An Overlapping Community Detection Approach Based on Deepwalk and Improved Label Propagation

Hongtao Yu, Ru Ma, Jinbo Chao, and Fuzhi Zhang

***Abstract*— Label propagation-based overlapping community detection algorithms have been widely used in complex networks due to their simplicity and efficiency. However, such algorithms need to randomly choose neighbor nodes and do not fully take the network's topology into consideration, resulting in low stability and accuracy. Aiming at this problem, we propose an overlapping community detection approach based on DeepWalk and the improved label propagation. We first use the DeepWalk model to learn the network's topology to obtain low-dimensional vector representations that reflect the spatial location of nodes and construct the weight matrix through vector dot product operation. Then, we design a label propagation algorithm with a preference selection strategy, which can obtain stable overlapping communities by exchanging information with fixed neighbors on the basis of preserving the nodes' own labels. The experimental results on the real network and synthetic datasets show that the proposed approach has better accuracy and stability than the baseline methods.**

***Index Terms*— DeepWalk, label propagation, overlapping community detection, preference selection strategy.**

NOMENCLATURE

G	Complex network.
V	Set of nodes in the complex network dataset.
E	Set of edges in the complex network dataset.
u	Node.
c	Community identifier.
$\boldsymbol{W}$	Weight matrix.
BC	Belonging coefficient.
$N(u)$	Set of neighbors of node u.
$\boldsymbol{\phi(u)}, \boldsymbol{\phi(v)}$	Low-dimensional vector representations of nodes u and v.
LP	Set of labels for all nodes before propagation.
LC	Set of labels for all nodes after propagation.
LP_u	Set of labels for node u before propagation.
LC_u	Set of labels for node u after propagation.
$\|.\|$	Number of elements in a set.

Manuscript received September 9, 2021; revised January 15, 2022 and February 12, 2022; accepted February 15, 2022. This work was supported in part by the National Natural Science Foundation of China under Grant 62072393. *(Corresponding author: Fuzhi Zhang.)*

The authors are with the School of Information Science and Engineering and the Key Laboratory for Computer Virtual Technology and System Integration of Hebei Province, Yanshan University, Qinhuangdao 066004, China (e-mail: yu5771@163.com; 3023349202@qq.com; xjcjb@ysu.edu.cn; xjzfz@ysu.edu.cn).

This article has supplementary downloadable material available at https://doi.org/10.1109/TCSS.2022.3152579, provided by the authors.

Digital Object Identifier 10.1109/TCSS.2022.3152579

I. INTRODUCTION

RESEARCH on complex networks has indicated that complex networks not only have the characteristics of small world and scale free but also have overlapping community structure [1]–[4]. Overlapping community refers to individuals in the network who may belong to multiple communities, resulting in overlapping among communities. For example, in the field of recommender systems, some active users may have multiple areas of interest, resulting in overlapping areas of interest; in the field of fake review detection, an individual spammer might participate in multiple spamming groups, resulting in the overlapping among spamming groups. Therefore, the detection of overlapping community structures is helpful for deeper analysis of complex networks.

In recent years, great progress has been made in the research of overlapping community detection. Research work focuses mainly on: 1) detecting overlapping communities using the properties of nodes [5], [6] such as seed node expansion; 2) using the properties of edges [7]–[9] to segment the links between nodes in the network to obtain the overlapping communities; and 3) obtaining the overlapping communities by dividing the subgraphs in the network from the perspective of subgraphs [10]–[12]. However, due to the prevalence of large-scale complex networks in real life, there are new demands for the processing ability and efficiency of overlapping community detection algorithms. Methods based on network representation learning [13]–[16] and label propagation [17], [18] can handle community detection problems of large-scale network data. Label propagation-based dynamic overlapping community detection algorithms have attracted more attention because of the label propagation's nearly linear time complexity. However, such algorithms have the disadvantages of strong randomness, poor robustness, and susceptible to assigning all nodes to a single community. Network representation learning-based methods first generate low-dimensional vector representations of network nodes and then use the classical clustering algorithms to detect the community structure in the network. Unfortunately, such methods only work for the detection of nonoverlapping communities.

To overcome the above limitations, we propose an overlapping community detection approach based on the DeepWalk model and the improved label propagation, which is called DPLPA. First, we use the DeepWalk model to generate low-dimensional representations of nodes, based on

which, the weight matrix between nodes in the network is calculated and taken as the basis of label propagation. Second, we devise a preference selection strategy by considering the static structural information and dynamic label information of the complex network and use the improved label propagation algorithm with this strategy to obtain a stable division of overlapping communities. Finally, we use the normalized mutual information (NMI) and extended modularity to detect the overlapping communities.

Our main contributions include the following.

1) We propose a method to calculate the weights between network nodes. This method employs the DeepWalk model to learn the network's topology to obtain the low-dimensional vector representations that reflect the spatial position of nodes and takes the vector dot products as the weights between nodes in the network, which can preserve the network topology information to a great extent. Compared with existing weight calculation methods (e.g., Jaccard and the shortest path), our method can obtain better similarities between nodes by considering the higher order proximity in the network.
2) We propose an improved label propagation strategy. The proposed strategy considers the static topology information and dynamic label information of the complex network and chooses the neighbor nodes with the maximum preference probability for exchanging information, which can overcome the drawback of the traditional label propagation algorithms that choose nodes randomly and improve the stability of overlapping community detection.
3) We improve the traditional label updating method. The improved method not only learns the label information of the node's neighbor nodes but also reserves the node's own label information. Compared with the traditional label updating methods, this method can save more information about nodes and help to distinguish key nodes.
4) To show the effectiveness of DPLPA, we conduct extensive experiments on the synthetic and real network datasets and compare DPLPA with six baseline methods.

