# Network Embedding Based on Biased Random Walk for Community Detection in Attributed Networks

Kun Guo, Zizheng Zhao, Zhiyong Yu, *Member, IEEE*, Wenzhong Guo, Ronghua Lin, Yong Tang, and Ling Wu

***Abstract*— Community detection is a fundamental problem in complex network analysis that aims to find closely related groups of nodes. Recently, network embedding techniques have been integrated into community detection in two manners to capture the intricate relationships between nodes. The two-staged manner generates node embedding vectors and obtains communities by running a clustering algorithm on them. The single-staged manner simultaneously obtains node embedding vectors and communities by optimizing a hybrid objective concerning with node–community relationships. The general-purpose network embedding algorithms used in the first manner do not emphasize retaining node–community relationships. The second manner ignores the influence of a node's location in a community (at the center or boundary) and its attributes on community generation. In this article, we propose a biased-random-walk-based community detection (BRWCD) algorithm to tackle the issues. First, a topology-weighted degree is designed to enhance the random walk at the boundary of and inside a community to extract communities precisely. Second, we design an attribute-to-node influence index and an attribute-weighted degree to distinguish different attributes' influence on node transition to obtain communities with high internal cohesion. Comprehensive experiments on the real-world and synthetic networks demonstrate that BRWCD achieves nearly 10% higher accuracy at most than the state-of-the-art algorithms.**

***Index Terms*— Community detection, complex network, matrix factorization, network embedding, random walk.**

## NOMENCLATURE

| | |
|---|---|
| $G$ | Attributed network. |
| $G_a$ | Attribute-augmented network. |
| $V, V_a$ | Set of nodes in $G$ and $G_a$. |
| $E, E_a$ | Set of edges in $G$ and $G_a$ |
| $n, n_a$ | Number of nodes in $G$ and $G_a$. |
| $e, e_a$ | Number of edges in $G$ and $G_a$. |
| $f$ | Dimension of attribute vectors. |
| $q$ | Dimension of embedding vectors. |
| $c_{im}$ | Influence of attribute $m$ to node $i$. |
| $N(i)$ | Topology neighbors of node $i$. |
| $N(im)$ | Topology neighbors of node $i$ with attribute $m$. |
| $d_i^w$ | Topology-weighted degree. |
| $d_i^{w(c)}$ | Attribute-weighted degree. |
| $\mathbf{A} \in \mathbb{R}^{n\times n}$ | Original adjacency matrix. |
| $\mathbf{X} \in \mathbb{R}^{n\times f}$ | Attribute matrix. |
| $\mathbf{Z} \in \mathbb{R}^{n\times q}$ | Embedding matrix. |
| $\mathbf{S}^{(t)} \in \mathbb{R}^{n\times n}$ | T ransition probability matrix from topology nodes to topology nodes. |
| $\mathbf{S}^{(a)} \in \mathbb{R}^{n\times f}$ | T ransition probability matrix from topology nodes to attribute nodes. |
| $\mathbf{S}^{(c)} \in \mathbb{R}^{f\times n}$ | T ransition probability matrix from attribute nodes to topology nodes. |
| $\mathbf{S}^{(g)} \in \mathbb{R}^{(n+f)\times(n+f)}$ | Transition probability matrix in $G_a$. |

Manuscript received December 17, 2021; revised March 29, 2022; accepted May 9, 2022. This work was supported in part by the National Natural Science Foundation of China under Grant 62002063 and Grant U21A20472, in part by the National Key Research and Development Plan of China under Grant 2021YFB3600503, and in part by the Fujian Provincial Guiding Project under Grant 2020H0008. *(Corresponding author: Wenzhong Guo.)*

Kun Guo, Zizheng Zhao, Zhiyong Yu, Wenzhong Guo, and Ling Wu are with the Fujian Provincial Key Laboratory of Network Computing and Intelligent Information Processing and the College of Computer and Data Science, Fuzhou University, Fuzhou 350108, China (e-mail: gukn@fzu.edu.cn; zhzz7364@gmail.com; yuzhiyong@fzu.edu.cn; guowenzhong@fzu.edu.cn; wuling1985@fzu.edu.cn).

Ronghua Lin and Yong Tang are with the School of Computer Science, South China Normal University, Guangzhou 510631, China (e-mail: rhlin@m.scnu.edu.cn; ytang@m.scnu.edu.cn).

Digital Object Identifier 10.1109/TCSS.2022.3174693

## I. INTRODUCTION

COMMUNITY detection aims to identify closely related nodes in complex networks [1], which has a wide variety of practical applications. For instance, finding friend circles in social networks can aid in increasing online sales, while discovering new protein structures in biological networks can assist in new drug discovery. Recently, network embedding techniques have been applied in community detection to extract deep relationships between nodes [2], [3]. In addition, network embedding can naturally take advantage of not only the topology of a network but also its node attributes, which is crucial for community detection in attributed networks (networks with node or edge attributes).

There are generally two manners to integrate network embedding into community detection. The first manner obtains communities in two stages. First, a network embedding algorithm such as DeepWalk [4] is employed to learn each node's embedding vector. Second, a classical clustering algorithm such as Kmeans is used to partition embedding vectors into groups. Such a two-staged scheme is perfect in incorporating

the advantages of network embedding and classical clustering algorithms. However, it may not work well for community detection because most network embedding algorithms do not pay much attention to retaining a node's community membership. Therefore, the clustering results may not be optimal. The second manner handles embedding vector generation and community detection in a single unified stage. Community membership vectors and embedding vectors of each node are optimized and output simultaneously. However, most existing single-staged algorithms neglect the influence of a node's location in a community and its attributes on the computation of node relationships. The lack of the above considerations may lead to inaccurate community detection in complex attributed networks, e.g., probabilistic generative model is a more informative keyword than data mining in a citation network in computer science, and ignoring the difference may result in communities containing less relevant papers.

In this article, we propose a biased-random-walk-based community detection (BRWCD) algorithm to solve the above problems. As discussed in [5], generating node embedding vectors based on random walk is equivalent to factorizing a transition probability matrix. Therefore, we construct a transition probability matrix considering both topology and attribute relationships between nodes and factorize the matrix by randomized singular value decomposition (rSVD) [6]. On the topology side, we define a topology-weighted degree and use it to construct a community-oriented transition probability matrix so that a random walk based on the matrix prefers the boundary and interior of a community. On the attribute side, we first build an attribute-augmented network by mapping each attribute value to an attribute node. Second, an attribute-to-node influence index and an attribute-weighted degree are designed to augment the former community-oriented matrix with the transition probability between topology and attribute nodes. In this manner, the influence of different locations and attributes of nodes is considered in a random walk to obtain communities with high internal cohesion.

The contributions of this article are summarized as follows.

1) By emphasizing the random walk at the boundary and interior of a community based on the topology-weighted degree, BRWCD outperforms the state-of-the-art algorithms in recognizing community boundaries precisely when nodes from different communities are highly mixed up.
2) By emphasizing the random walk on nodes with high attribute-to-node influence and attribute-weighted degree, BRWCD can distinguish the subtle influence of different attributes on the random walk on nodes and generate higher cohesive communities than the state-of-the-art algorithms.
3) Comprehensive experiments on real-world and synthetic networks prove that BRWCD can achieve over nearly 10% higher normalized mutual information (NMI) value compared with the state-of-the-art algorithms.

The rest of this article is organized as follows. Section II presents the studies related to community detection. In Section III, we define the problem of community detection and introduce the notations used in this article. Section IV presents the proposed algorithm BRWCD. The experimental results are given in Section V. Section VI concludes this article and discusses the future work.

## II. Related Work

In this section, we briefly introduce the community detection approaches based on traditional graph computing and newly emerged network embedding techniques, respectively.

### A. Traditional Community Detection

The traditional community detection algorithms apply graph computing techniques such as k-clique computation and graph clustering directly to partition nodes into communities. The typical algorithms include label propagation, link clustering, seed expansion, spectral clustering, clique percolation, modularity-based, and probabilistic-model-based algorithms [1]. NI-LPA [7] is an improved label propagation algorithm (LPA) that employs degree-based node importance to guide label update. HOSCIEN [8] calculates topology and attribute similarity between edges and merges the edges with high similarity recursively until the whole network becomes one community. ECES [9] adopts an extended Jaccard index to find central nodes and expand them to communities based on the memberships of other nodes to them. TADW-SC [10] and ASC [11] are spectral clustering methods that construct affinity matrices. TADW-SC employs TADW [12] to generate node embedding vectors and builds an affinity matrix by calculating the similarity between each pair of nodes according to the embedding vectors. ASC uses biased random walks considering network topology and attributes to calculate the similarity between each pair of nodes for the construction of an affinity matrix. MEMOEA [13] generates maximal cliques according to degree distribution and computes link strengths between cliques to construct a merged maximal clique graph for efficient community detection. Literature [14] proposes an algorithm that employs a fine-tuned community splitting and merging strategy to maximize modularity for community detection. NEMBP [15] combines a nested expectation-maximization algorithm with a belief propagation process to train a probabilistic community generative model to overcome the mismatch between topology and attribute similarity between nodes.

### B. Network Embedding-Based Community Detection

The community detection algorithms based on network embedding can be classified into two- and single-staged algorithms depending on whether they consider community detection and network embedding as a unified process or not.

*1) Two-Staged Algorithms:* The two-staged algorithms learn node embedding vectors and then perform classical clustering such as Kmeans on the embedding vectors to build communities. In the network embedding part, the existing network embedding techniques include graph autoencoder (GAE) [16], [17], multiobjective optimization [18], nonnegative matrix factorization (NMF) [19], random walk [20], and so on. AMIL [21] is an adversarial learning algorithm based on

autoencoders that transform attributes to node embedding vectors based on network topology. ONE [22] optimizes three loss functions associated with different types of outliers to obtain node embedding vectors. AANE [19] employs an attribute similarity matrix and splits the factorization of the matrix into subproblems. As one of the pioneering network embedding algorithms, DeepWalk [4] performs random walks on a network to obtain node sequences and employs the skip-gram model to learn node embedding vectors. CSADW [23] calculates the topological and attributed similarity between nodes to construct a weighted adjacency matrix, and then performs biased random walks according to the matrix. FeatWalk [24] treats attribute values as attribute nodes and the original nodes as topology nodes and selects the attribute nodes with the maximum attribute value as the next hop of nodes. RoSANE [25] calculates the similarity of attributes and prefers walking toward the most similar neighbor of a node. MIRand [20] and COANE [26] use the skip-gram model to learn node embedding vectors. MIRand performs biased random walks on a multilayer graph. COANE uses LDA [27] to generate initial communities and prefers walking inside communities.

