholars try to apply it to community detection tasks [4][5]. However, most existing graph embedding models are not designed for community detection, so they may not be able to effectively detect the community structures in networks.

Inspired by [6] and [7], we modify the definition of modularity and two-level coding length by using the nodes embedding, and propose the GEMod and GEMap graph embedding models. Same as DeepWalk [8] model, GEMod and GEMap are both based on random walk, but they take the community structure into consideration while learning the nodes embedding, so that the learned nodes embedding can be better applied to detect the communities in networks. Specifically, the GEMod model will try to optimize the modified modularity, and the GEMap model will try to optimize the modified coding length. Experimental results show that our methods can generally generate more compact and separable nodes embedding as shown in Fig. 1.

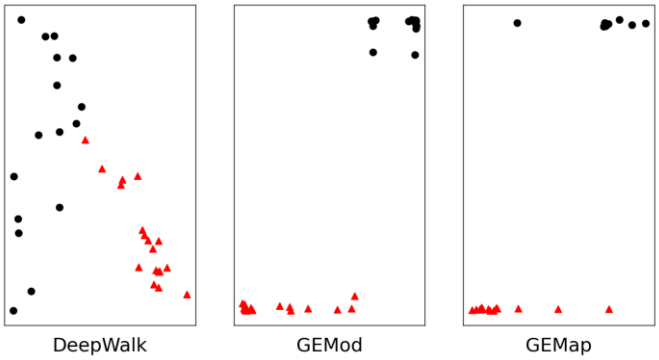

Figure 1. Node embeddings of Karate Club network. Different colors represent different community nodes.

The contributions of this paper are summarized as follows:

- Based on nodes embedding, a modified definition of modularity and two-level coding length are proposed.
- The modified community structure metrics are explicitly introduced into the graph embedding models, so that the learned nodes embedding can be better applied to the community detection tasks.
- The methods proposed can achieve more compact and divisible clustering results.

II. RELATED WORK

A. Community Detection

Newman et al. first introduced the definition of modularity [6] and used it as the evaluation metric of community partition. Specifically, the modularity is defined as follows:

* These authors have contributed equally to this work.
+ Corresponding Author. E-mail: cslidong@scut.edu.cn

DOI reference number: 10.18293/SEKE2022-005

$$Q = \frac{1}{2m}\sum_{i,j}\left[\boldsymbol{A}_{i,j} - \frac{d_i d_j}{2m}\right]\delta(\boldsymbol{C}_i, \boldsymbol{C}_j) \quad (1)$$

where m denotes the number of edges of the network and $\boldsymbol{A}_{i,j}$ denotes the number of edges between node i and node j. d_i and $\boldsymbol{C}_i$ respectively denote the degree of node i and the community that node i is located in. $\delta(\boldsymbol{C}_i, \boldsymbol{C}_j)$ is the Kronecker delta, which equals to 1 if $\boldsymbol{C}_i$ is equal to $\boldsymbol{C}_j$, otherwise 0. Many subsequent community detection algorithms based on modularity optimization have also been proposed, such as [9][10].

Besides the optimization method based on modularity, community detection based on information theory is also a widely studied direction. [7][11] Among them, the Infomap algorithm regards community detection in networks as a problem of map creating, and holds that a good map needs to be well compressed, so that the length of each path in the map should be short as possible. The algorithm uses information entropy to represent the average path length, and proposes the idea of two-level coding to measure the average coding length of random walking in the network. Specifically, the coding length is defined as:

$$L(M) = qH(Q) + \sum_{i=1}^{m} p^i H(P^i) \quad (2)$$

where M represent the partition scheme, $H(Q)$ represents the average coding length between communities, and $H(P^i)$ represents the average coding length of community i, q represents the probability of jumping between different communities, and p^i represents the probability of staying inside community i.

B. Graph Embedding

Bryan et al. proposed the DeepWalk algorithm [8] based on natural language model. The basic idea is to apply the process of random walk for each node in the network to obtain node sequences, then regard each node as a word and node sequences as sentences. After that, based on the SkipGram [12] language model, the low-dimensional vector representation of each node is learned. [13][14][15]

In recent years, network representation algorithms based on graph neural networks have also been proposed, such as [16][17][18]. However, most of them are supervised learning models or semi-supervised learning models, while community detection is an unsupervised learning task. Consequently, these graph neural networks can not be directly applied to community detection in networks.