The remainder of this article is organized as follows. The related work is introduced in Section II. Section III provides preliminaries. Section IV describes the proposed overlapping community detection approach. The experimental results are reported and analyzed in Section V. Finally, Section VI concludes this article.

II. Related Work

The earliest research on community structure can be traced back to Weiss and Jacobson in 1955 on the relationship between government agency staffs [19]. They removed the relationships between different working groups so that they could obtain separate working groups. This idea became the basis of many subsequent community detection algorithms. Since then, the community detection algorithms [20]–[22] have been continuously improved and developed, mainly from the perspectives of graph segmentation, clustering, modularity, label propagation, and so on. However, most of the nodes in complex networks in the real world may belong to more than one community. Therefore, the detection of overlapping communities has become a key issue.

Researchers have proposed various methods to detect overlapping communities from different perspectives. These methods focus mainly on local optimization approaches, link approaches, k-clique approaches, and dynamic propagation approaches.

Local optimization-based overlapping community detection methods first obtain a set of nonoverlapping nodes as seeds and then use seed node expansion to generate overlapping communities. Lancichinetti *et al.* [5] proposed a local fitness method (LFM), and they randomly selected a node that was not assigned to any community as the initial seed and then adopted a fitness function to expand communities. This method is suitable for detecting overlapping communities in large-scale complex networks. Lee *et al.* [6] proposed a greedy clique expansion method, which mined a maximum clique from the complex network as the seed and then adopted a greedy strategy to build communities. This method solves the poor stability of the LFM algorithm. However, the local optimization-based overlapping community detection methods will eventually have some nodes that do not belong to any community.

Link-based overlapping community detection approaches first partition the edge sets of the complex network into disjoint link communities, which will be then transformed into the final node communities. The nodes belonging to different linked communities' edges are overlapping nodes. Ma *et al.* [7] proposed a loop edges delete method to detect overlapping communities. They calculated the structural similarity between two nodes as the weight of the edge connecting the two nodes, deleted the unweighted edges, and reconnected the isolated nodes to find overlapping communities. Gui *et al.* [8] applied a spectral analysis to line graphs to unify the overlapping and hierarchical structure of communities and employed the angular distance to compute the similarity between vertices. Hao [9] proposed a least replica method to detect overlapping communities. They defined the strength of edges and extracted edges whose strength was greater than the given threshold as skeleton edges. All potential overlapping nodes were discovered from skeleton edges and replicated them optimally. They applied the congregate strategy to generate initial communities and continuously reassemble the replicas of the same original node belonging to the same communities to find overlapping communities. Link-based overlapping community detection approaches perform well in the complex networks with a large number of nodes, but they ignore the weight and direction in the complex networks.

The k-clique-based overlapping community detection methods mine and traverse all k-clique subgraphs. If a k-clique overlaps with another k-clique by $k-1$ nodes, the two k-cliques are connected. The sets of all connected k-clique subgraphs are overlapping communities. Palla *et al.* [10] proposed a clique percolation method (CPM). They mined all k-clique subgraphs and treated each k-clique subgraph as a node to build an overlapping matrix in which the elements

whose diagonals were less than k and nondiagonals were less than $k-1$ were set to 0 to generate the community adjacency matrix and detect overlapping communities. Farkas *et al.* [11] applied the CPM algorithm to a weighted network and proposed a CPMw method, which could obtain overlapping communities in both directed and undirected graphs. Kumpula *et al.* [12] presented a sequential CPM, which could quickly find overlapping communities in both weighted and unweighted complex networks. They sequentially inserted the constituent links to the network and simultaneously kept track of the emerging community structure. The k-clique-based overlapping community detection methods perform well when the setting of k value is reasonable. However, if k is too big or too small, most nodes in complex network are deleted, resulting in poor performance.

For dynamic propagation-based overlapping community detection approaches, a unique label is assigned to each node, the label propagation strategy is designed to update labels, and overlapping communities are obtained when the iteration termination condition is reached. Gregory [17] proposed a community overlapping propagation algorithm (COPRA) in which a unique label was assigned to each node and the nodes updated their own labels according to the neighbor's labels that appear the most frequent in the label sets. Xie and Szymanski [18] proposed a speaker–listener label propagation algorithm (SLPA). They defined a new dynamic iterative rule to determine the communities of nodes. Dynamic propagation-based overlapping community detection methods need to randomly select neighbor's labels in the phase of label propagation, which leads to inconsistent results of overlapping communities.