In the clustering part, the algorithms mentioned above run a clustering algorithm on node embedding vectors to obtain communities. Most of the two-staged algorithms do not consider community structure during embedding vector generation. Therefore, the learned embedding vectors may retain few node–community relationships. Some algorithms that utilize nonlinear transformation can better capture the nonlinear relation in networks and obtain effective node embedding vectors for community detection. However, the nonlinear transformation-based algorithms generally need to input several hyperparameters that affect the performance of algorithms. The optimal values of hyperparameters are hard to determine for various networks.

*2) Single-Staged Algorithms:* The single-staged algorithms combine network embedding and community detection into a unified process by optimizing a hybrid objective. ComE [28] employs a cyclic optimization process to update community embedding vectors, node embedding vectors, and community assignments alternatively. M-NMF [29] adds a modularity constraint term to its objective and updates each node's community membership iteratively. CNRL [30] uses community embedding vectors (embedding vectors representing communities) to enhance node embedding vector generation and optimize an objective containing node–community memberships to obtain node embedding vectors and communities simultaneously. COSINE [31] adopts a Gaussian mixture model to optimize node–community relationships to detect communities. vGraph [32] is a probabilistic generative model that uses variational inference to ensure that the community memberships of a node's neighbors are similar. CommunityGAN [33] adopts a unified framework to combine the affiliation graph model (AGM) and generative adversarial nets (GANs) and optimizes each node's community membership by a GAN considering network cliques. NECS [34] utilizes the product of the node embedding matrix and the community structure embedding matrix of a network to construct a community membership matrix and updates each node's community membership by minimizing the difference between the similarity matrix and the power of the community membership matrix.

When confronted with attributed networks, some single-staged algorithms conduct matrix multiplication based on node attributes or compute an attribute similarity matrix to take node attributes into account. NMTF [35] employs three types of graph regularization strategies to capture the similarity between users based on their interaction. SCI [36] minimizes the difference between the power of a network's community membership matrix and its adjacency matrix to update each node's community membership. ASCD [37] improves SCI by inducing its hyperparameter automatically. CDE [38] utilizes the power of a network's community membership matrix to construct a community structure embedding matrix to update each node's community membership. NMFjGO [39] factorizes the attribute similarity matrix and the adjacency matrix of a network to obtain communities.

All of these methods treat each node equally regardless of whether it is at the center or the boundary of a community. They also do not distinguish the influence of different attributes on the random walk on nodes. Furthermore, some algorithms focus on optimizing specific objective functions that need several hyperparameters, which may bring an unstable performance in different networks. Therefore, it may be difficult for them to recognize communities with ambiguous boundaries in attributed networks. Unlike the existing network-embedding-based algorithms, BRWCD considers community structure in embedding vector generation based on the random walk biased toward topology nodes at community centers and boundaries and attributes with high influence on nodes to achieve more precise and delicate community division.

## III. Preliminaries

### A. Problem Definition

Consider an undirected attributed network $G = (V, E, \mathbf{A}, \mathbf{X})$, where $V$ is the set of $n$ nodes, $E \subseteq V \times V$ is the set of $e$ edges, $\mathbf{A} \in \{0, 1\}_{n\times n}$ is the adjacency matrix where $\mathbf{A}_{ij} = 1$ if $(i, j) \in E$ and 0 otherwise, and $\mathbf{X} \in \{0, 1\}_{n\times f}$ is the attribute matrix where $\mathbf{X}_{ij} = 1$ if node $i$ has attribute $j$ and 0 otherwise, and $f$ is the number of attributes. Network embedding aims to learn a node embedding matrix $\mathbf{Z}$ from a network $G$, with the $i$th row representing the $d$-dimensional embedding vector of node $i$. The problem of community detection is to partition the node set $V$ of a network $G$ into $k$ groups $V = \{c_1, \ldots, c_k\}$ where the nodes inside each group have closer relationships in both topology and attributes than the nodes in different groups. The notations used in this article are described in the Nomenclature.

### B. Matrix Factorization

Given an matrix $\mathbf{W} \in \mathbb{R}^{n\times m}$, matrix factorization aims to find matrices $\mathbf{V} \in \mathbb{R}^{n\times s}$ and $\mathbf{H} \in \mathbb{R}^{s\times m}$ $(s < n, s < m)$ that satisfy $\mathbf{W} \approx \mathbf{UV}$. rSVD [6] is a recently proposed efficient approach based on random projection to speed up matrix factorization. Specifically, rSVD employs a Gaussian random matrix $\mathbf{\Omega}$ to compute a small matrix $\mathbf{Y} = \mathbf{W\Omega}$ so that the

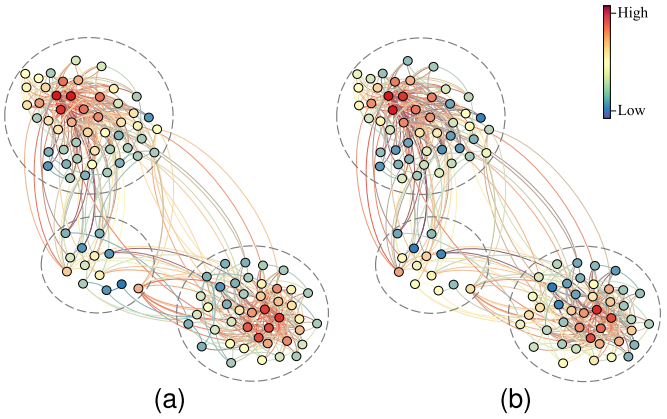


Fig. 1. Distribution of (a) degree and (b) topology-weighted degree of nodes. Dotted circles represent ground-truth communities. Different colors represent different levels of degree and topology-weighted degree of nodes.

approximate singular value decomposition (SVD) of $\mathbf{W}$ can be computed quickly by applying QR decomposition to $\mathbf{Y}$.

## IV. Proposed Algorithm

BRWCD is composed of two stages: 1) factorize a transition probability matrix biased toward the random walk at community centers and boundaries and on attributes with high influence on nodes and 2) run Kmeans on the embedding vectors to generate communities.

### A. Transition Probability Matrix Construction

We first construct a transition probability matrix based on a topology-weighted degree to emphasize the random walk at boundary and interior of each community. Second, the matrix is augmented based on an attribute-to-node influence index and an attribute-weighted degree to consider the influence of attributes on the random walk between nodes.

*1) Matrix Construction Based on Topology-Weighted Degree:* Literature [4] introduces the skip-gram model to process node sequences generated by random walk. Literature [40] proves that the skip-gram model with negative sampling is equivalent to NMF. We rewrite the matrix to be factorized in terms of random walk on networks as $\mathbf{M} = \log \mathbf{Q} - \log b$, where $\mathbf{Q}$ is defined as

$$\mathbf{Q} = \left( \frac{\Gamma_{(i,j)} |\mathcal{W}|}{\Gamma_{(i)} \cdot \Gamma_{(j)}} \right) \tag{1}$$

where $\mathcal{W}$ denotes the set of paths generated by random walk; $|\cdot|$ returns the number of elements in a set; and $\Gamma_{(i,j)}$, $\Gamma_{(i)}$, and $\Gamma_{(j)}$ are the numbers of node pair $(i, j)$, node $i$, and node $j$ in $\mathcal{W}$, respectively. Most algorithms treat each neighbor equally when selecting the next hop of a node. However, as revealed in Fig. 1, nodes at different locations of a community may contribute differently to its detection. The nodes at a community's center (center nodes) are vital to form its core. The nodes at the boundary (boundary nodes) are useful to segment adjacent communities. The nodes that are more strongly connected to center nodes than boundary nodes (intermediate nodes) are the majority in a community. Following this line of thinking, we define a node's topology-weighted degree as follows.

*Definition 1:* Topology-weighted degree of node $i$ is

$$d_i^w = \sum_{(i,j) \in E} \frac{1}{d_j} \tag{2}$$

where $d_j$ is node $j$'s degree.

Fig. 1 compares the degree and the topology-weighted degree of each node in three communities in the Polbooks network [41]. Colors from blue to red indicate the increase of degree values from low to high. As we can see in the figure, the colors of the nodes at community centers are close to red in both subfigures, which reflects that both degree and topology-weighted degree work well in locating community centers. On the other hand, the colors of the nodes at community boundaries in Fig. 1(b) are darker than that in Fig. 1(a), which means that the topology-weighted degree gives more weights to the boundary nodes than to the intermediate nodes because the former ones gain more weights from their low-degree neighbors than the latter ones. In this manner, the topology-weighted degree increases the probability of random walk on boundary nodes to distinguish community boundaries more precisely.

We can deduce an improved $\mathbf{Q}'$ based on $\mathbf{Q}$ in [5] and $d_i^w$ following [5, Proof of Theorem 2.3] as the length of random walk $L \to \infty$:

$$\begin{aligned}
\mathbf{Q}' &= \frac{\frac{\Gamma(i,j)}{|\mathcal{W}|}}{\frac{\Gamma(i)}{|\mathcal{W}|} \cdot \frac{\Gamma(j)}{|\mathcal{W}|}} \\
&\xrightarrow{p'(i)} \frac{\frac{1}{2L} \sum_{r=1}^{L} \left( \frac{d_i^w}{n} \left( \mathbf{D}^{-1}\mathbf{A} \right)_{i,j}^r + \frac{d_j^w}{n} \left( \mathbf{D}^{-1}\mathbf{A} \right)_{j,i}^r \right)}{\frac{d_i^w}{n} \cdot \frac{d_j^w}{n}} \\
&= \frac{n}{2L} \left( \frac{1}{d_j^w} \sum_{r=1}^{L} \left( \mathbf{D}^{-1}\mathbf{A} \right)_{i,j}^r + \frac{1}{d_i^w} \sum_{r=1}^{T} \left( \mathbf{D}^{-1}\mathbf{A} \right)_{j,i}^r \right) \\
&= \frac{n}{2L} \left( \sum_{r=1}^{L} (\mathbf{D}^{-1}\mathbf{A})^r \mathbf{D}_w^{-1} + \sum_{r=1}^{L} \mathbf{D}_w^{-1} ((\mathbf{D}^{-1}\mathbf{A})^T)^r \right)
\end{aligned} \tag{3}$$

where $\mathbf{D} = \mathrm{diag}(d_1, d_2, \ldots, d_n)$ is a diagonal matrix with $d_1, d_2, \ldots, d_n$ as its diagonal elements. $\mathbf{D}_w = \mathrm{diag}(d_1^w, d_2^w, \ldots, d_n^w)$ is another diagonal matrix with $d_1^w, d_2^w, \ldots, d_n^w$ as its diagonal elements. $p'(i) = d_i^w / n$ is the probability to select the starting node of a random path according to the topology-weighted degree, which is used to replace the $p(i) = d_i / \mathrm{vol}(G)$ used in [5]. $\mathrm{vol}(G) = \sum_i d_i$ is the volume of network $G$. For selecting the starting node, probability $p(i)$ is only biased to select the center nodes. However, the probability $p'(i)$ can bring a higher probability to select both the center nodes and the boundary nodes that are vital to community detection as the starting node.

