

Modularity-based representation learning for networks

Jialin He(何嘉林), Dongmei Li(李冬梅), and Yuexi Liu(刘阅希)
Citation: Chin. Phys. B, 2020, 29 (**12**): 128901. DOI: 10.1088/1674-1056/abbbec
Journal homepage: http://cpb.iphy.ac.cn; http://iopscience.iop.org/cpb

What follows is a list of articles you may be interested in

A uniform framework of projection and community detection for one-mode network in bipartite networks

Guolin Wu(吴果林), Changgui Gu(顾长贵), Lu Qiu(邱路), Huijie Yang(杨会杰)
Chin. Phys. B, 2017, 26 (**12**): 128901. DOI: 10.1088/1674-1056/26/12/128901

Community detection in signed networks based on discrete-time model

Jianrui Chen(陈建芮), Li Zhang(张莉), Weiwei Liu(刘维维), Zaizai Yan(闫在在)
Chin. Phys. B, 2017, 26 (**1**): 018901. DOI: 10.1088/1674-1056/26/1/018901

Detecting community structure using label propagation with consensus weight in complex network

Liang Zong-Wen, Li Jian-Ping, Yang Fan, Athina Petropulu
Chin. Phys. B, 2014, 23 (**9**): 098902. DOI: 10.1088/1674-1056/23/9/098902

Dynamic evolutionary community detection algorithms based on the modularity matrix

Chen Jian-Rui, Hong Zhi-Min, Wang Li-Na, Wu Lan
Chin. Phys. B, 2014, 23 (**11**): 118903. DOI: 10.1088/1674-1056/23/11/118903

An improvement of the fast uncovering community algorithm

Wang Li, Wang Jiang, Shen Hua-Wei, Cheng Xue-Qi
Chin. Phys. B, 2013, 22 (**10**): 108903. DOI: 10.1088/1674-1056/22/10/108903

Modularity-based representation learning for networks*

Jialin He(何嘉林)[1,3,†], Dongmei Li(李冬梅)[2], and Yuexi Liu(刘阅希)[1]

[1] School of Computer Science and Engineering, China West Normal University, Nanchong 637009, China
[2] Department of Scientific Research, China West Normal University, Nanchong 637009, China
[3] Internet of Things Perception and Big Data Analysis Key Laboratory of Nanchong, Nanchong 637009, China

(Received 17 July 2020; revised manuscript received 31 August 2020; accepted manuscript online 28 September 2020)

Network embedding aims at learning low-dimensional representation of vertexes in a network and effectively preserving network structures. These representations can be used as features for many complex tasks on networks such as community detection and multi-label classification. Some classic methods based on the skip-gram model have been proposed to learn the representation of vertexes. However, these methods do not consider the global structure (*i.e.*, community structure) while sampling vertex sequences in network. To solve this problem, we suggest a novel sampling method which takes community information into consideration. It first samples dense vertex sequences by taking advantage of modularity function and then learns vertex representation by using the skip-gram model. Experimental results on the tasks of community detection and multi-label classification show that our method outperforms three state-of-the-art methods on learning the vertex representations in networks.

Keywords: network embedding, low-dimensional representation, vertex sequences, community detection

PACS: 89.75.Hc, 89.75.–k, 64.60.aq **DOI:** 10.1088/1674-1056/abbbec

1. Introduction

In real world, many complex systems can be studied in the form of networks which are composed of nodes and edges. Many complex tasks, such as community detection[1,2] and multi-label classification,[3,4] can be performed on small networks quickly. However, as online social networks such as Facebook and Weibo expand rapidly, network data is constantly growing, posing a challenge in dealing with these large-scale data efficiently. How to represent network data are critical to solve the problem. Although a vertex can be represented by its corresponding row vector in the adjacency matrix of the network, the extremely high dimensionality of such a representation in a large network makes it difficult to perform some complex network tasks. In recent years, network representation learning (NRL)[5,6] has been widely adopted to network analysis, which aims to build low-dimensional vectors for vertices according to their structural roles in networks. With the learnt low-dimensional representations, NRL enables us to process large-scale network data by using parallel and distributed computing.