C. Graph Embedding and Community Detection

An intuitive way of community detection based on network representation is to obtain the nodes embedding of the network by applying some kind of graph embedding model, and then cluster the embeddings by a clustering algorithm [4][5], so as to achieve the goal of community detection. However, in such approach, the network representation process is independent of the node clustering process, and the network representation model cannot get feedback from the nodes clustering model.

In order to alleviate the above problem, the ComE [19] model combines node embedding, community embedding and community detection into a single process, so as to complement each other. However, it assumes that the community embedding obeys a multivariate Gaussian distribution. GEMSEC [20] model introduces a self-clustering process into the nodes embedding process, thus improving the clustering quality of nodes representation, but it does not explicitly introduce community structure metrics.

III. The Methods

A. Problem Definition

The methods mainly focus on detecting non-overlapping communities by using graph embedding methods, given an undirected and unweighted network $G = (V, E)$.

Definition 1 Non-overlapping Community Detection

Given a network $G = (V, E)$, non-overlapping community detection aims to divide V into K disjoint node subsets $\{P_i | P_i \subset V, P_i \cap P_j = \emptyset, i \neq j, i = 1, \ldots, K\}$, and $\cup P_i = V$, so that the nodes in each node subset share some kind of similarity, while different node subsets have great dissimilarity.

Definition 2 Graph Embedding

Given a network $G = (V, E)$, graph embedding models aim to find a mapping function $f: V \to \mathbb{R}^d$, so that the learned nodes embedding can effectively express the structural information of the network. d is the dimension of the embedding space. That is, the nodes are projected from discrete space to a continuous vector space.

B. Node Similarity

Given nodes $u, v \in V$ and mapping function f, let $\boldsymbol{h}_u = f(u)$ and $\boldsymbol{h}_v = f(v)$, $\boldsymbol{h}_u, \boldsymbol{h}_v \in \mathbb{R}^d$. Graph embedding models often use the softmax or sigmoid function to measure the similarity or adjacency probability of u and v. Nevertheless, nodes embedding will generally serve as the input of some kind of clustering model, and many clustering models usually uses Euclidean distance to measure the dissimilarity between different samples. The dissimilarity between node u and node v is defined as:

$$dissim(u, v) = \|\boldsymbol{h}_u - \boldsymbol{h}_v\|_2 \quad (3)$$

The opposite number of the dissimilarity is defined as the similarity measure between nodes:

$$sim(u, v) = -dissim(u, v) \quad (4)$$

C. GEMod Algorithm

The GEMod model includes two stages: embedding initialization and modified modularity optimization. Specifically, the algorithm firstly takes each node as the starting point to do multiple truncated random walks. The process of random walk can be regarded as the process of message propagation. Since the small world effect [21] generally exists in networks, the length of each walk is set to a value less than 6. After that, the nodes in the same walk sequence are regarded as the friend nodes, and let the friend nodes of node u be $F(u)$. It is assumed that the a node

and its friend nodes should have great similarity for they share some kind of characteristic. Similar to the SkipGram model, GEMod also performs negative sampling to obtain another set of node sequences, and takes the nodes in the sequence as stranger nodes of the source node, and let the stranger nodes of node u be $S(u)$. The negative sampling process of GEMod is the same as that of SkipGram model.

GEMod expects to maximize the similarity between node u and its friend nodes, and simultaneously maximize the dissimilarity between node u and its stranger nodes. Consequently, the loss function corresponding to the first stage is:

$$L_1 = -\left[\sum_{u\in V}\sum_{v\in F(u)} sim(u,v) + \sum_{u\in V}\sum_{v'\in S(u)} dissim(u,v')\right] \quad (5)$$