$\mathbf{Q}'$ can be further improved by considering the locations of nodes. The Laplacian matrix $\mathbf{D}^{-1}\mathbf{A}$ in (3) treats each node equally in random walk. We can replace it with a new matrix

$\mathbf{S^{(t)}} \in \mathbb{R}^{n\times n}$ based on the topology-weighted degree as follows:

$$\mathbf{S^{(t)}}_{ij} = \begin{cases} \dfrac{w_t \sigma\left(\frac{\alpha}{d_j^w}\right)}{\sum_{(i,k)\in E} \sigma\left(\frac{\alpha}{d_k^w}\right)}, & (i,j) \in E \\ 0, & (i,j) \notin E \end{cases} \tag{4}$$

where $\sigma$ is the sigmoid function and $\alpha$ and $w_t$ are two parameters to finely tune the impact of the topology-weighted degree. Substituting $\mathbf{D}^{-1}\mathbf{A}$ with $\mathbf{S^{(t)}}$ in (3), we have

$$\mathbf{Q^{(t)}} = \frac{n}{2L}\sum_{r=1}^{L}\left(\left(\mathbf{S^{(t)}}\right)^r \mathbf{D}_w^{-1} + \mathbf{D}_w^{-1}\left(\left(\mathbf{S^{(t)}}\right)^T\right)^r\right). \tag{5}$$

As shown in Fig. 1(b), the random walk based on $\mathbf{Q^{(t)}}$ not only puts more emphasis on the nodes at community boundaries but also pushes the random walk at boundary nodes back toward community centers because the value of $\mathbf{S^{(t)}}_{ij}$ is reciprocal to $d_j^w$ in (4). Therefore, we do not sacrifice the exploration of the interior of a community for shedding more light on its boundary. In summary, BRWCD can recognize community boundaries more precisely by the construction of the transition probability matrix based on the topology-weighted degree.

### *B. Matrix Augmentation Based on Attribute Influence*

The community detection algorithms based solely on network topology can hardly find the attribute similarity between the nodes far away in topology in attributed networks. Therefore, the generated communities may miss some distant but closely related nodes. An intuitive solution is to consider both attribute and topology in the calculation of two nodes' similarity [20], [25]. However, calculating the similarity between all pairs of nodes is time-consuming. Moreover, selecting a proper similarity metric for a network is not an easy task. Another more complicated solution is to map each attribute value to an attribute node and add them to the original network to build an attribute-augmented network. The original nodes are called topology nodes. In this manner, the relationship between two topology nodes can be acquired naturally by a random walk on the network. Literature [42] is the first attempt in this direction. However, it calculates node similarity in a very time-consuming way.

In contrast, we construct a new attribute-augmented network to conduct an efficient random walk on it instead of the time-consuming similarity calculation, as shown in Fig. 2. The solid points denote the added attributed nodes and the hollow points represent the original topology nodes. The solid lines are the edges from the original network and the hollow lines are attribute edges connecting the attribute nodes and topology nodes. $\mathbf{X}_i$ is the attribute vector of topology node $i$. For example, the attribute vector of topology node $v_4$ is $\mathbf{X}_4 = (0,1,1)$. There are two attribute edges connecting $v_4$ to attribute nodes $a_2$ and $a_3$. If the attribute of a topology node $v_i$ is represented by a real-value vector such as $\mathbf{X}_4 = (0.1, 0.3, 0.2)$, BRWCD regards the real values as the transition probability from node $v_i$ to the corresponding attribute nodes. The transition probability matrix $\mathbf{S^{(g)}}$ for the attribute-augmented network is constructed by augmenting

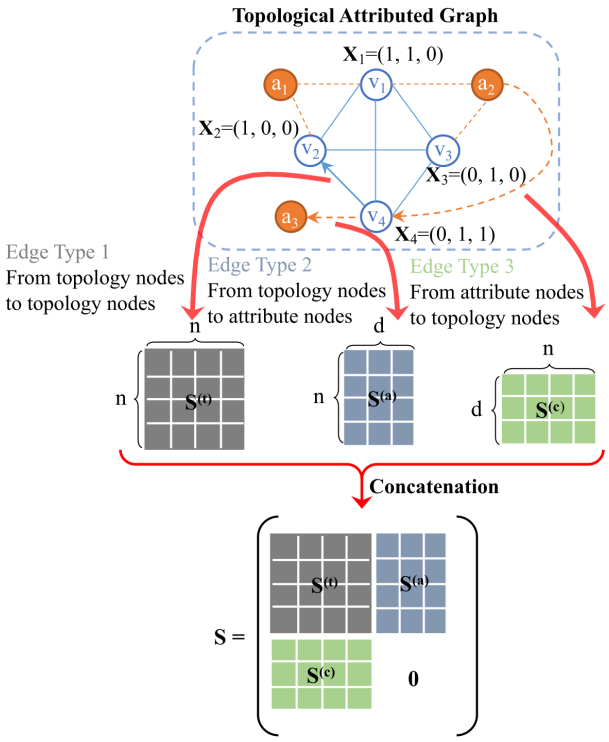


Fig. 2. Augmented attribute network and the augmented transition probability matrix $\mathbf{S}$.

matrix $\mathbf{S^{(t)}}$ with a matrix $\mathbf{S^{(a)}} \in \mathbb{R}^{n\times f}$ representing the transition probability from a topology node to an attribute node and a matrix $\mathbf{S^{(c)}} \in \mathbb{R}^{f\times n}$ representing the transition probability from an attribute node to a topology node as follows:

$$\mathbf{S^{(g)}} = \begin{pmatrix} \mathbf{S^{(t)}}_{n\times n} & \mathbf{S^{(a)}}_{n\times f} \\ \mathbf{S^{(c)}}_{f\times n} & \mathbf{0}_{f\times f} \end{pmatrix}_{(n+f)\times(n+f)}. \tag{6}$$

In order to construct matrix $\mathbf{S^{(a)}}$, we first measure the influence of an attribute on a node according to the augmented attribute network by an attribute-to-node influence index.

*Definition 2:* The influence index of attribute $m$ to node $i$ is

$$c_{im} = \frac{n|N(im)|}{e|N(i)|} + \left(1 - \frac{n}{e}\right)\frac{|N(im)|}{|N(m)|} \tag{7}$$

where $e$ is the number of topology edges in the original network, $N(i)$ is the topology neighbor set of topology node $i$, $N(m)$ is the topology neighbor set of attribute node $m$, and $N(im)$ is the intersection of $N(i)$ and $N(m)$. The nodes in $N(im)$ are node $i$'s topology neighbors that share the same value of attribute $m$. $|\cdot|$ returns the number of elements in a set. $|N(im)|/|N(i)|$ and $|N(im)|/|N(m)|$ indicate how many of node $i$'s topology neighbors has attribute value $m$ and how many topology nodes having attribute value $m$ are node $i$'s neighbors, respectively. The former term is not very effective in measuring the influence of attribute $m$ to node $i$ when a network is dense (more edges than nodes). On the contrary, the latter term is not so effective when a network is sparse. Therefore, we employ the ratio of $n/e$ to evaluate the density of a network to coordinate their impact. Specifically,

a high value of $n/e$ means that there are far more nodes than edges, and therefore, the contribution of $|N(im)|/|N(i)|$ should receive more concerns and vice versa. A high value of $c_{im}$ propels the walk from node $i$ to its topology neighbors sharing the same attribute $m$ with it. Matrix $\mathbf{S^{(a)}}$ is constructed based on $c_{im}$ as follows:

$$\mathbf{S^{(a)}}_{im} = \begin{cases} \dfrac{(1-w_t)c_{im}}{\sum_{(i,k)\in E_a} c_{ik}}, & (i,m) \in E_a \\ 0, & (i,m) \notin E_a \end{cases} \tag{8}$$

where $E_a$ is the attribute edge set. A random walk from topology node $i$ to an attribute node based on $\mathbf{S^{(a)}}$ is biased toward the attribute that has the highest influence on it.

In order to construct matrix $\mathbf{S^{(c)}}$, we first define an attribute-weighted degree for each topology node as follows.