Many NRL methods are proposed to approximately preserve local network structures. DeepWalk[7] preserves the neighborhood structure of vertexes by using information from truncated random walks. Node2vec[8] demonstrates that DeepWalk does not capture the diversity of connectivity patterns in a network. It suggests a biased random walking strategy which combines BFS style and DFS style neighborhood exploration. LINE[9] is proposed for large scale network embedding. It preserves the high-order proximity and easily scales to millions of vertices. Cao *et al.*[10] presented a deep graph representation model which adopts a random surfing strategy to capture graph structural information. Feng *et al.*[11] proposed the "degree penalty" principle which preserves the scale-free property by punishing the proximity between high-degree vertexes. Wang *et al.*[12] suggested a semi-supervised deep model which is able to capture the highly nonlinear network structure by optimizing multiple layers of nonlinear functions. Yanardag *et al.*[13] presented a general framework for modeling sub-structure similarity in graphs. In addition, some methods are also proposed to preserve the global network structures. Wang *et al.*[14] suggested a modularized nonnegative matrix factorization model, which preserves the community structures in network. Tu *et al.*[15] proposed a novel community enhancement mechanism which jointly embeds global community patterns into vertex representation by taking consideration of the community information. Chen *et al.*[16] presented a multilevel network representation learning paradigm which captures the global structure of the input network by recursively coalescing the input network into smaller but structurally similar networks.

In this paper, we suggest an NRL method based on the skip-gram model.[17,18] It first samples a set of vertex sequences with dense links from networks and then uses the skip-gram model to learn the representations of vertexes. Comparing with the three state-of-the-art methods (*i.e.*, DeepWalk,[7] Node2vec,[8] and DP-Walker,[11]) our method

*Project supported by the National Natural Science Foundation of China (Grant No. 61673085), the Program from the Sichuan Provincial Science and Technology, China (Grant No. 2018RZ0081), and the Fundamental Research Funds of China West Normal University (Grant No. 17E063).

†Corresponding author. E-mail: hejialin32@126.com

© 2020 Chinese Physical Society and IOP Publishing Ltd

can perform better on the tasks of community detection and multi-label classification.

2. Preliminaries

Let $G=(V,E)$ be an unweight network, where V is the vertex set containing n vertexes and E is the edge set. Each $e_{ij}\in E$ indicates an edge between the vertexes v_i and v_j. Traditionally, vertexes are represented by the rows of adjacency matrix reflecting the relationships between vertexes in network. Although the adjacency matrix can capture the topological properties in network, the high computation and space cost make it hard to apply to the large networks. So the NRL methods aim to map each vertex $v_i\in V$ into a low-dimensional space $\mathscr{R}^d$, *i.e.*, learning a function $\boldsymbol{f}_G: V\rightarrow\mathscr{R}^d$, where d represents the number of dimensions of feature representation.

2.1. Skip-gram model

The skip-gram[17,18] model has proven to be an efficient way to learn the high-quality word representations. Specifically, given a word sequence $\boldsymbol{w}=\{w_1,w_2,\ldots,w_n\}$, the skip-gram model learns word representation by minimizing the following objective function:

$$J=-\frac{1}{n}\sum_{i=1}^{n}\sum_{i-t\leq j\leq i+t,j\neq i}\log p(w_j|w_i), \tag{1}$$

where t is the window size, w_j is the context word of w_i within a window, and the probability $p(w_j|w_i)$ is defined by

$$p(w_j|w_i)=\frac{\exp\left(\boldsymbol{\Phi}\left(w_i\right)\cdot\boldsymbol{\Phi}'\left(w_j\right)\right)}{\sum_{w\in\boldsymbol{W}}\exp\left(\boldsymbol{\Phi}\left(w_i\right)\cdot\boldsymbol{\Phi}'\left(w\right)\right)}, \tag{2}$$

where $\boldsymbol{\Phi}\left(w\right)$ and $\boldsymbol{\Phi}'\left(w\right)$ are w's word representations when it behaves as a target and context respectively, and $\boldsymbol{W}$ represents the vocabulary set.

2.2. Vertex sequences sampling

Many NRL methods such as DeepWalk,[7] Node2vec,[8] and DP-Walker[11] perform *random walks* over the given network G first, and then form a set of vertex sequences $\boldsymbol{S}=\{\boldsymbol{s_1},\boldsymbol{s_2},\ldots,\boldsymbol{s_n}\}$, where each vertex sequence can be denoted as $\boldsymbol{s}=\{v_1,\ldots,v_{|\boldsymbol{s}|}\}$. By treating each vertex sequence as word sentence, the skip-gram model can be used to learn the representation of vertexes from the set $\boldsymbol{S}$.