III. Preliminaries

A. Problem Definition

In this article, we focus on the overlapping community detection problem on the complex networks. Given a graph $G = (V, E)$, where V denotes the set of nodes, E denotes the set of edges, and $A \in R^{|V|\times|V|}$ denotes the adjacency matrix of G, our goal is to divide all nodes in G into M communities. Formally, the problem of overlapping community detection is defined as follows:

$$(G, A) \xrightarrow{F} C$$

where $C = \{C_1, C_2, \ldots, C_M\}$ denotes a set of communities and satisfies $C_1 \cup C_2 \cup \cdots \cup C_M = V$ and $C_i \cap C_j \neq \varnothing(\exists(i, j \in \{1, 2, \ldots, M\}))$ and F denotes the method of overlapping community detection.

B. DeepWalk

DeepWalk is an online learning technique, which utilizes node sequences obtained from truncated random walks to learn low-dimensional vector representations of nodes in the graph [13]. It mainly includes the following two parts.

1) *Random Walks:* Each node in graph $G = (V, E)$ is uniformly chosen as the starting node of the random walk, a walk samples uniformly from the neighbors of the currently visited node until the maximum walk length is reached. This process is repeated until the number of times that each node is selected as the starting node of the random walk reaches the given threshold.
2) *SkipGram:* Treat the node sequences obtained from truncated random walks as sentences to train the SkipGram model to obtain the low-dimensional vector representations of nodes in graph G.

C. Label Propagation Algorithm

The label propagation algorithm [23] is a graph-based semisupervised learning method. It propagates the labels of known nodes to obtain the labels of unknown nodes in the graph. It includes the following three key steps.

Step 1 (Label Initialization): Each node is assigned to a unique label, and each node can only hold one label.

Step 2 (Label Updating): Each node updates its label(s) according to the label updating rule.

Step 3 (Label Iteration): Repeat Step 2 until the termination condition is satisfied.

IV. Proposed DPLPA Model

The overlapping community detection model DPLPA proposed in this article is shown in Fig. 1, which includes four stages. In the first stage, a weight matrix is constructed based on low-dimensional vector representations of nodes obtained from the DeepWalk model. In the second stage, a unique label, node ID, is assigned to each node and the node's local clustering coefficient (LCC) is calculated. The local structural similarity (LSS) and the tightness of connection between nodes are calculated. In the third stage, an improved label propagation algorithm with a preference selection strategy is proposed to update the labels. In the fourth stage, the overlapping communities are detected when the label propagation process is stopped. The details of DPLPA model are introduced in the following. To facilitate discussion, the descriptions of notations used in this article are listed in the Nomenclature.

A. Generation of Weight Matrix

In this section, we first obtain the low-dimensional vector representations of nodes using the DeepWalk model and then calculate the weight matrix through the vector dot product operation.

Many network embedding techniques, such as DeepWalk [13], Node2Vec [14], LINE [15], and SDNE [16], can be used to obtain low-dimensional vector representations of nodes in the network. DeepWalk can not only capture the higher order proximity in the network but also have scalability, which makes it suitable for many real-world applications. While Node2Vec can capture the higher order proximity in the network and is scalable, it needs to set the biased parameters p and q to control the walks. This is a difficult task in real-world applications. LINE only captures the first- and second-order proximities and does not consider the higher order proximity in the network. SDNE can capture both the first- and second-order proximities in the network, but it is unable to capture the higher order proximity in the network and has many hyperparameters to be set. Therefore, we use DeepWalk to obtain the low-dimensional vector representations.