*Definition 3:* The attribute-weighted degree of node $i$ is

$$d_i^{w(c)} = w_t d_i^w \frac{1}{d_j} + (1-w_t) \sum_{(i,m)\in E_a} \frac{1}{d_m} \tag{9}$$

where $d_m$ is the degree of attribute node $m$ and $d_i^{w(c)}$ differs from $d_i^w$ by considering the influence of attribute on the random walk from attributes to nodes. Second, we define matrix $\mathbf{S^{(c)}}$ as follows:

$$\mathbf{S^{(c)}}_{mi} = \begin{cases} \dfrac{\sigma\left(\frac{\alpha}{d_i^{w(c)}}\right)}{\sum_{(m,k)\in E_a} \sigma\left(\frac{\alpha}{d_k^{w(c)}}\right)}, & (m,i) \in E_a \\ 0, & (m,i) \notin E_a. \end{cases} \tag{10}$$

Finally, we arrive at a new matrix $\mathbf{Q^{(g)}}$ and a new matrix $\mathbf{M^{(g)}}$ for factorization as follows:

$$\mathbf{Q^{(g)}} = \frac{n}{2T} \sum_{r=1}^{L} \left( \left(\mathbf{S^{(g)}}\right)^r \mathbf{D}_w^{-1} + \mathbf{D}_w^{-1} \left( \left(\mathbf{S^{(g)}}\right)^T \right)^r \right) \tag{11}$$

$$\mathbf{M^{(g)}} = \log \max(\mathbf{Q^{(g)}}, 1) - \log b. \tag{12}$$

Since $\log \mathbf{Q^{(g)}}$ is ill-defined ($\log 0 = -\infty$), we use $\max(\mathbf{Q^{(g)}}, 1)$ to replace $\mathbf{Q}$ in matrix $\mathbf{M}$ according to [5]. After factorizing $\mathbf{M^{(g)}}$, Kmeans is run on the obtained embedding vectors to generate communities. The pseudocode of BRWCD is given in Algorithm 1 and the code of BRWCD is provided at the following link.[1]

### C. Complexity Analysis

Let $n_a = |V_a| + |V|$ be the total number of nodes and $e_a = |E_a| + |E|$ be the total number of edges in an attribute-augmented network. First, the construction of the attribute-augmented network in step 1 requires $O(e_a)$ time. Second, the time complexity of calculating matrices $\mathbf{S^{(t)}}$, $\mathbf{S^{(a)}}$, and $\mathbf{S^{(c)}}$ in step 2 and constructing matrix $\mathbf{S^{(g)}}$ in step 3 is $O(e_a)$. Let $L$ be the length of random walk and $z_r$ be the maximum number of reachable nodes after $r$ hops of random walk from a node. The time for simulating the $L$-hop walks to generate matrix $\mathbf{Q^{(g)}}$ in step 4 and matrix $\mathbf{M^{(g)}}$ in step 5 is $O(\sum_{r=1}^{L} z_1 \times z_r \times n_a)$. In step 6, rSVD factorizes matrix $\mathbf{M}$ in $O(n_a \times q^2)$ time, where $q$ is the dimension of node embedding vectors. Therefore, the total time complexity of BRWCD is $O(e_a + n_a \times q^2 + \sum_{r=1}^{L} z_1 \times z_r \times n_a)$.

[1] https://github.com/Sionzzz/BRWCD

**Algorithm 1** BRWCD

**Input**: Adjacency matrix $\mathbf{A}$; Attribute matrix $\mathbf{X}$.
**Parameters**: Tuning coefficents $\alpha$, $w_t$; Length of walks $L$.
**Output**: Community assignment vector $\mathbf{C}$.

1: Construct the attribute-augmented network $G_a$.
2: Calculate three sub-matrices $\mathbf{S^{(t)}}$, $\mathbf{S^{(a)}}$, $\mathbf{S^{(c)}}$ according to (4), (8) and (10) respectively.
3: Construct the transition probability matrix $\mathbf{S^{(g)}}$ based on $\mathbf{S^{(t)}}$, $\mathbf{S^{(a)}}$ and $\mathbf{S^{(c)}}$ according to (6).
4: Calculate matrix $\mathbf{Q^{(g)}}$ according to (11).
5: Calculate matrix $\mathbf{M^{(g)}}$ according to (12).
6: Factorize $\mathbf{M^{(g)}} = \mathbf{U}\Sigma\mathbf{V}^T$ by rSVD.
7: Obtain node embedding matrix $\mathbf{Z} = \mathbf{U}\sqrt{\Sigma}$.
8: Run Kmeans on $\mathbf{Z}$'s row vectors and build $\mathbf{c} = \{c_i\}$ where $c_i$ is node $i$'s community id.
9: **return** $\mathbf{C}$.

TABLE I
REAL-WORLD NETWORKS. $n$, $e$, $f$, AND $k$ ARE THE NUMBER OF NODES, THE NUMBER OF EDGES, THE DIMENSION OF ATTRIBUTE VECTORS, AND THE NUMBER OF COMMUNITIES, RESPECTIVELY

| Network | $n$ | $e$ | $f$ | $k$ |
|---|---|---|---|---|
| *Cora* | 2708 | 5278 | 1433 | 7 |
| *Citeseer* | 3327 | 4732 | 3708 | 6 |
| *Wiki* | 2405 | 17981 | 4973 | 19 |
| *BlogCatalog* | 5196 | 171743 | 8189 | 6 |
| *Flickr* | 7575 | 239738 | 12047 | 9 |
| *Polbooks* | 105 | 441 | - | 3 |

## V. EXPERIMENTS

In this section, we investigate the effectiveness of BRWCD on real-world and synthetic networks. First, the experiment is conducted to investigate the impact of parameters $w_t$, $L$, and $\alpha$ on BRWCD's accuracy. Second, we perform the accuracy experiment to compare the accuracy of BRWCD with the baseline algorithms. Finally, an ablation study is conducted on the result of the accuracy experiment to reveal the effectiveness of the two weighted degrees and the attribute-to-node influence index.

### A. Datasets

Six well-known real-world networks with ground-truth communities and five synthetic networks are used to evaluate the algorithms' performance.

*1) Real-World Networks:* Table I describes the real-world networks, where $n$ and $e$ are the number of nodes and the number of edges, respectively, $f$ denotes the dimension of attribute vectors, and $k$ represents the number of communities.

1) Cora [43] and Citeseer [43] are both representative citation networks with their nodes representing papers. The edges are formed by the citation relationships between papers. A set of keywords are selected from the papers to

TABLE II

SYNTHETIC NETWORKS. $n$, $e$, $f$, AND $k$ ARE THE NUMBER OF NODES, THE NUMBER OF EDGES, THE DIMENSION OF ATTRIBUTE VECTORS, AND THE NUMBER OF COMMUNITIES, RESPECTIVELY

| Network | $n$ | $e$ | $f$ | $k$ |
|---|---|---|---|---|
| *D1* | 2000 | 5453 | 1591 | 53 |
| *D2* | 4000 | 11274 | 3061 | 102 |
| *D3* | 6000 | 16141 | 4531 | 151 |
| *D4* | 8000 | 22436 | 6271 | 209 |
| *D5* | 10000 | 27928 | 8491 | 283 |

generate binary attribute vectors. The value of a node's attribute denoting a word is set to 0 if the word is not present in this article corresponding to the node; otherwise, it is set to 1.

2) Wiki [12] is a document network the attributes of which store the TF-IDF values of the words in the documents.
3) BlogCatalog [44] is a blog network. Its nodes represent user blogs and the interactions between users constitute the edges. The binary attributes of a node represent the presence of certain words in the blog corresponding to the node.
4) Flickr [44] is an online social network with its nodes representing users in Flickr. Users share photographs and follow each other, which constitutes the edges. The binary attributes are generated according to the presence of words in tags of shared images.
5) Polbooks [41] is a small-scale network without attributes. Its nodes represent political books in Amazon and books are classified according to political affiliation. Two nodes are connected by an edge if many customers buy these two books corresponding to the nodes simultaneously.

*2) Synthetic Networks:* Table II describes five synthetic networks with varying numbers of nodes generated by the LFR benchmark [45]. The settings of the additional parameters are: the average degree avgd and maximum degree maxd are set to 10 and 50, respectively, the minimum community size minc and maximum community size maxc are set to 10 and 100, respectively, and the mixing parameter $\mu$, which controls the percentage of a nodes neighboring edges that link to the nodes outside the community to which it belongs, is set to 0.5. A higher value of $\mu$ means a network with more blurry communities. A value of 0.5 is used to simulate networks with highly mixed communities. The binary node attributes are generated based on the methods proposed in [24] and [46] to guarantee that the nodes in the same community have similar attribute values, while the nodes in different communities are more likely to have dissimilar ones.

### *B. Baselines*

We compare BRWCD with 18 state-of-the-art algorithms: DeepWalk, LINE, Node2Vec, vGraph, NECS, ComE, CommunityGAN, NetMF, ARGA, ARVGA, TADW, AANE, RoSANE, SCI, DGI, GRACE, GCA, and CFANE. The first eight algorithms handle only topology. The rest algorithms can handle topology and attributes.

1) DeepWalk [4] learns embedding vectors based on topological random walk to extract high-order relationships.
2) LINE [47] embeds the first- and second-order proximities in the optimization of the skip-gram model.
3) Node2Vec [48] employs a biased random walk strategy to explore a node's neighbors both in breadth and depth according to its parameters $p$ and $q$, respectively.
4) vGraph [32] is a single-staged probabilistic generative method that directly obtains the community memberships of nodes.
5) NECS [34] learns node embedding vectors preserving high-order proximity as well as each node's community membership.
6) ComE [28] employs a multidimensional Gaussian distribution to learn community embedding vectors and detect communities.
7) CommunityGAN [33] optimizes the membership of each node to communities through the competition between a clique-level generator and a discriminator.
8) NetMF [5] factorizes DeepWalk's implicit matrix by SVD.
9) ARGA [49] uses adversarially regularized GAE for network embedding.
10) ARVGA [49] is a variant of ARGA by replacing GAE with VGAE.
11) TADW [12] adds an attribute matrix in the factorization of the defined word-context matrix.
12) AANE [19] employs an attribute similarity matrix and factorizes the matrix parallelly.
13) RoSANE [25] samples node sequences according to attributes similarity and network topology and employs the skip-gram model to learn node embedding vectors.
14) SCI [36] is a single-staged method that uses the NMF technique to combine topology and attributes and directly obtain community memberships of nodes.
15) DGI [50] maximizes the mutual information between node embedding vectors and high-order local structures of a network to learn node embedding vectors.
16) GRACE [51] employs a node feature masking and edge dropping strategy to contrast node embedding vectors with node feature vectors to refine the former ones.
17) GCA [18] extends the idea of GRACE by introducing an adaptive augmentation strategy to keep important node features unmasked.
18) CFANE [16] is a GAE-based method preserving local community structure and uses integrated fusion layers to exchange information between topology and attributes.