For DeepWalk, it regards $p\left(v_{i+1}|v_i\right)$ as a uniform distribution where each neighbor of v_i has equal chance to be sampled. For Node2vec, it uses a biased random walk and the probability of the random walk jumping from v_i to v_{i+1} is defined as:

$$p\left(v_{i+1}|v_i\right)\propto\begin{cases}\dfrac{1}{p}, & \text{if } l_{i-1,i+1}=0,\\ 1, & \text{if } l_{i-1,i+1}=1,\\ \dfrac{1}{q}, & \text{if } l_{i-1,i+1}=2,\end{cases} \tag{3}$$

and $l_{i-1,i+1}$ denotes the shortest path distance between two vertexes v_{i-1} and v_{i+1}. In Eq. (3), the parameters p and q allow search procedure to interpolate between the breadth-first sampling and the depth-first sampling. For DP-Walker, the probability $p(v_{i+1}|v_i)$ is defined as

$$p\left(v_{i+1}|v_i\right)\propto\frac{C_{i,i+1}}{(k_ik_{i+1})^{\beta}}, \tag{4}$$

where k_i is the degree of vertex v_i, $C_{i,i+1}$ is the number of common neighbors of v_i and v_{i+1}, and β is the model parameter.

2.3. Mean shift clustering

Mean shift[19] represents a general non-parametric clustering procedure. In contrast to the classic *k-means* clustering approach,[20] there are neither the assumptions on the shape of the distribution nor the number of clusters. Given n data points $\boldsymbol{x}_i\in\mathscr{R}^d$ $(i=1,2,\ldots,n)$, the multivariate kernel density estimate using a radially symmetric kernel, $K(\boldsymbol{x})$, is given by

$$g\left(x\right)=\frac{1}{nh^d}\sum_{i=1}^{n}K\left(\frac{\boldsymbol{x}-\boldsymbol{x}_i}{h}\right), \tag{5}$$

where h is the radius of kernel. For each data point $\boldsymbol{x}_i$, one performs a gradient ascent procedure on the local estimated density until convergence. The data points associated with the same stationary point belong to the same cluster.

2.4. Community-based metric

Normalized mutual information measure (NMI)[21] is a similarity measure between two community partitions A and B based on the information theory. The NMI is defined as

$$\mathrm{NMI}\left(A,B\right)=\frac{-2\sum_{i=1}^{M_A}\sum_{j=1}^{M_B}M_{ij}\log\left(M_{ij}N/M_{i.}M_{.j}\right)}{\sum_{i=1}^{M_A}M_{i.}\log\left(M_{i.}/N\right)+\sum_{j=1}^{M_B}M_{.j}\log\left(M_{.j}/N\right)}, \tag{6}$$

where $\boldsymbol{M}$ is the confusion matrix in which rows correspond to the 'truth' communities and columns correspond to the 'observed' communities and N is the number of vertexes. M_{ij} is the number of vertexes in the truth community i that appear in the observed community j. M_A and M_B represent the numbers of truth communities and observed communities, respectively. $M_{i.}$ and $M_{.j}$ represent the sum over the i-th row and j-th column of the matrix $\boldsymbol{M}$, respectively. The value of NMI ranges from 0 to 1. The NMI equals 1 if the estimated community matches the truth community and 0 in the opposite case.

2.5. Label-based metric

Let $B(TP_j,FP_j,TN_j,FN_j)$ represent some specific binary classification metric where TP_j, FP_j, TN_j, and FN_j denote the accuracy, precision, recall, and F^{β} respectively, the label-based classification metrics can be obtained in either of the following models:[21]

(i) macro-F_1 model

$$B_{\text{macro}} = \frac{1}{q}\sum_{j=1}^{q} B(TP_j, FP_j, TN_j, FN_j); \tag{7}$$

(ii) micro-F_1 model

$$B_{\text{micro}} = B\left(\sum_{j=1}^{q} TP_j, \sum_{j=1}^{q} FP_j, \sum_{j=1}^{q} TN_j, \sum_{j=1}^{q} FN_j\right). \tag{8}$$

3. Modularity-based random walk

The NRL methods aim at learning the low-dimensional vertex representation which should preserve many important structural characteristics in network such as community structure.[23] However, for the three state-of-the-art methods mentioned in Subection 2.2, they did not consider the community structure while sampling vertex sequences in network. So they might not perform better when their learnt representations are used as features for complex tasks on networks.