In order to make the node embeddings better reflect the community structures, and make the connections within community closer, while the connections between communities more sparse, a modified definition of modularity is proposed:

$$M = \sum_{u,v\in V} sim(u,v)\delta(\mathbf{C}_u,\mathbf{C}_v) + \sum_{i=1}^{K}\sum_{j=i+1}^{K} dissim(C^{i,0}C^{j,0}) \quad (6)$$

$$h^{i,0} = \frac{1}{|C^i|}\sum_{u\in C^i} h_u \quad (7)$$

where $\mathbf{C}_u$ represents the community to which the node u belongs, C^i represents the i -th community, $C^{i,0}$ represents the center of community i, $h^{i,0}$ is embedding of $C^{i,0}$, K is the number of communities, and $\delta(\mathbf{C}_u,\mathbf{C}_v)=1$ if $\mathbf{C}_u=\mathbf{C}_v$, otherwise, $\delta(\mathbf{C}_u,\mathbf{C}_v)=0$. Herein, we use k-means to cluster the nodes embedding to obtain the community partition of the network, and then calculate the modified modularity. It should be pointed out that other clustering methods are also feasible. The meaning of maximizing the above equation is to maximize the similarity of nodes within the same community and the dissimilarity between different community centers. Thus, the connections within communities are tight while the communities are far away from each other. As a result, the nodes embedding generated by GEMod model can get more compact and separable clusters. The loss function of the second stage is,

$$L_2^{mod} = L_1 - \alpha M \quad (8)$$

where α is a hyper-parameter used to balance the influence of M on the result.

In order to accelerate the convergence of the algorithm, GEMod will be trained for a certain number of rounds in the first stage, and then enter the second stage.

D. GEMap Algorithm

The GEMap algorithm is similar to the GEMod algorithm, but the second stage of GEMap tries to optimize the modified coding length instead of the modified modularity. The modified coding length also uses the idea of two-level coding, including coding within communities and coding between communities. However, unlike the Infomap algorithm, GEMap expects to minimize the coding length within communities and maximize the coding length between communities.

Suppose that there is a signal source in the center of each community, which is called a local signal source, and the nodes closer to the signal source have more opportunities to receive the message sent by the signal source. Therefore, the probability that a node receives a message by the distance between the node and the signal source can be measured. Specifically, for the community C^i, the distances between each node in the community and the community center $C^{i,0}$ are first calculated, then divided by the sum of all distances, and finally sorted in descending order to get the probability distribution p^i. $p^{i,1}$ represents the receiving probability of the nearest node from the community center, $p^{i,2}$ represents the receiving probability of the next nearest node from the community center, and so on. Actually, the average coding length of each community has nothing to do with the order of p^i, so the sorting process can be omitted.

Similarly, suppose that there is also a signal source in the center of the network composed of all community centers, which is called the global signal source, and then calculate the probability that each community center receives the message sent by the global signal source as described above, the average coding length between communities can be calculated.

Since we only focus on non-overlapping community detection in networks, we make an assumption that each signal source only produces messages belong to a specific topic, and each community is only interested in a specific topic, while different communities do not share the same interest. Thus, we expect to minimize the average coding length within communities and maximize the average coding length between communities. In summary, the average intra-community coding length of each community is defined as follows:

$$E_{intra} = -\sum_{i=1}^{K}\sum_{u\in C^i} p^{i,u}\log p^{i,u} \quad (9)$$

$$p^{i,u} = \frac{sim(C^{i,u},C^{i,0})}{\sum_{u\in C^i} sim(C^{i,u}C^{i,0})} \quad (10)$$

where $C^{i,u}$ represents the node u in community i. And the average inter-community coding length is defined as follows:

$$E_{inter} = -\sum_{i=1}^{K} q^i \log q^i \quad (11)$$

$$q^i = \frac{sim(C^{i,0},C^0)}{\sum_i^K sim(C^{i,0},C^0)} \quad (12)$$

$$h^0 = \frac{1}{K}\sum_{i}^{K} h^{i,0} \quad (13)$$

where $C^{i,0}$ is the center of community i, C^0 is the centroid of community centers, $h^{i,0}$ is the embedding of $C^{i,0}$, h^0 is the embedding of C^0 and K is the number of communities in the network. And the overall coding length is,

$$E = E_{intra} + E_{inter} \quad (14)$$

In summary, the loss function of the second stage of GEMap algorithm is,

$$L_2^{map} = L_1 + \beta E \tag{15}$$

where β is a hyperparameter used to balance the influence of E on the result.