In addition, two variants of BRWCD are involved in comparison in the ablation study. BRWCD-T is a simplified BRWCD considering only the topology-weighted degree by replacing $\mathbf{Q^{(g)}}$ in (12) with $\mathbf{Q^{(t)}}$. BRWCD-A is a simplified BRWCD considering only the attribute-to-node influence index and the attribute-weighted degree by replacing $\mathbf{Q^{(g)}}$ in (12) with the following $\mathbf{Q^{(a)}}$:

$$\mathbf{Q^{(a)}} = \frac{n}{2T}\sum_{r=1}^{L}\left(\left(\mathbf{S^{(a)}}\right)^r\mathbf{D}_w^{-1} + \mathbf{D}_w^{-1}\left(\left(\mathbf{S}\right)^T\right)^r\right) \tag{13}$$

$$\mathbf{S}^{(\mathbf{a})} = \begin{pmatrix} \mathbf{D}^{-1}\mathbf{A} & \mathbf{S}^{(\mathbf{a})} \\ \mathbf{S}^{(\mathbf{c})} & \mathbf{0} \end{pmatrix}. \tag{14}$$

### C. Evaluation Metrics

All algorithms are evaluated by three widely used metrics [52]: NMI, adjusted Rand index (ARI), and micro F1-score (F1). The greater the values of the metrics, the higher the accuracy of an algorithm is.

Given two partitions $A = \{A_1, \ldots, A_a\}$ and $B = \{B_1, \ldots, B_b\}$ of $V$, NMI is defined as follows:

$$\mathrm{NMI}(A, B) = \frac{-2\sum_{i=1}^{a}\sum_{j=1}^{b}\mathbf{C}_{ij}\log\left(\mathbf{C}_{ij}n/\mathbf{C}_i\mathbf{C}_j\right)}{\sum_{i=1}^{a}\mathbf{C}_i\log(\mathbf{C}_i/n) + \sum_{j=1}^{b}\mathbf{C}_j\log\left(\mathbf{C}_j/n\right)} \tag{15}$$

where $\mathbf{C}_{ij}$ denotes the number of the overlapping nodes between $A_i$ and $B_j$ and $\mathbf{C}_i$ and $\mathbf{C}_j$ are the sums over row $i$ and column $j$ of $\mathbf{C}$, respectively.

ARI and F1-score evaluate the matches between the predicted communities and the ground-truth communities. Before introducing them, the following definitions are necessary.

*Definition 4 (True Positive (TP)):* This shows the number of nodes in the same predicted community and also in the same ground-truth community.

*Definition 5 (True Negative (TN)):* This shows the number of nodes in different predicted communities and in different ground-truth communities.

*Definition 6 (False Positive (FP)):* This shows the number of nodes in the same predicted community but not in the same ground-truth community.

*Definition 7 (False Negative (FN)):* This shows the number of nodes in different predicted communities but in the same ground-truth community.

The Rand index (RI) and ARI are defined as follows:

$$\mathrm{RI} = \frac{\mathrm{TP} + \mathrm{TN}}{\mathrm{TP} + \mathrm{FP} + \mathrm{FN} + \mathrm{TN}} \tag{16}$$

$$\mathrm{ARI} = \frac{\mathrm{RI} - E(\mathrm{RI})}{\max(\mathrm{RI}) - E(\mathrm{RI})} \tag{17}$$

where $E(\mathrm{RI})$ and $\max(\mathrm{RI})$ denote the expected and maximal values of RI, respectively.

Two indices, namely, precision and recall, can be defined based on TP, FP, and FN as follows.

*Definition 8 (Precision):*

$$\mathrm{Precision} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FP}}. \tag{18}$$

*Definition 9 (Recall):*

$$\mathrm{Recall} = \frac{\mathrm{TP}}{\mathrm{TP} + \mathrm{FN}}. \tag{19}$$

Therefore, we can define F1-score (F1) as follows:

$$\mathrm{F1} = \frac{2 \cdot \mathrm{Precision} \cdot \mathrm{Recall}}{\mathrm{Precision} + \mathrm{Recall}}. \tag{20}$$

### D. Parameter Settings

For a fair comparison, we use the default parameter values of the algorithms suggested in their original papers or search for the optimal parameter values if possible to achieve their best performance. The detailed parameter settings are given as follows.

1) *ARGA, ARVGA, vGraph, CommunityGAN, and DGI:* The default parameter values suggested in their original papers are used.
2) *DeepWalk, LINE, Node2Vec, NetMF, and BRWCD-T:* The number of walks from each node is set to 80, the walk length is set to 40, and the window size is set to 10. For Node2Vec, the optimal values of $p$ and $q$ are searched in {0.25, 0.5, 1, 2, 4}.
3) *RoSANE, AANE, BRWCD-A, and BRWCD:* The optimal value of the parameter balancing topology and attributes is searched in (0, 1). For RoSANE, the number of walks from each node is set to 80 and the walk length is set to 40. The window size of RoSANE, BRWCD-A, and BRWCD is set to 5. The constant $b$ in (12) is set to 1 and the parameter $\alpha$ is set to 15 for BRWCD and BRWCD-A.
4) *NECS:* The optimal values of $l$ and $w$ are searched in 2, 3 and 0, 0.1, 0.2, 0.3, respectively. The optimal values of parameters $\alpha$ and $\beta$ are searched in 0.1, 0.5, 1, 5, 10.
5) *ComE:* The optimal values of parameters $\alpha$ and $\beta$ are searched in {0.1, 1, 5, 10}.
6) *TADW:* The optimal value of the parameter balancing topology and attributes is searched in (0, 10).
7) *SCI:* The optimal values of parameters $\alpha$ and $\beta$ are searched in (1, 100).
8) *GRACE, GCA:* The optimal values of the parameters determining the generation of graph views are searched in {0.0, 0.2, 0.4}.
9) *CFANE:* The optimal values of $p$ and $q$ are searched in the same manner as in Node2Vec. The optimal value of $\beta$ is searched in {0.5, 1, 2}.

The common parameters, including learning rate, batch size, and max number of iterations, are set to the default values as suggested in the original papers. The dimension of embedding vectors is set to 128 for all the algorithms. For all two-staged algorithms, Kmeans is used to cluster node embedding vectors into communities. The value of Kmeans' parameter $k$ is set to the number of ground-truth communities for all networks. The average accuracy obtained from 20 runs of an algorithm on the same network is reported. BRWCD is implemented in Python, while the source code of other algorithms is downloaded from the addresses provided by their authors. All experiments are conducted on a PC with AMD Ryzen 7 4700U (2.00 GHz) and 16-GB RAM.

### E. Parameter Experiment

We investigate the effect of the three parameters of BRWCD, namely, $\alpha$, $w_t$, and $L$, on real-world and synthetic networks. First, we set $w_t = 0.5$ and $L = 5$ and vary the value of $\alpha$ from 5 to 50. The NMI values remain stable as the value of $\alpha$ changes, as shown in Fig. 3(a) and (d). Second,

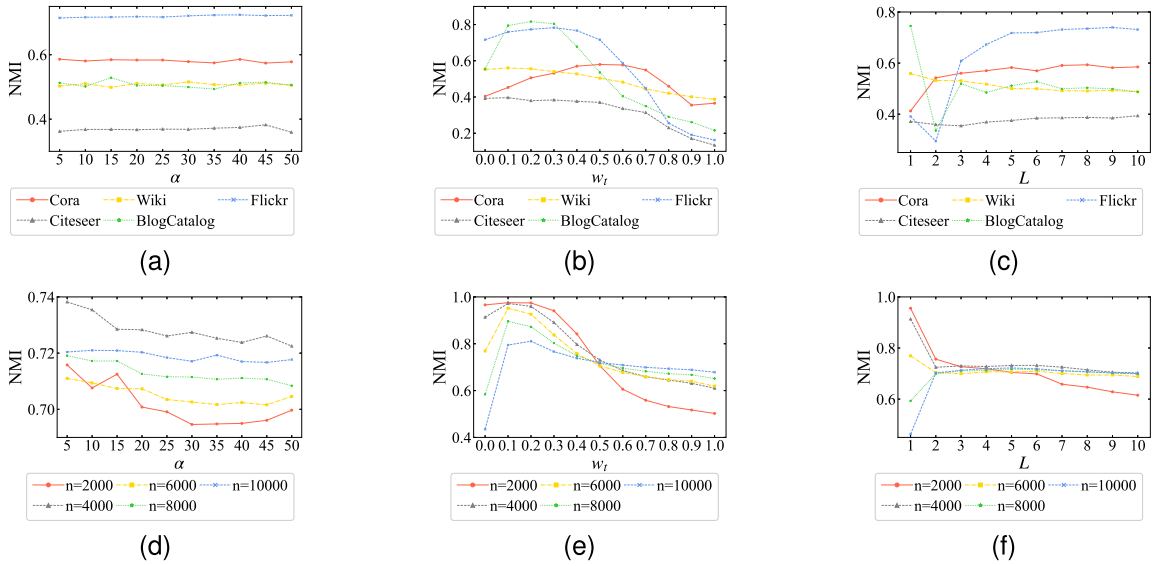


Fig. 3. Results of parameter experiment. $\alpha$ and $w_t$ are tuning coefficients used in (4). $L$ is the length of a random walk. (a) $\alpha$ on real-world networks. (b) $w_t$ on real-world networks. (c) $L$ on real-world networks. (d) $\alpha$ on synthetic networks. (e) $w_t$ on synthetic networks. (f) $L$ on synthetic networks.

we set $L = 5$ and $\alpha = 15$ and vary the value of $w_t$ from 0.0 to 1.0. As shown in Fig. 3(b) and (e), BRWCD achieves better performance when $w_t \leq 0.4$ because attributes are of vital importance for effective community recognition on attributed networks. The sharp decrease of the NMI values on BlogCatalog and Flickr with the increase of the value of $w_t$ demonstrates that attributes have a greater influence on BRWCD than topology on BlogCatalog and Flickr. The parameters of synthetic networks are identical except network size and the attributes of them are generated in the same manner. Therefore, the NMI values on synthetic networks have similar variation trends. The performance of BRWCD changes violently with varying values of $w_t$. However, lower $w_t$ values always correspond to high accuracy. Third, we set $w_t = 0.5$ and $\alpha = 15$ and vary the value of $L$ from 1 to 10. As shown in Fig. 3(c) and (f), the accuracy of BRWCD keeps stable when $L \geq 4$. In the networks with small diameters, namely, the synthetic networks ($n = 2000$, $n = 4000$, and $n = 6000$) and the BlogCatalog and Flickr networks, many nodes in a community are only one hop away from each other. Therefore, random walks from a node may go beyond the boundary of a community when $L = 2$ and back to the community with the increase of the value of $L$. This is why BRWCD's accuracy is high when $L = 1$ and drops sharply when $L = 2$. According to the experimental result, we set $\alpha = 15$, $w_t \in [0.1, 0.4]$, and $L = 5$ in the remaining experiments.