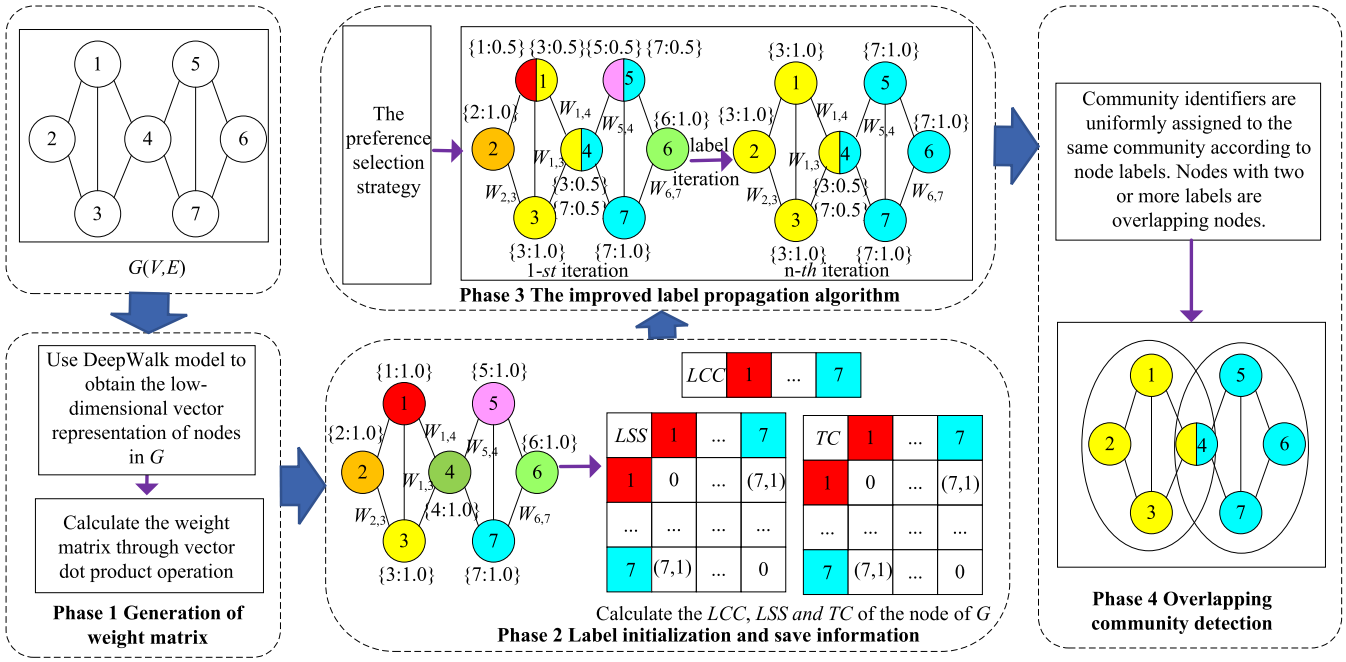


Fig. 1. Overview of DPLPA. In phase 1, the low-dimensional vector representations of nodes are obtained with the DeepWalk model and the weight matrix is generated through the vector dot operation. In phase 2, each node is assigned a unique label and the static information is saved. In phase 3, the improved label propagation algorithm is applied to exchange labels by selecting neighbors with the maximum preference probability. In phase 4, overlapping communities are detected according to the labels.

Definition 1 (Weight Between Nodes): For any two nodes $u, v \in V$ and $(u, v) \in E$, the weight between nodes u and v in G refers to the dot product of low-dimensional vector representations of nodes u and v, i.e.,

$$\boldsymbol{W}(u, v) = \boldsymbol{\phi}(u) \cdot \boldsymbol{\phi}(v). \tag{1}$$

We can apply (1) to compute $\boldsymbol{W}$ according to the low-dimensional vector representations of nodes obtained with the DeepWalk model. Algorithm 1 describes the process of weight matrix generation.

In Algorithm 1, Line 1 generates the low-dimensional vector representations of nodes using the DeepWalk model. Lines 2–9 obtain the weight matrix by calculating the vector dot products.

Algorithm 1 Generation of Weight Matrix

Input : complex network $G(V, E)$
window size ws
embedding size d
walks per vertex p
walk length wl
adjacency matrix $\boldsymbol{A}$
Output: the matrix of weight $\boldsymbol{W} \in \mathbb{R}^{|V|\times|V|}$

1 $\boldsymbol{\phi} \leftarrow PerformDeepWalk(G, ws, d, p, wl)$
2 $\boldsymbol{W} \leftarrow \boldsymbol{0}_{|V|\times|V|}$
3 **for** *each node* u *in* V **do**
4 **for** *each node* v *in* V **do**
5 **if** $\boldsymbol{A}[u][v] = 1$ **then**
6 $\boldsymbol{W}(u, v) \leftarrow \boldsymbol{\phi}(u) \cdot \boldsymbol{\phi}(v)$
7 **end**
8 **end**
9 **end**
10 **return** $\boldsymbol{W}$

B. Label Initialization

In this section, we first give several related definitions and then devise an algorithm for label initialization.

Definition 2 (BC [17]): For any node $u \in V$ in G, the BC of the label reflects the strength that node u belongs to the community c, i.e.,

$$\mathrm{BC}_j(c, u) = \frac{\sum_{v\in N(u)} \mathrm{BC}_{j-1}(c, v)}{|N(u)|} \tag{2}$$

where j denotes the jth iteration.

Definition 3 (LCC [24]): For any node $u \in V$ in G, the LCC reflects the ratio of the number of edges formed by the neighbors of u to the number of possible edges of nodes in $N(u)$ and is calculated as follows:

$$\mathrm{LCC}(u) = \frac{2|E_u|}{|N(u)| \times (|N(u)| - 1)} \tag{3}$$

where E_u denotes the set of edges between the neighbors of node u.

Definition 4 (LSS [25]): For any two nodes $u, v \in V$ and $(u, v) \in E$ in G, the LSS between nodes u and v is calculated as follows:

$$\mathrm{LSS}(u, v) = \frac{|\mathrm{SS}(u) \cap \mathrm{SS}(v)|}{|\mathrm{SS}(u) \cup \mathrm{SS}(v)|} \tag{4}$$

where $\mathrm{SS}(u)$ denotes the set of node u and its neighbor nodes and $\mathrm{SS}(v)$ denotes the set of node v and its neighbor nodes.

Definition 5 (Tightness of Connection, TC): For any two nodes $u, v \in V$ and $(u, v) \in E$ in G, the tightness of connection between nodes u and v refers to the ratio of the number of common four-clique subgraphs between nodes u

Algorithm 2 Label Initialization

Input : complex network G
weight matrix $\boldsymbol{W}$
Output: LP, LCC, LSS, TC

```
LP ← ∅
for each node u in V do
    BC(u, u) ← 1
    LP_u ← (u; BC(u, u)) //LP_u is the label set of node u
    LP ← LP + LP_u
end
for each node u in V do
    LCC(u) ← calculate the local clustering coefficient of node u according to Eq. (3)
    for each node v in N(u) do
        LSS(u, v) ← calculate the local structural similarity between node u and node v according to Eq. (4)
        TC(u, v) ← calculate the tightness of connection between node u and node v according to Eq. (5)
    end
end
return LP, LCC, LSS, TC
```

and v to the number of four-clique subgraphs of node v, i.e.,

$$\mathrm{TC}(u, v) = \frac{|S(u) \cap S(v)|}{|S(v)|} \tag{5}$$

where $S(u)$ and $S(v)$ denote the four-clique subgraphs of nodes u and v, respectively.