E. Models Optimization

Both GEMod and GEMap models need to optimize the parameter set of $\mathbf{H} = \{h_u | u \in V\}$, and its size is $O(d|V|)$. Herein, we use the back-propagation algorithm to calculate the derivative of the loss function, and choose the Adam [22] optimizer to optimize the model parameters.

IV. Experimental Results and Analysis

A. Data-sets

In this paper, the effectiveness of the algorithms is verified on four real-world data-sets [23][24][25] and four LFR [26] synthetic data-sets. The specific network structure information of each data-set is shown in Table I. In which n represents the number of nodes, m represents the number of edges, k represents the number of ground-truth communities, d represents the average degree of nodes, and μ represents the mixing parameter for synthetic networks.

TABLE I. Main properties of the data-sets

Data-set	n	m	k	d	μ
Karate	34	78	2	4.6	-
Dolphin	62	162	2	5.1	-
Polbooks	105	441	3	10.7	-
Football	115	613	12	8.4	-
L1	1,000	15,304	49	15	0.3
L2	1,000	30,708	29	30	0.3
L3	1,000	15,206	49	15	0.5
L4	1,000	30,156	31	30	0.5

B. Comparison Algorithms

Here six graph embedding models are selected to compare with GEMod and GEMap algorithms, including DeepWalk [8], Node2vec [13], WALKLETS [14], LINE [15], ComE [19] and GEMSEC [20]. Specifically, the embeddings learned by these models are clustered using k-means, to obtain the community partition for a network.

C. Evaluation Metric

Because the data-sets have ground-truth community partition, normalized mutual information (NMI) [27] is used to measure the similarity between the partition output by algorithm and the ground-truth partition, which is defined as follows:

$$NMI = \frac{-2\sum_{i=1}^{C_A}\sum_{j=1}^{C_B} C_{ij}\log_2\left(C_{ij}N/C_{i.}C_{.j}\right)}{\sum_{i=1}^{C_A} C_{i.}\log_2(C_{i.}/N) + \sum_{j=1}^{C_B} C_{.j}\log_2\left(C_{.j}/N\right)} \tag{16}$$

where C_A and C_B respectively represents the community partition obtained by the algorithm and the ground-truth community partition, and C_A and C_B respectively represents the number of communities in partition C_A and partition B. C is the confusion matrix, and C_{ij} represents the number of nodes in the community i divided by C_A and also in the community j divided by B. $C_{i.}$ represents the sum of elements in the i -th row of the confusion matrix, $C_{.j}$ represents the sum of elements in the j -th column of the confusion matrix, and N is the total number of nodes in the network. The value range of NMI is $[0,1]$. The larger the NMI value, the closer the partition result obtained by the algorithm is to the ground-truth community partition.

D. Experimental Results

Each algorithm runs five times on each data-set, and finally take the average of the results. The comparison results of the algorithms are shown in Table II and Table III. The error of the experimental results is indicated in parentheses, which is measured by the standard deviation of the results.

TABLE II. Average NMI of each algorithm on real-world networks

Data-set	Karate	Dolphins	Polbooks	Football
DeepWalk	0.663 (±0.038)	0.817 (±0.048)	0.562 (±0.003)	0.925 (±0.001)
Node2vec	0.946 (±0.120)	0.874 (±0.033)	0.561 (±0.024)	0.927 (±0.002)
WALKLETS	0.869 (±0.073)	**0.889** (±0.000)	0.557 (±0.012)	0.927 (±0.000)
LINE	0.473 (±0.103)	0.322 (±0.131)	0.409 (±0.045)	0.852 (±0.020)
ComE	0.604 (±0.049)	0.453 (±0.019)	0.465 (±0.035)	0.691 (±0.117)
GEMSEC	0.226 (±0.000)	0.293 (±0.000)	0.103 (±0.000)	**0.930** (±0.000)
GEMod	**1.000** (±0.000)	**0.889** (±0.000)	0.568 (±0.007)	0.927 (±0.001)
GEMap	**1.000** (±0.000)	**0.889** (±0.000)	**0.573** (±0.009)	0.926 (±0.004)