### *F. Accuracy Experiment*

*1) Comparison With Baselines:* We evaluate the accuracy of BRWCD with that of the baselines. Kmeans is used to obtain communities for the two-staged algorithms generating only node embedding vectors. The average accuracy obtained from 20 runs on real-world and synthetic networks is shown in Sections III and IV, respectively. The best and second-best accuracy values are shown in bold font and underline, respectively.

As shown in Tables III and IV, most methods capable of handling topology and attributes perform better than those can handle only topology. The significant accuracy difference between the two types of methods on BlogCatalog and Flickr demonstrates that the topology of BlogCatalog and Flickr is not so beneficial to community detection. For methods considering only topology, Node2Vec, ComE, and BRWCD-T surpass other methods on all real-world and synthetic networks except Flickr. The possible reason is that both Node2Vec and BRWCD-T adopt biased random walks to capture potential community structure, and ComE learns node embedding vectors and detects communities simultaneously. However, Node2Vec and ComE need several fine-tuned parameters to achieve optimal accuracy. BRWCD-T can use fixed parameters on different networks to obtain competitive performance, which reveals the stability and superiority of our method in capturing community structure based on topology-weighted degree. In addition, BRWCD outperforms the baselines on most real-world networks. Specifically, its NMI values increase by 3.2%, 0.7%, 4.9%, 4.5%, and 3.0% over the runner-up baselines on Cora, Citeseer, Wiki, BlogCatalog, and Flickr, respectively. The communities on BlogCatalog and Flickr are mainly constructed according to node attributes. Therefore, the experimental results demonstrate that the biased random walk considering attribute influence can fully utilize attributes in community detection. BRWCD achieves a slightly lower ARI value than RoSANE on Citeseer, which may be due to the fact that Citeseer is a fragmented network with small connected components and ARI is particularly sensitive to clusters on fragments. On synthetic networks, BRWCD surpasses all baselines by at least 3.4%. BRWCD-A outperforms all other methods except for BRWCD because it can recognize the subtle difference in the influence of attributes on community detection. TADW, RoSANE, and

TABLE III

RESULTS OF ACCURACY EXPERIMENT ON REAL-WORLD NETWORKS. THE HIGHEST AND SECOND HIGHEST VALUES ARE DISPLAYED IN BOLD FONT AND UNDERLINED, RESPECTIVELY

| Algorithms | Cora | | | Citeseer | | | Wiki | | | BlogCatalog | | | Flickr | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | NMI | ARI | F1 | NMI | ARI | F1 | NMI | ARI | F1 | NMI | ARI | F1 | NMI | ARI | F1 |
| DeepWalk | 0.370 | 0.269 | 0.492 | 0.126 | 0.134 | 0.386 | 0.350 | 0.187 | 0.358 | 0.175 | 0.086 | 0.378 | 0.171 | 0.107 | 0.319 |
| LINE | 0.250 | 0.149 | 0.453 | 0.078 | 0.035 | 0.284 | 0.335 | 0.151 | 0.362 | 0.229 | 0.152 | 0.414 | 0.141 | 0.069 | 0.289 |
| Node2Vec | 0.434 | 0.304 | 0.612 | 0.251 | 0.176 | 0.468 | 0.379 | 0.208 | 0.380 | 0.210 | 0.113 | 0.382 | 0.190 | 0.121 | 0.343 |
| vGraph | 0.094 | 0.058 | 0.297 | 0.080 | 0.063 | 0.262 | 0.263 | 0.131 | 0.304 | 0.169 | 0.097 | 0.371 | 0.149 | 0.043 | 0.263 |
| NECS | 0.136 | 0.095 | 0.304 | 0.011 | 0.009 | 0.316 | 0.246 | 0.103 | 0.275 | 0.089 | 0.008 | 0.297 | 0.079 | 0.036 | 0.214 |
| ComE | 0.437 | 0.316 | 0.615 | 0.248 | 0.126 | 0.471 | 0.372 | 0.200 | 0.384 | 0.217 | 0.104 | 0.372 | 0.179 | 0.109 | 0.338 |
| CommunityGAN | 0.070 | 0.043 | 0.311 | 0.009 | 0.003 | 0.281 | 0.115 | 0.003 | 0.172 | 0.073 | 0.004 | 0.278 | 0.023 | 0.003 | 0.196 |
| NetMF | 0.362 | 0.164 | 0.527 | 0.157 | 0.022 | 0.364 | 0.365 | 0.170 | 0.342 | 0.187 | 0.065 | 0.359 | 0.121 | 0.058 | 0.271 |
| BRWCD-T | 0.445 | 0.324 | 0.620 | 0.171 | 0.081 | 0.403 | 0.398 | 0.212 | 0.400 | 0.219 | 0.115 | 0.383 | 0.128 | 0.067 | 0.285 |
| ARGA | 0.500 | 0.441 | 0.646 | 0.268 | 0.189 | 0.346 | 0.418 | 0.190 | 0.181 | 0.269 | 0.187 | 0.458 | 0.141 | 0.063 | 0.259 |
| ARVGA | 0.493 | 0.409 | 0.635 | 0.351 | 0.347 | 0.510 | 0.361 | 0.175 | 0.076 | 0.258 | 0.141 | 0.419 | 0.120 | 0.075 | 0.270 |
| TADW | 0.539 | 0.428 | 0.677 | 0.394 | 0.360 | 0.592 | 0.349 | 0.172 | 0.372 | 0.553 | 0.521 | 0.720 | 0.408 | 0.333 | 0.556 |
| AANE | 0.245 | 0.149 | 0.437 | 0.303 | 0.270 | 0.537 | 0.475 | 0.127 | 0.401 | 0.354 | 0.275 | 0.554 | 0.123 | 0.087 | 0.249 |
| RoSANE | 0.542 | 0.486 | 0.690 | 0.401 | **0.413** | 0.613 | 0.413 | 0.208 | 0.387 | 0.772 | 0.796 | 0.906 | 0.757 | 0.742 | 0.863 |
| SCI | 0.199 | 0.144 | 0.357 | 0.104 | 0.084 | 0.326 | 0.270 | 0.134 | 0.289 | 0.414 | 0.302 | 0.582 | 0.266 | 0.199 | 0.494 |
| DGI | 0.541 | 0.458 | 0.652 | 0.253 | 0.253 | 0.515 | 0.313 | 0.151 | 0.302 | 0.191 | 0.102 | 0.384 | 0.176 | 0.076 | 0.271 |
| GRACE | 0.363 | 0.168 | 0.548 | 0.276 | 0.181 | 0.490 | 0.417 | 0.208 | 0.382 | 0.189 | 0.124 | 0.400 | 0.169 | 0.163 | 0.296 |
| GCA | 0.497 | 0.416 | 0.643 | 0.384 | 0.353 | 0.500 | 0.421 | 0.194 | 0.436 | 0.182 | 0.108 | 0.385 | 0.164 | 0.118 | 0.326 |
| CFANE | 0.556 | 0.504 | 0.708 | 0.390 | 0.372 | 0.596 | 0.515 | 0.346 | 0.472 | 0.617 | 0.631 | 0.820 | 0.145 | 0.093 | 0.290 |
| BRWCD-A | 0.552 | 0.455 | 0.698 | 0.388 | 0.363 | 0.581 | 0.545 | 0.344 | 0.481 | 0.803 | 0.829 | 0.921 | 0.780 | 0.777 | 0.878 |
| BRWCD | **0.588** | **0.519** | **0.712** | **0.408** | 0.394 | **0.618** | **0.564** | **0.383** | **0.506** | **0.817** | **0.836** | **0.924** | **0.787** | **0.785** | **0.899** |

TABLE IV

RESULTS OF ACCURACY EXPERIMENT ON SYNTHETIC NETWORKS. THE HIGHEST AND SECOND HIGHEST VALUES ARE DISPLAYED IN BOLD FONT AND UNDERLINED, RESPECTIVELY