To capture the community structure, we sample dense vertex sequences from network because communities have relatively high link density. We use the modularity function[24] to obtain the dense vertex sequences. The modularity function for vertex sequence $\boldsymbol{s}$ is defined as

$$Q_s = \frac{a_s}{m} - \left(\frac{k_s}{2m}\right)^2, \tag{9}$$

where a_s is the number of internal edges in $\boldsymbol{s}$, and k_s is the total degree of the vertexes in $\boldsymbol{s}$. The gain in modularity $\Delta Q_{\boldsymbol{s}}^{v}$ obtained by moving a vertex v into a vertex sequence $\boldsymbol{s}$ can easily be computed by

$$\Delta Q_{\boldsymbol{s}}^{v} = Q_{\boldsymbol{s}+\{v\}} - Q_{\boldsymbol{s}}, \tag{10}$$

where the symbol $\boldsymbol{s}+\{v\}$ indicates the new vertex sequence deduced from $\boldsymbol{s}$ when vertex v is added. Let $\boldsymbol{s} = \{v_1, \ldots, v_{|\boldsymbol{s}|}\}$ denote the vertex sequence in the walk where vertex $v_{|\boldsymbol{s}|}$ is the last added vertex, and let $v_{|\boldsymbol{s}|}$ be the current vertex, and let $N'(v_{|\boldsymbol{s}|})$ denote the neighbors of the current vertex $v_{|\boldsymbol{s}|}$ not included in $\boldsymbol{s}$. The next vertex $v_{|\boldsymbol{s}|+1}$ is generated by the following steps.

(I) Sample a vertex $v_{|\boldsymbol{s}|+1}$ from the set $N'(v_{|\boldsymbol{s}|})$ randomly;

(II) Calculate the $\Delta Q_{\boldsymbol{s}}^{v_{|\boldsymbol{s}|+1}}$ according to Eq. (10);

(III) If $\Delta Q_{\boldsymbol{s}}^{v_{|\boldsymbol{s}|+1}} < 0$, remove $v_{|\boldsymbol{s}|+1}$ from the set $N'(v_{|\boldsymbol{s}|})$ and return to Step (I);

(IV) If $\Delta Q_{\boldsymbol{s}}^{v_{|\boldsymbol{s}|+1}} > 0$, append $v_{|fs|+1}$ to the set $\boldsymbol{s}$ and mark $v_{|\boldsymbol{s}|+1}$ as the current vertex.

Steps (I)–(IV) are repeated until the length of vertex sequence $\boldsymbol{s}$ are met. Most of vertexes in $\boldsymbol{s}$ have relatively high link density and tend to be similar with each other because we only sample the vertexes which can increase the density of $\boldsymbol{s}$. The pseudo-code for our method is given in Algorithm 1 and Algorithm 2, describing the representation learning of vertexes and the sampling of vertex sequences respectively.

Algorithm 1 M-Walker(G, t, d, r, η)

Input: graph $G(V,E)$, window size t, embedding size d, walks per vertex r, walk length η
Output: matrix of vertex representation $\Phi \in \mathscr{R}^{|V|\times d}$

```
S ← φ
i ← 0
while i < r do
    for each v ∈ V do
        s ← VertexSequenceSample(G, v, η)
        Append s to S
    end for
    i = i + 1
end while
Φ ← SkipGram(S, t)
return Φ
```

Algorithm 2 VertexSequenceSample(G, v_1, η)

Input: graph $G = (V,E)$, starting vertex v_1, walk length η
Output: vertex sequence $\boldsymbol{s}$

```
s ← φ
i ← 0
Append v_1 to s
N'(v_|s|) ← N(v_1)
while i < l do
    Sample a vertex u from N'(v_|s|) randomly
    Calculate the ΔQ_s^u according to Eq. (10)
    if ΔQ_s^u < 0 then
        Remove u from the set N'(v_|s|)
    else
        Append u to s
        N'(v_|s|) ← {x|x ∈ N(u) and x ∉ s}
        i ← i + 1
    end if
end while
return s
```

4. Experiments

In experiments, we evaluate the performance of our method on the tasks of both community detection and multi-label classification respectively.

4.1. Baseline methods

We compare our method (M-Walker) with the following NRL methods:

i) **DeepWalk**: DeepWalk[7] first samples vertex sequences by performing random walks on networks, and then the SkipGram model[17,18] is used to learn the representation of vertexes.

ii) **Node2vec**: Node2vec[8] adopts a biased random walk to sample vertex sequences. By controlling parameters p and q, the search procedure can interpolate between breadth-first

sampling and depth-first sampling.

iii) **DP-Walker**: DP-Walker[11] designs a novel random walk strategy to sample vertex sequences. The method preserves the first- and second-order proximity by punishing vertexes with high degree.

For all methods, we set the representation dimension d to 128, the window size t to 10, the walk length η to 40, and the number of walks per vertex r to 10. Besides, both p and q are set to 0.25 for Node2vec.

4.2. Community detection

We apply *mean shift* to the learned representation of vertexes and use the NMI index to measure the quality of community partitions in network.