TABLE III. AVERAGE NMI OF EACH ALGORITHM ON SYNTHETIC NETWORKS

Data-set	L1	L2	L3	L4
DeepWalk	0.977 (±0.007)	0.996 (±0.049)	0.959 (±0.010)	0.994 (±0.006)
Node2vec	0.974 (±0.007)	0.994 (±0.006)	0.939 (±0.009)	0.994 (±0.005)
WALKLETS	0.984 (±0.007)	0.992 (±0.005)	0.957 (±0.012)	0.994 (±0.006)
LINE	0.541 (±0.014)	0.772 (±0.024)	0.335 (±0.008)	0.241 (±0.013)
ComE	0.504 (±0.023)	0.599 (±0.037)	0.435 (±0.009)	0.559 (±0.034)
GEMSEC	0.884 (±0.000)	**1.000** (±0.000)	0.832 (±0.000)	0.996 (±0.000)
GEMod	**0.995** (±0.001)	**1.000** (±0.000)	0.959 (±0.005)	0.998 (±0.002)
GEMap	0.994 (±0.003)	**1.000** (±0.000)	**0.960** (±0.004)	**1.000** (±0.000)

The results show that in the real-world data-sets, except for Football data-set, GEMod and GEMap algorithms outperform other benchmark algorithms. On Football data-set, GEMod and GEMap algorithms are only 0.3% and 0.4% inferior to the best results respectively. In addition, both GEMod and GEMap algorithms have very small experimental errors, which shows the stability of the algorithms.

E. Parameters Analysis

In order to test the impact of hyper-parameters on the clustering effect, GEMod and GEMap are run with different hyper-parameters on Football data-set. The experimental results are shown in Fig. 2.

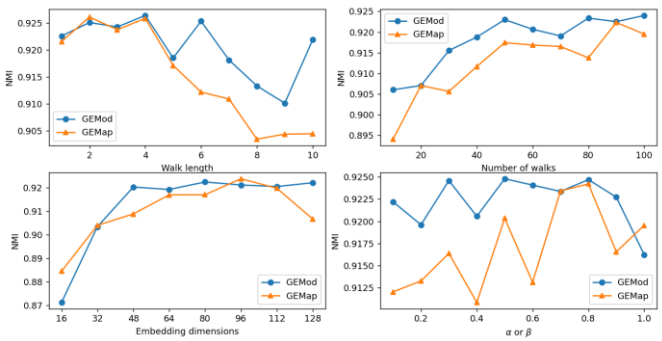

Figure 2. Influence of cluster quality to parameters changes measured by NMI

The random-walk length is set from 1 to 10 in consideration of the small-world effect [26]. The results show that when the random-walk length is between 2 and 4, the clustering performs best. Because the length of each random walk is short, in order to increase the data-set to get better fitting result, we increase the number of random-walk iterations made by each node here. Experimental results show that when the number of random walks is between 50 and 100, the clustering effect is better. In addition, in order to get a stable and better clustering effect, the dimension of nodes embedding should be between 48 and 128. Finally, when α is between 0.5 and 0.8, GEMod can generally get better clustering results, and when β is between 0.7 and 0.8, GEMap can achieve better clustering quality, besides β has unstable influence on the clustering results.

V. CONCLUSION

In this paper, two novel graph embedding models, GEMod and GEMap is proposed, which are customized for community detection tasks. The former uses the modified modularity, while the latter uses the modified coding length to optimize the community structure in the process of nodes embedding. Experimental results show that GEMod and GEMap are both superior to most community detection algorithms based on graph embedding models, and the nodes embedding generated by these models are generally more compact and separable.

REFERENCES

[1] Bhagat S, Cormode G, Muthukrishnan S. Node classification in social networks. In Social network data analytics. Springer, Boston, MA, 2011: 115-148.

[2] Liben‐Nowell D, Kleinberg J. The link‐prediction problem for social networks. *Journal of the American society for information science and technology*, 2007, 58(7): 1019-1031.

[3] Van der Maaten L, Hinton G. Visualizing data using t-SNE. *Journal of machine learning research*, 2008, 9(11)..

[4] Chen Y, Wang L, Qi D, W Zhang. Community detection based on deepwalk in large scale networks. International Conference on Big Data and Security. Springer, Singapore, 2019: 568-583.