| Algorithms | n=2000 | | | n=4000 | | | n=6000 | | | n=8000 | | | n=10000 | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | NMI | ARI | F1 | NMI | ARI | F1 | NMI | ARI | F1 | NMI | ARI | F1 | NMI | ARI | F1 |
| DeepWalk | 0.448 | 0.169 | 0.354 | 0.573 | 0.274 | 0.459 | 0.592 | 0.263 | 0.453 | 0.634 | 0.296 | 0.475 | 0.667 | 0.311 | 0.486 |
| LINE | 0.250 | 0.017 | 0.339 | 0.303 | 0.013 | 0.366 | 0.312 | 0.011 | 0.336 | 0.343 | 0.013 | 0.339 | 0.361 | 0.008 | 0.348 |
| Node2Vec | 0.512 | 0.254 | 0.446 | 0.596 | 0.304 | 0.474 | 0.597 | 0.268 | 0.448 | 0.635 | 0.291 | 0.459 | 0.670 | 0.304 | 0.480 |
| vGraph | 0.202 | 0.008 | 0.128 | 0.264 | 0.009 | 0.133 | 0.280 | 0.012 | 0.157 | 0.313 | 0.011 | 0.166 | 0.356 | 0.013 | 0.169 |
| NECS | 0.183 | 0.002 | 0.084 | 0.243 | 0.003 | 0.098 | 0.281 | 0.007 | 0.109 | 0.315 | 0.011 | 0.115 | 0.359 | 0.014 | 0.158 |
| ComE | 0.496 | 0.220 | 0.430 | 0.589 | 0.286 | 0.462 | 0.599 | 0.259 | 0.449 | 0.636 | 0.288 | 0.458 | 0.671 | 0.301 | 0.481 |
| CommunityGAN | 0.142 | 0.003 | 0.080 | 0.138 | 0.001 | 0.055 | 0.182 | 0.005 | 0.080 | 0.224 | 0.009 | 0.098 | 0.243 | 0.012 | 0.112 |
| NetMF | 0.480 | 0.169 | 0.427 | 0.607 | 0.302 | 0.510 | 0.620 | 0.292 | 0.494 | 0.651 | 0.307 | 0.491 | 0.677 | 0.311 | 0.514 |
| BRWCD-T | 0.494 | 0.209 | 0.446 | 0.614 | 0.317 | 0.516 | 0.626 | 0.299 | 0.503 | 0.656 | 0.319 | 0.498 | 0.686 | 0.326 | 0.503 |
| ARGA | 0.482 | 0.186 | 0.306 | 0.396 | 0.068 | 0.168 | 0.338 | 0.022 | 0.103 | 0.311 | 0.005 | 0.078 | 0.347 | 0.005 | 0.087 |
| ARVGA | 0.385 | 0.094 | 0.203 | 0.297 | 0.016 | 0.101 | 0.302 | 0.007 | 0.079 | 0.313 | 0.004 | 0.074 | 0.363 | 0.004 | 0.076 |
| TADW | 0.942 | 0.896 | 0.850 | 0.914 | 0.838 | 0.771 | 0.855 | 0.720 | 0.674 | 0.796 | 0.591 | 0.575 | 0.715 | 0.399 | 0.484 |
| AANE | 0.910 | 0.785 | 0.722 | 0.837 | 0.711 | 0.582 | 0.658 | 0.501 | 0.350 | 0.494 | 0.145 | 0.181 | 0.363 | 0.012 | 0.075 |
| RoSANE | 0.943 | 0.884 | 0.823 | 0.903 | 0.800 | 0.719 | 0.800 | 0.621 | 0.585 | 0.723 | 0.454 | 0.504 | 0.679 | 0.329 | 0.459 |
| SCI | 0.775 | 0.676 | 0.561 | 0.843 | 0.722 | 0.605 | 0.825 | 0.682 | 0.645 | 0.865 | 0.748 | 0.647 | 0.787 | 0.538 | 0.537 |
| DGI | 0.219 | 0.120 | 0.183 | 0.266 | 0.006 | 0.169 | 0.312 | 0.018 | 0.369 | 0.340 | 0.024 | 0.362 | 0.382 | 0.072 | 0.362 |
| GRACE | 0.394 | 0.152 | 0.315 | 0.370 | 0.063 | 0.164 | 0.417 | 0.126 | 0.323 | 0.433 | 0.213 | 0.316 | 0.427 | 0.211 | 0.271 |
| GCA | 0.451 | 0.167 | 0.355 | 0.418 | 0.106 | 0.196 | 0.426 | 0.132 | 0.339 | 0.436 | 0.236 | 0.312 | 0.431 | 0.226 | 0.286 |
| CFANE | 0.607 | 0.370 | 0.369 | 0.360 | 0.068 | 0.145 | 0.318 | 0.011 | 0.082 | 0.330 | 0.004 | 0.066 | 0.365 | 0.002 | 0.061 |
| BRWCD-A | 0.980 | 0.917 | 0.913 | 0.974 | 0.907 | 0.880 | 0.948 | 0.873 | 0.810 | 0.890 | 0.772 | 0.679 | 0.804 | 0.564 | 0.614 |
| BRWCD | **0.986** | **0.921** | **0.917** | **0.981** | **0.915** | **0.882** | **0.952** | **0.876** | **0.818** | **0.906** | **0.778** | **0.688** | **0.821** | **0.576** | **0.678** |

CFANE perform well on real-world networks, but they are inferior to BRWCD-A and BRWCD on synthetic networks. The experimental result demonstrates the superiority of the biased random walk toward boundary nodes and the consideration of attribute-to-node influence in BRWCD. The baselines either consider only network topology or ignore the influence of different attributes on the random walk when considering network topology and attributes, leading to their suboptimal performance.

*2) Ablation Study:* We analyze the result in Tables III and IV to study the effectiveness of the topology-weighted degree and the attribute-to-node influence and the attribute-weighted degree, respectively. The focus is on the NMI values because the accuracy measured by the other two metrics is similar.

When the attribute-to-node influence and the attribute-weighted degree are not considered, the NMI values of BRWCD-T are 10.7%, 21.7%, 14.7%, 58.4%, and 65.2% lower than BRWCD on Cora, Citeseer, Wiki, BlogCatalog, and Flickr, respectively, which strongly proves the importance of distinguishing the influence of different attributes on the random walk on nodes. When the topology-weighted degree is not considered, the NMI values of BRWCD-A are 3.6%, 2.0%, 1.9%, 1.4%, and 0.7% lower than BRWCD on Cora, Citeseer, Wiki, BlogCatalog, and Flickr, respectively, which indicates that the biased random walk considering topology-weighted are beneficial to learn better node embedding vectors for clustering.

## VI. CONCLUSION

In this article, we propose a novel biased random walk algorithm BRWCD to detect communities in attributed networks. We first design a topology-weighted degree to take the nodes at center and boundary of each community into

account in node embedding learning. Second, an attribute-to-node influence index and an attribute-weighted degree are designed to consider the influence of attributes in random walks on nodes. The experimental results on real-world and synthetic networks verify the effectiveness of BRWCD. In the future, we will first study the methods to determine BRWCDs parameters automatically and explore more random walk strategies. Second, we will study the integration of our random walk strategies with more network embedding methods such as the one-staged graph neural networks.

## REFERENCES


[1] M. A. Javed, M. S. Younis, S. Latif, J. Qadir, and A. Baig, "Community detection in networks: A multidisciplinary review," *J. Netw. Comput. Appl.*, vol. 108, pp. 87–111, Apr. 2018.

[2] A. Markovitz, G. Sharir, I. Friedman, L. Zelnik-Manor, and S. Avidan, "Graph embedded pose clustering for anomaly detection," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 10539–10547.

[3] J. Chen *et al.*, "Self-training enhanced: Network embedding and overlapping community detection with adversarial learning," *IEEE Trans. Neural Netw. Learn. Syst.*, early access, Jun. 10, 2021, doi: 10.1109/TNNLS.2021.3083318.

[4] B. Perozzi, R. Al-Rfou, and S. Skiena, "DeepWalk: Online learning of social representations," in *Proc. 20th Int. Conf. Knowl. Discovery Data Mining*, Aug. 2014, pp. 701–710.

[5] J. Qiu, Y. Dong, H. Ma, J. Li, K. Wang, and J. Tang, "Network embedding as matrix factorization: Unifying DeepWalk, LINE, PTE, and node2vec," in *Proc. 11th ACM Int. Conf. Web Search Data Mining*, Feb. 2018, pp. 459–467.

[6] T. Sarlos, "Improved approximation algorithms for large matrices via random projections," in *Proc. 47th Annu. IEEE Symp. Found. Comput. Sci. (FOCS)*, Oct. 2006, pp. 143–152.

[7] I. Ben El Kouni, W. Karoui, and L. B. Romdhane, "Node importance based label propagation algorithm for overlapping community detection in networks," *Expert Syst. Appl.*, vol. 162, Dec. 2020, Art. no. 113020.

[8] C. Wang, C. Hao, and X. Guan, "Hierarchical and overlapping social circle identification in ego networks based on link clustering," *Neurocomputing*, vol. 381, pp. 322–335, Mar. 2020.

[9] K. Berahmand, A. Bouyer, and M. Vasighi, "Community detection in complex networks by detecting and expanding core nodes through extended local similarity of nodes," *IEEE Trans. Comput. Soc. Syst.*, vol. 5, no. 4, pp. 1021–1033, Dec. 2018.

[10] K. Berahmand, E. Nasiri, R. P. Mohammadiani, and Y. Li, "Spectral clustering on protein-protein interaction networks via constructing affinity matrix using attributed graph embedding," *Comput. Biol. Med.*, vol. 138, Nov. 2021, Art. no. 104933.

[11] K. Berahmand, M. Mohammadi, A. Faroughi, and R. P. Mohammadiani, "A novel method of spectral clustering in attributed networks by constructing parameter-free affinity matrix," *Cluster Comput.*, vol. 25, no. 2, pp. 1–20, 2021.

[12] C. Yang, Z. Liu, D. Zhao, M. Sun, and E. Chang, "Network representation learning with rich text information," in *Proc. 24th Int. Conf. Artif. Intell.*, 2015, pp. 2111–2117.

[13] A. C. Ramesh and G. Srivatsun, "Evolutionary algorithm for overlapping community detection using a merged maximal cliques representation scheme," *Appl. Soft Comput.*, vol. 112, Nov. 2021, Art. no. 107746.

[14] M. Chen, K. Kuzmin, and B. K. Szymanski, "Community detection via maximization of modularity and its variants," *IEEE Trans. Comput. Soc. Syst.*, vol. 1, no. 1, pp. 46–65, Mar. 2014.

[15] D. He, Z. Feng, D. Jin, X. Wang, and W. Zhang, "Joint identification of network communities and semantics via integrative modeling of network topologies and node contents," in *Proc. 31st AAAI Conf. Artif. Intell.*, 2017, pp. 116–124.

[16] G. Pan, Y. Yao, H. Tong, F. Xu, and J. Lu, "Unsupervised attributed network embedding via cross fusion," in *Proc. 14th ACM Int. Conf. Web Search Data Mining*, Mar. 2021, pp. 797–805.

[17] H. Sun *et al.*, "Network embedding for community detection in attributed networks," *ACM Trans. Knowl. Discovery Data*, vol. 14, no. 3, pp. 1–25, May 2020.

[18] Y. Zhu, Y. Xu, F. Yu, Q. Liu, S. Wu, and L. Wang, "Graph contrastive learning with adaptive augmentation," in *Proc. Web Conf.*, Apr. 2021, pp. 2069–2080.

[19] X. Huang, J. Li, and X. Hu, "Accelerated attributed network embedding," in *Proc. Int. Conf. Data Mining*, 2017, pp. 633–641.

[20] S. Bandyopadhyay, A. Biswas, H. Kara, and M. Murty, "A multilayered informative random walk for attributed social network embedding," in *Proc. ECAI*, 2020, pp. 1738–1745.

[21] D. He *et al.*, "Adversarial mutual information learning for network embedding," in *Proc. 29th Int. Joint Conf. Artif. Intell.*, Jul. 2020, pp. 3321–3327.

[22] S. Bandyopadhyay, N. Lokesh, and M. N. Murty, "Outlier aware network embedding for attributed networks," in *Proc. AAAI*, vol. 33, 2019, pp. 12–19.

[23] K. Berahmand *et al.*, "A modified DeepWalk method for link prediction in attributed social network," *Computing*, vol. 103, no. 10, pp. 2227–2249, Oct. 2021.