The specific process of label initialization is given as follows:

1) A unique label is assigned to each node. A label is mainly composed of two parts: the community identifier and the label BC. The community identifier is determined by the node ID, and the BC is initialized to 1.
2) In order to reduce the computational cost, we calculate and save the network static structure information, which includes the LCC, the LSS, and the tightness of connection.

According to the above steps, the algorithm of label initialization is described in Algorithm 2.

In Algorithm 2, Lines 1–6 initialize the labels of all nodes. Lines 7–13 calculate and save the network static structure information.

C. Improved Label Propagation Algorithm

In this section, we design a preference selection strategy and propose an improved label update rule.

The traditional label propagation algorithms have two shortcomings in the stage of label propagation. On the one hand, the neighbor nodes are randomly selected when the propagation rule fails. On the other hand, the nodes' own label information is not considered. To address these limitations, we propose an improved label propagation algorithm.

First, we design a preference selection strategy to ensure that each node obtains labels from the neighbors with the maximum preference probability.

Definition 6 (Label Similarity, LS): For any two nodes $u, v \in V$ and $(u, v) \in E$ in G, the label similarity refers to the sum of the BC products when the community identifiers in the labels of nodes u and v are the same, i.e.,

$$\mathrm{LS}(u, v) = \sum_{(c_u \in \mathrm{LP}_u, c_v \in \mathrm{LP}_v) \wedge c_u = c_v} \mathrm{BC}(c_u, u) \times \mathrm{BC}(c_v, v). \tag{6}$$

By considering the static structure information and dynamic label information, the preference selection value of node $u \in V$ to node $v \in V$ is calculated as follows:

$$\mathrm{prefer}(u, v) = \exp((\mathrm{LCC}(v) + \mathrm{TC}(u, v)) + \mathrm{LS}(u, v) \times (\boldsymbol{W}(u, v) + \mathrm{LSS}(u, v))). \tag{7}$$

Second, we improve the label update rule through which all nodes not only accept their neighbor's labels with the highest probability of preference selection decision but also retain their own labels.

The probability that node $u \in V$ chooses its neighbor $v \in V$ is calculated as follows [26]:

$$P(u, v) = \frac{\mathrm{prefer}(u, v)}{\sum_{\hat{v} \in N(u)} \mathrm{prefer}(u, \hat{v})}. \tag{8}$$

The neighbor node $v \in N(u)$ is selected to exchange labels with node $u \in V$, i.e.,

$$l_u = \underset{v \in N(u)}{\mathrm{argmax}}\ P(u, v). \tag{9}$$

Based on the above discussions, the improved label propagation algorithm is described In Algorithm 3.

Algorithm 3 The Improved Label Propagation Algorithm

Input : complex network G
x: the number of communities to which each node belongs
LP, LCC, LSS, TC
Output: LC_u

```
for each node v in N(u) do
    P(u, v) ←calculate preference probability according to Eqs. (6)-(8)
end
l_u ← The neighbor node v with the highest probability of preference selection is obtained as the learning target node of node u according to Eq. (9)
if LC_{l_u} ≠ ∅ then
    LC_u ← LC_{l_u} + LP_u
else
    LC_u ← LP_{l_u} + LP_u
end
normalize(LC_u)
if BC(c, u) ≤ 1/x then
    remove the label and the belonging coefficient of node u that is less than the given threshold from LC_u
    normalize(LC_u)
end
return LC_u
```

Algorithm 3 updates the labels of node u with the labels of its neighbor nodes. Lines 1–10 take the preference selection strategy into consideration when a node receives labels from its neighbor nodes. Lines 11–14 filter out the labels of u when the label BC is less than the given threshold.

D. Overlapping Community Detection

The label iteration termination condition is the key to overlapping community detection based on the label propagation algorithm. In the proposed approach, each node updates the labels iteratively according to the label propagation algorithm with a preference selection strategy, and the iteration is stopped when the type and number of labels are the same before and after updating. Nodes with the same labels are divided into the same community. If a node has $n(n \geq 1)$ community identifiers in its label set, the node will be divided into more than one community. This means that communities are overlapped.

The sets of community identifiers in the jth iteration are given as follows:

$$i_j = \{c \in V : v \in V \wedge (\mathrm{BC}_j(c, v) > 0)\}. \tag{10}$$

The set of the numbers of nodes labeled with the community identifiers in the jth iteration is calculated as follows:

$$n_j = \left\{(c, i) : c \in V \wedge i = \sum_{v \in V, \mathrm{BC}_j(c,v)>0} 1\right\}. \tag{11}$$

The set of the minimum numbers of nodes labeled with the community identifiers in the jth iteration is given as follows:

$$m_j = \begin{cases} \{(c, i) : \exists x \exists y((c, x) \in n_{j-1} \wedge (c, y) \in n_{j-1} \\ \qquad \wedge i = \min(x, y))\}, \quad i_j = i_{j-1} \\ n_j, \quad \text{otherwise.} \end{cases} \tag{12}$$

The algorithm of overlapping community detection is given in Algorithm 4.