[5] Hu F, Liu J, Li L, J Liang. Community detection in complex networks using Node2vec with spectral clustering. *Physica A: Statistical Mechanics and its Applications*, 2020, 545: 123633..

[6] Newman M E J. Modularity and community structure in networks. *Proceedings of the national academy of sciences*, 2006, 103(23): 8577-8582.

[7] Rosvall M, Bergstrom C T. Maps of random walks on complex networks reveal community structure. *Proceedings of the national academy of sciences*, 2008, 105(4): 1118-1123.

[8] Perozzi B, Al-Rfou R, Skiena S. Deepwalk: Online learning of social representations. In Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining. 2014: 701-710..

[9] Blondel V D, Guillaume J L, Lambiotte R, et al. Fast unfolding of communities in large networks. *Journal of statistical mechanics: theory and experiment*, 2008, 2008(10): P10008.

[10] Zhuang D, Chang J M, Li M. DynaMo: Dynamic community detection by incrementally maximizing modularity. *IEEE Transactions on Knowledge and Data Engineering*, 2019, 33(5): 1934-1945.

[11] Shen H, Cheng X Q, Chen H Q, Liu Y. Information bottleneck based community detection in network. *Chinese Journal of Computers (Chinese Edition)*, 2008, 31(4): 677.

[12] Mikolov T, Sutskever I, Chen K, Corrado G S, Dean J. Distributed representations of words and phrases and their compositionality. *Advances in neural information processing systems*, 2013, 26.

[13] Grover A, Leskovec J. node2vec: Scalable feature learning for networks. In Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining. 2016: 855-864..

[14] Perozzi B, Kulkarni V, Chen H, Skiena S. Don't Walk, Skip! Online learning of multi-scale network embeddings. In Proceedings of the 2017 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining. 2017: 258-265.

[15] Tang J, Qu M, Wang M, Zhang M, Yan J, Mei Q. Line: Large-scale information network embedding. In Proceedings of the 24th international conference on world wide web. 2015: 1067-1077.

[16] Kipf T N, Welling M. Semi-supervised classification with graph convolutional networks. arXiv preprint arXiv:1609.02907, 2016.

[17] Kipf T N, Welling M. Variational graph auto-encoders. arXiv preprint arXiv:1611.07308, 2016.

[18] Xu D, Ruan C, Korpeoglu E, Kumar S, Achan K. Inductive representation learning on temporal graphs. arXiv preprint arXiv:2002.07962, 2020..

[19] Cavallari S, Zheng V W, Cai H, Chang K C, Cambria E. Learning community embedding with community detection and node embedding on graphs. In Proceedings of the 2017 ACM on Conference on Information and Knowledge Management. 2017: 377-386..

[20] Rozemberczki B, Davies R, Sarkar R, Sutton C. Gemsec: Graph embedding with self clustering. In Proceedings of the 2019 IEEE/ACM international conference on advances in social networks analysis and mining. 2019: 65-72.

[21] Watts D J, Strogatz S H. Collective dynamics of 'small-world' networks. *Nature*, 1998, 393(6684): 440-442.

[22] Kingma D P, Ba J. Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980, 2014.

[23] Zachary W W. An information flow model for conflict and fission in small groups. *Journal of anthropological research*, 1977, 33(4): 452-473.

[24] Lusseau D, Schneider K, Boisseau O J, Haase P, Slooten E, Dawson S M. The bottlenose dolphin community of Doubtful Sound features a large proportion of long-lasting associations. *Behavioral Ecology and Sociobiology*, 2003, 54(4): 396-405.

[25] Girvan M, Newman M E J. Community structure in social and biological networks. *Proceedings of the national academy of sciences*, 2002, 99(12): 7821-7826..

[26] Lancichinetti A, Fortunato S. Benchmarks for testing community detection algorithms on directed and weighted graphs with overlapping communities. *Physical Review E*, 2009, 80(1): 016118..

[27] Danon L, Diaz-Guilera A, Duch J, Arenas A. Comparing community structure identification. *Journal of statistical mechanics: Theory and experiment*, 2005, 2005(09): P09008..