4.2.1. Real networks

Four real networks with true communities are used to evaluate the performance of M-Walker method, including Karate,[25] Football,[26] Dolphin,[27] and PolBooks.[28] Karate network reflects the social relations in a karate club where a vertex represents a club member, and an edge indicates social communication between two club members. Football network is the match network of American football games between Division IA colleges during the regular season Fall 2000. Its vertexes represent teams, and edges indicate matches between two corresponding teams. Dolphin network is a social network of bottlenose dolphins in Doubtful Sound, New Zealand. Vertexes represent dolphins, and edges are established by observation when there is a statistically significant frequent association between dolphins. PolBooks network is derived from the politic books about US politics published around the time of the 2004 presidential election. Vertexes represent books sold by the online bookseller Amazon.com, and edges represent frequent co-purchasing of books by the same buyers. More details about these four networks are listed in Table 1, including the number of Vertexes ($|V|$), the number of edges ($|E|$), average degree ($\langle k\rangle$), mean squared degree($\langle k^2\rangle$), average clustering coefficient(cc), and the number of truth communities (nc).

Table 1. Statistics of four real networks with truth communities.

Network	$\|V\|$	$\|E\|$	$\langle k\rangle$	$\langle k^2\rangle$	cc	nc
Karate	34	78	4.59	35.65	0.59	2
Dolphin	62	159	5.12	0.6600	0.30	2
PolBooks	105	441	8.4	100.25	0.49	3
Football	115	613	10.66	114.43	0.40	12

The results are shown in Table 2. From Table 2, it can be seen that M-Walker outperforms all three other methods on all networks. M-Walker can correctly detect all truth communities on the Karate and Polbooks networks, but three other methods can not. In particular, the NMI of Node2vec on the Karate network is only 0.267. M-Walker is the best method on the Polbooks network. Compared with DP-Walker, M-Walker improves NMI by 11.36%. On the Football network, although all methods perform well, the NMI of M-Walker is still slightly higher than those of three other methods. In addition, we can also notice that M-Walker's NMI is different on different networks, so M-Walker's performance has to do with the network structures.

Table 2. The NMI on four real networks with truth communities.

Method	Karate	Dolphin	PolBooks	Football
DeepWalk	0.714	0.814	0.568	0.920
Node2vec	0.267	0.889	0.574	0.916
DP-Walker	0.837	0.889	0.581	0.924
M-Walker	**1.0**	**1.0**	**0.647**	**0.930**

4.2.2. Synthetic networks

We also evaluate the performance of M-Walker method on the synthetic scale-free networks generated by the LFR model.[29] In the LFR model, the distributions of degree and community size are power laws, with exponents α and β, respectively. In order to better analyze the performance of M-Walker, we only compare it with the DP-Walker method.

First of all, we generate six synthetic networks with different community structures. The mixing parameter μ is set to 0.1, 0.2, 0.3, 0.4, 0.5, and 0.6 respectively. The smaller the parameter μ is, the stronger the community structure is. The other parameters for six synthetic networks are the same. Details are described in Table 3, which includes the number of vertexes, the number of edges, the degree exponent (α), the community exponent (β), the global mixing parameter (μ), the average degree $\langle k\rangle$, and the modularity (Q). Newman *et al.*[24] pointed out that networks have obvious community structure when $Q > 0.3$. Specifically, when $0.3 < Q < 0.5$, networks have a weak community structure; when $Q > 0.5$, networks have a strong community structure. Thus, in our experiments, LFR1-4 networks have a strong community structure and LFR4-5 networks have a weak one.

Table 3. Statistics of six synthetic networks with different community structure.

Network	$\|V\|$	$\|E\|$	α	β	μ	$\langle k\rangle$	Q
LFR1	500	2477	2.5	1.5	**0.1**	10	0.835
LFR2	500	2508	2.5	1.5	**0.2**	10	0.733
LFR3	500	2340	2.5	1.5	**0.3**	10	0.646
LFR4	500	2389	2.5	1.5	**0.4**	10	0.553
LFR5	500	2507	2.5	1.5	**0.5**	10	0.439
LFR6	500	2347	2.5	1.5	**0.6**	10	0.338

The results are shown in Fig. 1. It can be seen from Fig. 1, that the M-Walker method outperforms the DP-Walker method on all networks. It should also be noticed that M-Walker's gain is proportional to the $\langle\mu\rangle$, *i.e.*, the higher the $\langle\mu\rangle$ is, the higher the M-Walker's gain is. When $\mu = 0.1$, the