[24] X. Huang, Q. Song, Y. Li, and X. Hu, "Graph recurrent networks with attributed random walks," in *Proc. 25th Int. Conf. Knowl. Discovery Data Mining*, Jul. 2019, pp. 732–740.

[25] C. Hou, S. He, and K. Tang, "RoSANE: Robust and scalable attributed network embedding for sparse networks," *Neurocomputing*, vol. 409, pp. 231–243, Oct. 2020.

[26] Y. Gao, M. Gong, Y. Xie, and H. Zhong, "Community-oriented attributed network embedding," *Knowl.-Based Syst.*, vol. 193, Apr. 2020, Art. no. 105418.

[27] D. M. Blei, A. Y. Ng, and M. I. Jordan, "Latent Dirichlet allocation," *J. Mach. Learn. Res.*, vol. 3, pp. 993–1022, Mar. 2003.

[28] S. Cavallari, V. W. Zheng, H. Cai, K. C.-C. Chang, and E. Cambria, "Learning community embedding with community detection and node embedding on graphs," in *Proc. ACM Conf. Inf. Knowl. Manage.*, Nov. 2017, pp. 377–386.

[29] X. Wang, P. Cui, J. Wang, J. Pei, W. Zhu, and S. Yang, "Community preserving network embedding," in *Proc. 31st AAAI Conf. Artif. Intell.*, 2017, pp. 203–209.

[30] C. Tu *et al.*, "A unified framework for community detection and network representation learning," *IEEE Trans. Knowl. Data Eng.*, vol. 31, no. 6, pp. 1051–1065, Jun. 2018.

[31] Y. Zhang, T. Lyu, and Y. Zhang, "COSINE: Community-preserving social network embedding from information diffusion cascades," in *Proc. AAAI Conf. Artif. Intell.*, 2018, vol. 32, no. 1, pp. 2620–2627.

[32] F.-Y. Sun, M. Qu, J. Hoffmann, C.-W. Huang, and J. Tang, "VGraph: A generative model for joint community detection and node representation learning," 2019, *arXiv:1906.07159*.

[33] Y. Jia, Q. Zhang, W. Zhang, and X. Wang, "CommunityGAN: Community detection with generative adversarial nets," in *Proc. World Wide Web Conf.*, May 2019, pp. 784–794.

[34] Y. Li, Y. Wang, T. Zhang, J. Zhang, and Y. Chang, "Learning network embedding with community structural information," in *Proc. 28th Int. Joint Conf. Artif. Intell.*, Aug. 2019, pp. 2937–2943.

[35] Y. Pei, N. Chakraborty, and K. Sycara, "Nonnegative matrix tri-factorization with graph regularization for community detection in social networks," in *Proc. 24th Int. Conf. Artif. Intell.*, 2015, pp. 2083–2089.

[36] X. Wang, D. Jin, X. Cao, L. Yang, and W. Zhang, "Semantic community identification in large attribute networks," in *Proc. AAAI Conf. Artif. Intell.*, 2016, vol. 30, no. 1, pp. 265–271.

[37] M. Qin, D. Jin, K. Lei, B. Gabrys, and K. Musial-Gabrys, "Adaptive community detection incorporating topology and content in social networks," *Knowl. Based Syst.*, vol. 161, pp. 342–356, Dec. 2018.

[38] Y. Li, C. Sha, X. Huang, and Y. Zhang, "Community detection in attributed graphs: An embedding approach," in *Proc. 32nd AAAI Conf. Artif. Intell.*, 2018, pp. 338–345.

[39] Z. Huang, X. Zhong, Q. Wang, M. Gong, and X. Ma, "Detecting community in attributed networks by dynamically exploring node attributes and topological structure," *Knowl.-Based Syst.*, vol. 196, May 2020, Art. no. 105760.

[40] O. Levy and Y. Goldberg, "Neural word embedding as implicit matrix factorization," in *Proc. 27th Int. Conf. Neural Inf. Process. Syst.*, vol. 27, 2014, pp. 2177–2185.

[41] A. Malathi and D. Radha, "Analysis and visualization of social media networks," in *Proc. Int. Conf. Comput. Syst. Inf. Technol. Sustain. Solutions (CSITSS)*, Oct. 2016, pp. 58–63.

[42] H. Cheng, Y. Zhou, and J. X. Yu, "Clustering large attributed graphs: A balance between structural and attribute similarities," *ACM Trans. Knowl. Discovery Data*, vol. 5, no. 2, pp. 1–33, Feb. 2011.

[43] P. Sen, G. Namata, M. Bilgic, L. Getoor, B. Galligher, and T. Eliassi-Rad, "Collective classification in network data," *AI Mag.*, vol. 29, no. 3, p. 93, 2008.

[44] X. Huang, J. Li, and X. Hu, "Label informed attributed network embedding," in *Proc. 10th ACM Int. Conf. Web Search Data Mining*, Feb. 2017, pp. 731–739.
[45] A. Lancichinetti, S. Fortunato, and F. Radicchi, "Benchmark graphs for testing community detection algorithms," *Phys. Rev. E, Stat. Phys. Plasmas Fluids Relat. Interdiscip. Top.*, vol. 78, no. 4, 2008, Art. no. 046110.
[46] K. Berahmand, S. Haghani, M. Rostami, and Y. Li, "A new attributed graph clustering by using label propagation in complex networks," *J. King Saud Univ.-Comput. Inf. Sci.*, vol. 34, no. 5, pp. 1869–1883, May 2022.
[47] J. Tang, M. Qu, M. Wang, M. Zhang, J. Yan, and Q. Mei, "LINE: Large-scale information network embedding," in *Proc. 24th Int. Conf. World Wide Web*, May 2015, pp. 1067–1077.
[48] A. Grover and J. Leskovec, "node2vec: Scalable feature learning for networks," in *Proc. 22nd Int. Conf. Knowl. Discovery Data Mining*, Aug. 2016, pp. 855–864.
[49] S. Pan, R. Hu, S.-F. Fung, G. Long, J. Jiang, and C. Zhang, "Learning graph embedding with adversarial training methods," *IEEE Trans. Cybern.*, vol. 50, no. 6, pp. 2475–2487, Jun. 2020.
[50] P. Velickovic, W. Fedus, W. L. Hamilton, P. Liò, Y. Bengio, and R. D. Hjelm, "Deep graph infomax," in *Proc. ICLR*, 2019, vol. 2, no. 3, p. 4.
[51] Y. Zhu, Y. Xu, F. Yu, Q. Liu, S. Wu, and L. Wang, "Deep graph contrastive representation learning," 2020, *arXiv:2006.04131*.
[52] T. Chakraborty, A. Dalmia, A. Mukherjee, and N. Ganguly, "Metrics for community analysis: A survey," *ACM Compt. Surv.*, vol. 50, no. 4, pp. 1–37, Aug. 2017.

**Kun Guo** received the M.E. and Ph.D. degrees in computer science and technology and management from Fuzhou University, Fuzhou, China, in 2005 and 2012, respectively.

He was a Visiting Scholar with the Hong Kong University of Science and Technology, Hong Kong, China, from 2019 to 2020. He is currently an Associate Professor with the College of Mathematics and Computer Science, Fuzhou University. His research interests include complex network data mining, distributed computing, federated learning, and gray system theory.

Dr. Guo is a member of the China Computer Federation (CCF) and the Fujian Provincial Key Laboratory of Network Computing and Intelligent Information Processing.

**Zizheng Zhao** received the B.S. degree in computer science from Fuzhou University, Fuzhou, China, in 2020, where he is currently pursuing the M.S. degree with the College of Computer and Data Science.

His research interest includes community detection.

**Zhiyong Yu** (Member, IEEE) received the M.E. and Ph.D. degrees in computer science and technology from Northwestern Polytechnical University, Xi'an, China, in 2007 and 2011, respectively.

He was a Visiting Student with Kyoto University, Kyoto, Japan, from 2007 to 2009, and a Visiting Researcher with the Institut Mines-Telecom, TELECOM SudParis, Évry, France, from 2012 to 2013. He is currently a Full Professor with the College of Computer and Data Science, Fuzhou University, Fuzhou, China. His current research interests include pervasive computing, mobile social networks, mobile crowdsensing, and human–machine intelligence.

**Wenzhong Guo** received the B.S. and M.S. degrees in computer science and the Ph.D. degree in communication and information system from Fuzhou University, Fuzhou, China, in 2000, 2003, and 2010, respectively.

He is currently a Full Professor with the College of Mathematics and Computer Science, Fuzhou University. He currently leads the Fujian Provincial Key Laboratory of Network Computing and Intelligent Information Processing. His research interests include intelligent information processing, sensor networks, network computing, and network performance evaluation.

Dr. Guo is also a member of the Association for Computing Machinery (ACM) and a Senior Member of the China Computer Federation (CCF).

**Ronghua Lin** received the B.E. degree in computer science and technology from the School of Computer Science, South China Normal University (SCNU), Guangzhou, China, in 2017, where he is currently pursuing the Ph.D. degree in software engineering.

His research interests include recommendation systems, system optimization based on machine learning, and social networks.

**Yong Tang** received the B.S. and M.Sc. degrees from Wuhan University, Wuhan, China, in 1985 and 1990, respectively, and the Ph.D. degree from the University of Science and Technology of China, Hefei, China, in 2001, all in computer science.

He was the Dean of the School of Computer Science, South China Normal University (SCNU), Guangzhou, China, from 2009 to 2021, where he is currently a Professor. Before joining SCNU in 2009, he was the Vice Dean of the School of Information of Science and Technology, Sun Yat-sen University, Guangzhou. He has published more than 200 papers in various journals, conferences, and books. His current interests include data and knowledge engineering, social networking, and collaborative computing.

Dr. Tang is also the Director of the Technical Committee on Collaborative Computing (TCCC) of the China Computer Federation (CCF).

**Ling Wu** received the Ph.D. degree from the School of Economics and Management, Fuzhou University, Fuzhou, China, in 2019.

She is currently an Associate Professor with the College of Computer and Data Science, Fuzhou University. Her recent research interests include social networks, gray systems, and machine learning.

Dr. Wu is a member of the China Computer Federation (CCF) and the Fujian Key Laboratory of Network Computing and Intelligent Information Processing.