In Algorithm 4, Lines 1–17 judge whether the node labels reach the iteration termination condition. id() is used to get all labels from a set of labels according to (10), count() is used to get the number of labels according to (11), and mc() is used to get the smaller label number for the same label according to (12). Lines 18–28 detect overlapping communities according to the distribution of labels. $(l,\ n + u)$ represents that node u is added into community l if l in community set C.

E. Time Comlpexity Analysis

The DPLPA algorithm consists of four parts. The time complexity for generating weight matrix (Algorithm 1) is $O(|V|^2)$. The time complexity for initializing labels (Algorithm 2) is $O(|V| + |E|)$. In Algorithm 3, the time complexity for calculating preference probabilities (Lines 1–3) is $O(|E|/|V|)$ and selecting node to update labels with the highest probability (Line 4) is also $O(|E|/|V|)$. The time complexity for normalizing and deleting labels (Lines 5–14) is $O(x)$. Therefore, the time complexity of Algorithm 3 is $O(2 \times (|E|/|V|) + x)$. In Algorithm 4, the time complexity for updating labels (Lines 2–17) is $O(t_{\max} \times (2|E| + |V| \times x))$, where $t_{\max}$ represents the maximum number of label iterations. The time complexity for detecting overlapping communities (Lines 19–28) is $O(|V| \times L)$, where L represents the number of communities. Therefore, the time complexity of Algorithm 4 is $O(t_{\max} \times (2|E| + |V| \times x) + |V| \times L)$

Table I shows the comparison of time complexity for DPLPA and baselines.

Algorithm 4 Overlapping Community Detection

Input : complex network G
Output: Community set C

1 $be\_miic \leftarrow \varnothing, miic \leftarrow \varnothing$ // be_miic is the set of labels and the number of labels for all nodes before propagation. $miic$ is the set of labels and the number of labels for all nodes after propagation
2 **repeat**
3 $LC \leftarrow \varnothing$
4 **for** *each node u in V* **do**
5 $LC_u \leftarrow$ call Algorithm 3
6 $LC \leftarrow LC + LC_u$
7 **end**
8 **if** $id(LP) = id(LC)$ **then**
9 $miic \leftarrow mc(count(LP), count(LC))$
10 **else**
11 $miic \leftarrow count(LC)$
12 **end**
13 **if** $miic \neq be\_miic$ **then**
14 $LP \leftarrow LC$
15 $be\_miic \leftarrow miic$
16 **end**
17 **until** $miic = be\_miic$;
18 $C \leftarrow \varnothing$
19 **for** *each node u in V* **do**
20 $iid \leftarrow id(LP_u)$
21 **for** *each label l in iid* **do**
22 **if** $(l, n) \in C$ **then**
23 $C \leftarrow C - \{(l, n)\} + \{(l, n + u)\}$
24 **else**
25 $C \leftarrow C + \{(l, u)\}$
26 **end**
27 **end**
28 **end**
29 **return** C

TABLE I
COMPARISON OF ALGORITHM TIME COMPLEXITY

Algorithm	Time Complexity
COPRA	$O(x\|E\|log(x\|E\|/\|V\|))$
CPM	$O(3.14^{\|V\|/3})$
Demon	$O(\|V\| \times k_{max}^{3-\varepsilon})$
Neo-K-means	$O(t_{max} \times (n + \alpha n))$
SLPA	$O(T(\|E\|))$
UMSTMO	$O(\|E\|logk_{max})$
DPLPA	$O(t_{max} \times (2\|E\| + \|V\| \times x) + \|V\| \times L)$

V. EXPERIMENTAL EVALUATION

To evaluate the proposed DPLPA, six real-world datasets and four synthetic datasets [27] are used to conduct experiments. The hardware environment for all the experiments is as follows: Intel Core i7-10700F CPU, 2.90 GHz, and 128 GB of memory. The DPLPA algorithm is implemented in Pycharm 2020.

A. Experimental Datasets

The six real-world datasets are the Zachary karate club dataset [28], dolphin dataset [29], football dataset [30], Jazz

TABLE II
STATISTICS OF SIX REAL-WORLD DATASETS

Dataset	Nodes	Edges	AD	CC	DAC
karate	34	78	4.588	0.571	-0.4756
dolphin	62	159	5.129	0.259	-0.0436
footbal	115	613	10.661	0.403	0.1624
Jazz	198	2742	27.697	0.617	0.0202
Youtube	5000	4689	1.876	0.208	0.0008
Amazon	16716	48739	5.831	0.649	-0.0294

TABLE III
SOME COMMON PARAMETERS IN THE LFR MODEL

Parameter	Description
n	The number of nodes
k	Average degree(The average value of most large-scale real social networks is around 10)
$maxk$	Maximum degree
mu	The coefficient of the internal connection strength of the community,the larger the value, the weaker the connection.
$t1$	Power distribution exponent for node degree
$t2$	Power distribution exponent for community size
$minc$	Minimum community size
$maxc$	Maximum community size
on	Number of overlapping nodes
om	Number of memberships of the overlapping nodes

dataset [31], and Youtube and Amazon datasets [32]. Table II shows the statistics of nodes, edges, average degree (AD), cluster coefficient (CC), and degree assortativity coefficient (DAC) in these datasets.