M-Walker's gain is the lowest, less than 2.5%. The reason for this is that the LFR1 network has strong community structure. For DP-Walker method, because of the high link density inside a community, the random walker spends a long time within it. So DP-Walker's sampled vertex sequences $\boldsymbol{s}$ can well capture strong community structure since most vertexes in $\boldsymbol{s}$ come from the same community. The M-Walker'gain is the highest, when $\mu = 0.6$, more than 22.5%. The reason for this is because the LFR6 network has weak community structure. For DP-Walker, because of many edges between weak communities, the random walker can easily jump from one weak community to another. Thus the vertex sequences $\boldsymbol{s}$ sampled by DP-Walker can not well capture weak community structure since most vertexes in $\boldsymbol{s}$ come from different weak communities. Most vertexes in a vertex sequence $\boldsymbol{s}$, however, for M-Walker, come from the same community because the $\boldsymbol{s}$ has relatively dense internal links. Therefore M-Walker can well capture weak community structure. It is possible to explain similarly the other four networks, such as LFR2 and LFR5. From the above analysis, M-Walker not only performs well on weak-community networks but also on strong-community networks.

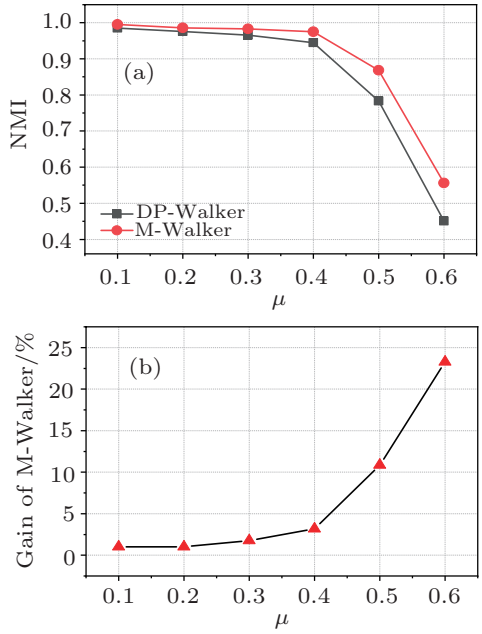


Fig. 1. The NMI (a) and the M-Walker's gain (b) on six synthetic networks with different community structure.

Secondly, we generate six synthetic networks with varying average degree $\langle k \rangle$. The other parameters for these six synthetic networks are the same and details are listed in Table 4. From Table 4, it can be seen that the difference between the maximal Q and the minimal Q is only 0.003, so these six synthetic networks have the same strength of community structure.

The results are illustrated in Fig. 2. It can be seen from Fig. 2 that on all networks the M-Walker method outperforms the DP-Walker method. In addition, M-Walker's gain changes very little as the average degree $\langle k \rangle$ increases. Thus, if the strength of community structure in network is the same, the change of $\langle k \rangle$ has little impact on the performance of M-Walker method.

Table 4. Statistics of six synthetic networks with different average degree $\langle k \rangle$.

Network	$\lvert V \rvert$	$\lvert E \rvert$	α	β	μ	$\langle k \rangle$	Q
ALFR1	500	2034	2.5	1.5	0.5	**8**	0.441
ALFR2	500	2482	2.5	1.5	0.5	**9**	0.443
ALFR3	500	2567	2.5	1.5	0.5	**10**	0.442
ALFR4	500	2809	2.5	1.5	0.5	**11**	0.442
ALFR5	500	2942	2.5	1.5	0.5	**12**	0.442
ALFR6	500	3226	2.5	1.5	0.5	**13**	0.444

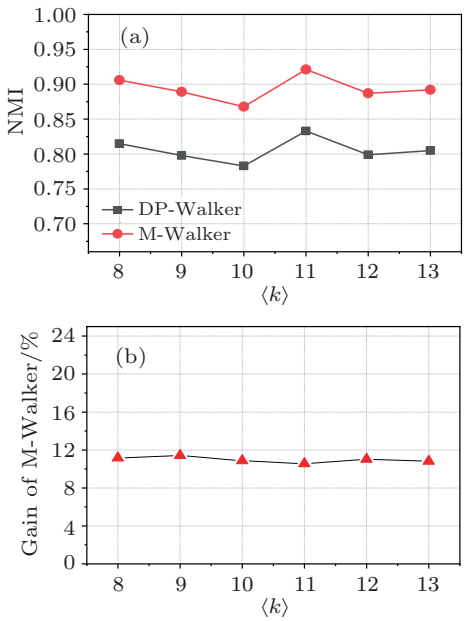


Fig. 2. The NMI (a) and the M-Walker's gain (b) on six synthetic networks with varying average degree $\langle k \rangle$.