The collection of complex networks is very difficult in the real world, which makes it difficult to verify the performance of the algorithms. Therefore, it has become a feasible way to use artificially generated networks to evaluate the effectiveness of algorithms. The Lancichinetti–Fortunato–Radicchi (LFR) network [33] is considered as the standard network for community detection. The parameters of LFR are shown in Table III. The synthetic datasets are shown in Table IV.

B. Evaluation Metrics

The extension of modularity (EQ) [34] and NMI [5] metrics are used to evaluate the performance of the DPLPA, which are defined as follows:

$$\mathrm{EQ} = \frac{1}{2|E|}\sum_{ij}\frac{1}{o_i o_j}\left(\boldsymbol{A}_{ij} - \frac{d_i d_j}{2|E|}\right)\sigma\left(c_i, c_j\right) \quad (13)$$

$$\mathrm{NMI}(P, Q) = 1 - \frac{1}{2}\left(\frac{H(P \mid Q)}{H(P)} + \frac{H(Q \mid P)}{H(Q)}\right) \quad (14)$$

where o_i and o_j represent the numbers of communities to which nodes i and j belong, d_i and d_j represent the degrees of nodes i and j, $\boldsymbol{A}$ represents the adjacency matrix, respectively, and $\sigma(c_i, c_j)$ is used to judge whether nodes i and j are in the same community. $H(P \mid Q)$ denotes the conditional entropy between the real community partition P and the algorithm partition Q, $H(Q \mid P)$ denotes the conditional entropy between the algorithm partition Q and the real community partition P, and $H(P)$ and $H(Q)$ represent the entropies of the real community partition P and the algorithm partition Q, respectively.

TABLE IV
STATISTICS OF FOUR SYNTHETIC DATASETS

Parameter	LFR1	LFR2	LFR3	LFR4
n	500	500	500	1000∼15000
k	25	25	25	25
$maxk$	50	50	50	50
mu	0.1∼0.5	0.4	0.4	0.3
$t1$	2	2	2	2
$t2$	1	1	1	1
$minc$	10	10	10	10
$maxc$	50	50	50	50
on	50	50∼300	50	50
om	2	2	2∼6	2

C. Experimental Results and Analysis

To show the effectiveness of our approach (DPLPA), we conduct experiments on ten datasets and compare DPLPA with the following six baselines.

1) *CPM [10]:* An approach for detecting overlapping communities, which treats communities as the sets of fully connected k-clique subgraphs. In our experiments, we set k to 4.
2) *COPRA [17]:* In this approach, each node is assigned a unique label in the complex network, and each node determines its own labels according to the label distribution of their neighbors. In the experiments, parameter x varies from 2 to 11 with an interval of 1.
3) *SLPA [18]:* An approach for detecting overlapping communities based on label propagation, in which a unique label is assigned to each node, candidate labels are obtained by traversing and scanning the sets of neighbor's labels, and each node's labels are assigned from the candidate labels. In this approach, the maximum number of iterations T and the minimum probability density of community members r need to be set. In the experiment, parameter T is chosen from 20 to 50 and parameter r is chosen from 0.2 to 0.5.
4) *Demon [35]:* The method allows each node to vote for the communities around in its limited global system view, and the local communities are merged into a global set. In the experiments, the parameter ε is chosen from 0.2 to 0.6. Parameter N is chosen from 3 to 6 in the real-world datasets and parameter N is set to the minimum number of nodes needed to form the community in the synthetic datasets.
5) *G2G/Neo-K-Means [36], [37]:* An unsupervised method to learn the network's topology to obtain the low-dimensional vector representations of nodes, and the improved K-means algorithm is used to detect overlapping communities. In our experiments, the embedding dimension is set to 128, and the number of clusters l is chosen from 2 to 12 in the real-world datasets. For the synthetic datasets, we set l to the number of real community divisions.

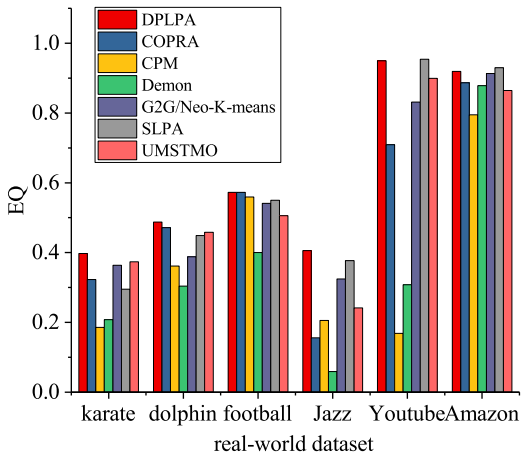


Fig. 2. Comparison of the EQ for seven methods on six real-world datasets.