4.3. Multi-label classification

We further assess the performance of our method in the task of multi-label classification. We use the same experimental procedure as in Ref. [7] to facilitate the comparisons between our method and three other methods. Specifically, a portion of labeled vertexes is selected randomly as a training set and the rest as a test set. The one-vs.-rest regression model implemented by LibLinear[30] is used to return the most likely label. In order to accelerate the training speed of multi-label classifiers, we employ smaller training ratios on all networks, ranging from 1% to 9%. Repeat the above process 50 times and report the average micro-F_1 results.

4.3.1. Datasets

Three real networks are used to evaluate the performance of our method on the task of multi-label classification. The DBLP network[31] is a co-author graph in computer science. The labels indicate the areas of research in which a researcher

publishes his work. The four research areas in this dataset are DB, DM, IR, and ML. The BlogCatalog network[3] is a social network of users on the BlogCatalog website. The labels are the topic categories provided by the authors. The CiteSeer[32] is a citation network between computer science publications. The labels indicate the area of research to which they belong. The papers are divided into 6 categories: Agents, AI, DB, IR, ML, and HCI. Further details on these three networks is given in Table 5, including the number of vertexes ($|V|$), the number of edges ($|E|$) and the number of class labels ($|L|$).

Table 5. Statistics of three real networks on the task of multi-label classification.

Network	DBLP	BlogCatalog	CiteSeer
$\|V\|$	29,199	10,312	3,312
$\|E\|$	133,664	333,983	4,732
$\|L\|$	4	39	6

4.3.2. Result analysis

The results are shown in Tables 6–8. Numbers in bold represent the highest performance in each column. From Tables 6–8, it can be seen that M-Walker outperforms three other methods on all networks. Specifically, on the CiteSeer network, when trained with 5% of the vertexes labeled, M-Walker performs better than the other three methods when they are given 9% of the data. On the BlogCatalog network, M-Walker's performance when 5% of the network is labeled beats the DeepWalk and Node2vec methods when they have been 9% of the data. On the DBLP network, when only trained with 3% of the vertexes labeled, M-Walker's is better than the DP-Walker when it is given 9% of the data. In other words, M-Walker can outperform the DP-Walker with 60% less training data.

Table 6. The micro-F_1 score on CiteSeer (%).

Training ratio	1%	3%	5%	7%	9%
DeepWalk	37.86	45.69	48.32	50.19	51.65
Node2vec	37.77	45.42	48.29	50.09	51.20
DP-Walker	36.94	45.93	49.39	51.24	52.46
Q-Walker	**40.36**	**48.54**	**51.68**	**53.48**	**54.68**

Table 7. The micro-F_1 score on BlogCatalog (%).

Training ratio	1%	3%	5%	7%	9%
DeepWalk	22.12	27.49	29.38	30.65	31.64
Node2vec	22.50	27.51	29.48	30.77	31.73
DP-Walker	24.22	28.80	30.68	31.86	32.84
Q-Walker	**25.61**	**30.58**	**32.54**	**33.71**	**34.63**

Table 8. The micro-F_1 score on DBLP (%).

Training ratio	1%	3%	5%	7%	9%
DeepWalk	58.08	63.25	65.45	66.48	67.06
Node2vec	58.09	63.33	65.61	66.57	67.20
DP-Walker	57.03	61.43	63.39	64.42	64.96
Q-Walker	**61.58**	**65.99**	**67.74**	**68.24**	**69.24**

5. Conclusion

Many classic methods for network representation learning are based on the skip-gram model. There is no mechanism, however, to ensure that the vertex sequences sampled by these classic methods have dense links and are similar with each other. To solve this problem, we use the modularity function to sample vertex sequences. The performance of our method is evaluated on the tasks of community detection and multi-label classification. Compared with the baseline methods, our method can better capture the weak community structure. Meanwhile, experimental results on three real networks also show that our method outperforms these baseline methods on multi-label classification.

Two open issues will require more research in the future. First, the vertex sequences sampled by our method can capture the community structure well. However, real networks have many important characteristics, such as clustering coefficient and mean shortest path length. We need a vertex sampling approach to capture more network characteristics in the future. Second, most real-world networks are dynamic — namely, they evolve over time. This poses a challenge for our method applied to large dynamic networks because it is impractical and inefficient to re-learn vertex representations from scratch for each snapshot. In the future, we should concentrate on how the vertex sequences sampled by our method can be modified when a network changes over time. Specifically, we will improve our method from two aspects: I) we are looking at how a pre-generated set of random walks generated from one network snapshot can be updated when the network changes; II) we are investigating how to update vertex representations incrementally given the current set of random walks, by treating the updates as a fine-tuning step in our method.

References

[1] Sun H L, Ch'ng E, Yong X, Garibaldi J M, See S and Chen D B 2018 *Physica A* **496** 108-120

[2] He J L, and Chen D B 2015 *Physica A* **429** 87-94

[3] Tang L and Liu H 2009 *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, June 28–July 1, 2009, Paris, France, p. 817

[4] Tang L, and Liu H 2009 *Proceedings of the 18th ACM Conference on Information and Knowledge Management*, November 2–6, 2009, Hong Kong, China, p. 1107

[5] Xie Y, Gong M, Qin A K, Tang Z and Fan X 2019 *Inf. Sci.* **504** 20

[6] Gao Y, Gong M, Xie Y and Zhong H 2020 *Knowl Based Syst* **193** 105418

[7] Perozzi B, Al-Rfou R and Skiena S 2014 *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, August 24–27, 2014, New York, USA, p. 701

[8] Grover A, and Leskovec J 2016 *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, August 24–27, 2016, San Francisco, USA, p. 855

[9] Tang J, Qu M, Wang M, Zhang M, Yan J and Mei Q 2015 *Proceedings of the 24th International Conference on World Wide Web*, May 18–22, 2015, Florence, Italy, p. 1067

[10] Cao S, Lu W and Xu Q 2016 *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence*, February 12–17, 2016, Phoenix, USA, p. 1145

[11] Feng R, Yang Y, Hu W, Wu F and Zhuang Y 2018 *Proceedings of the 32th AAAI Conference on Artificial Intelligence*, February 2–7, 2018, New Orleans, USA, p. 282

[12] Wang D, Cui P and Zhu W 2016 *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, August 24–27, 2016, San Francisco, USA, p. 1225

[13] Yanardag P and Vishwanathan S V N 2015 *Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, August 23–27, 2015, New York, USA, p. 1365

[14] Wang X, Cui P, Wang J, Pei J, Zhu W and Yang S 2017 *Proceedings of the 31 AAAI Conference on Artificial Intelligence*, February 4–9, 2017, San Francisco, USA, p. 203

[15] Tu C, Zeng X, Wang H, Zhang Z, Liu Z, Sun M and Lin L 2018 *IEEE Trans. Knowl. Data Eng.* **31** 1051

[16] Chen H, Perozzi B, Hu Y and Skiena S 2017 *Proceedings of the 32th AAAI Conference on Artificial Intelligence*, Feburary 2–7, 2018, New Orleans, USA, p. 2127

[17] Mikolov T, Chen K, Corrado G and Dean J 2013 arXiv:1301.3781 [cs.CL]

[18] Armandpour M, Ding P, Huang J and Hu X 2019 *Proceedings of the 33th AAAI Conference on Artificial Intelligence*, January 27–February 1, 2019, Hawaii, USA, p. 3191

[19] Cheng Y 1995 *IEEE Trans. Pattern Anal. Mach. Intell.* **17** 790-799

[20] Arthur D and Vassilvitskii S 2007 *Proceedings of the 18th Annual ACM-SIAM Symposium on Discrete Algorithms*, January 7–9, 2007, New Orleans, USA, p. 1027

[21] Danon L, Diaz-Guilera A, Duch J and Arenas A 2005 *J. Stat. Mech. Theory Exp.* **2005**, p. P09008

[22] Tsoumakas G and Vlahavas I 2007 *Machine Learning: ECML 2007* (Berlin: Springer-Verlag) p. 406

[23] Radicchi F, Castellano C, Cecconi F, Loreto V and Parisi D 2004 *Proc. Natl. Acad. Sci. USA* **101** 2658

[24] Newman M E and Girvan M 2004 *Phys. Rev. E* **69** 026113

[25] Zachary W W 1977 *J. Anthropol. Res.* **33** 452

[26] Girvan M and Newman M E 2002 *Proc. Natl. Acad. Sci. USA* **99** 7821

[27] Lusseau D, Schneider K, Boisseau O J, Haase P, Slooten E and Dawson S M 2003 *Behav. Ecol. Sociobiol.* **54** 396

[28] Rossi R and Ahmed N 2015 *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence*, January 25–30, 2015, Austin, USA, p. 777

[29] Lancichinetti A, Fortunato S and Radicchi F 2008 *Phys. Rev. E* **78** 046110

[30] Fan R E, Chang K W, Hsieh C J, Wang X R and Lin C J 2008 *J. Mach. Learn. Res.* **9** 1871

[31] Perozzi B, Kulkarni V, Chen H and Skiena S 2017 *Proceedings of the 2017 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, July 31–August 3, 2017, Sydney, Australia, p. 258

[32] Sen P, Namata G, Bilgic M, Getoor L, Galligher B and Eliassi-Rad T 2008 *AI Mag.* **29** 93