6) *UMSTMO [38]:* In this approach, the union of all maximum spanning trees (UMST) is explored and each node in the UMST is linked with its most similar neighbor. Then, the local community is extracted for each node, and overlapping communities are detected according to the number of shared nodes for local communities. In the experiment, we use the default parameters of UMSTMO.

All experiments are repeated 20 times and the average values are used as the final results.

1) Comparison of the EQ for Seven Methods on Six Real-World Datasets: Fig. 2 shows the comparison of EQ for DPLPA, COPRA, CPM, Demon, G2G/Neo-K-means, SLPA, and UMSTMO on six real-world datasets.

As shown in Fig. 2, DPLPA performs better than COPRA, CPM, Demon, G2G/Neo-K-means, SLPA, and UMSTMO on real-world datasets. It can be seen that the performance of CPM is poor on the karate and Youtube datasets. This is because CPM needs to mine all four-clique graphs on the dataset. When the number of four-clique graphs is small, the EQ is quite low. The performance of Demon is not stable and is very poor on the Jazz dataset. The EQ of G2G/Neo-K-means on the real-world datasets is relatively stable. This is because the structure of real-world datasets is clear, and G2G can learn the high-quality representations of datasets, making the Neo-K-means correctly find overlapping communities. The performance of UMSTMO is also stable, indicating that the noise in real-world datasets is small, the quality of the UMST is high, and all nodes can be divided into the correct communities. The EQ of DPLPA is better than that of SLPA in most datasets and higher than that of COPRA in the real-world datasets. The main reason is that DPLPA uses the preference selection strategy to select the neighbors with the highest probability, while COPRA and SLPA need to randomly select neighbor's labels under the condition of uniform labels.

2) Comparison of the NMI for Seven Methods on Four Synthetic Datasets: Fig. 3 shows the comparison of the NMI for DPLPA, COPRA, CPM, Demon, G2G/Neo-K-means, SLPA, and UMSTMO on four synthetic datasets.

As shown in Fig. 3(a), for the LFR1, the NMI of DPLPA, COPRA, CPM, Demon, G2G/Neo-K-means, SLPA, and UMSTMO decrease as *mu* increases from 0.1 to 0.5. The main reason is that all communities are mixed together and each individual community is influenced by the noise of neighbor communities. The average NMI of COPRA and SLPA drops sharply when *mu* is greater than 0.2. This is because the structure of network becomes more complex, and the label propagation of COPRA and SLPA cannot distinguish the noisy neighbors and the normal neighbors. CPM is affected by the number of four-clique subgraphs in the network. The NMI of G2G/Neo-K-means decreases smoothly when *mu* increases from 0.1 to 0.5. The possible reason is that G2G is limited to the noise of neighbor communities. UMSTMO is affected by the quality of UMST, and some nodes cannot be divided into the correct communities with the increase in *mu*. The NMI of DPLPA is better than that of SLPA, CPM, Demon, and G2G/Neo-K-means UMSTMO and is overall better than that of COPRA on the LFR1, which means that DPLPA is effective for detecting overlapping communities even if the network structure is not obvious.

As shown in Fig. 3(b), the NMI of DPLPA is better than that of COPRA, CPM, Demon, SLPA, G2G/Neo-K-means, and UMSTMO on LFR2. This indicates that DPLPA is effective on LFR2. As for COPRA and SLPA, their NMI decreases sharply as *on* increases from 50 to 300 because the label propagation of COPRA and SLPA treats the nodes that actually belong to different communities as one community. In most cases, the NMI of CPM drops with the increase in *on*. However, the NMI of CPM increases when *on* is equal to 200. The main reason is that fewer outliers exist in the network. Demon performs poorly when *on* is less than 150. However, the NMI of Demon is superior to that of G2G/Neo-K-means, SLPA, and COPRA when *on* is greater than 150, which means that Demon is more stable. While UMSTMO shows a steady decline in NMI, its performance is superior to that of most baselines, indicating that UMSTMO is more stable.

As shown in Fig. 3(c), DPLPA outperforms COPRA, SLPA, G2G/Neo-K-means, and Demon on the LFR3 in terms of NMI metric. This confirms that DPLPA is superior to most of the baselines in the LFR3 as *om* increases from 2 to 6. This is because the label propagation of DPLPA selects fix neighbor nodes to exchange labels and avoids randomly selecting neighbor nodes. As for CPM, the NMI reaches the peak when *om* is equal to 4. This shows that CPM can detect most of the overlapping communities in the network. The reason is that there are fewer outliers exist in the network. The NMI of DPLPA is slightly lower than that of UMSTMO when *om* is equal to 3, but it is higher than that of UMSTMO when *om* is equal to 2 or greater than 3. The NMI of COPRA and SLPA decreases sharply with the increase in *om*, which means that COPRA and SLPA have poor performance as the value of *om* continues to increase. This is because the label propagation of COPRA and SLPA cannot assign nodes to right communities with high overlapping diversity. The NMI of G2G/Neo-K-means decreases as *om* increases from 2 to 6, which means that G2G/Neo-K-means cannot effectively detect the overlapping communities in this dataset.

It can be seen from Fig. 3(d) that the NMI of DPLPA on the LFR4 dataset is better than those of COPRA, Demon, G2G/Neo-K-means, SLPA, and UMSTMO as *n* increases

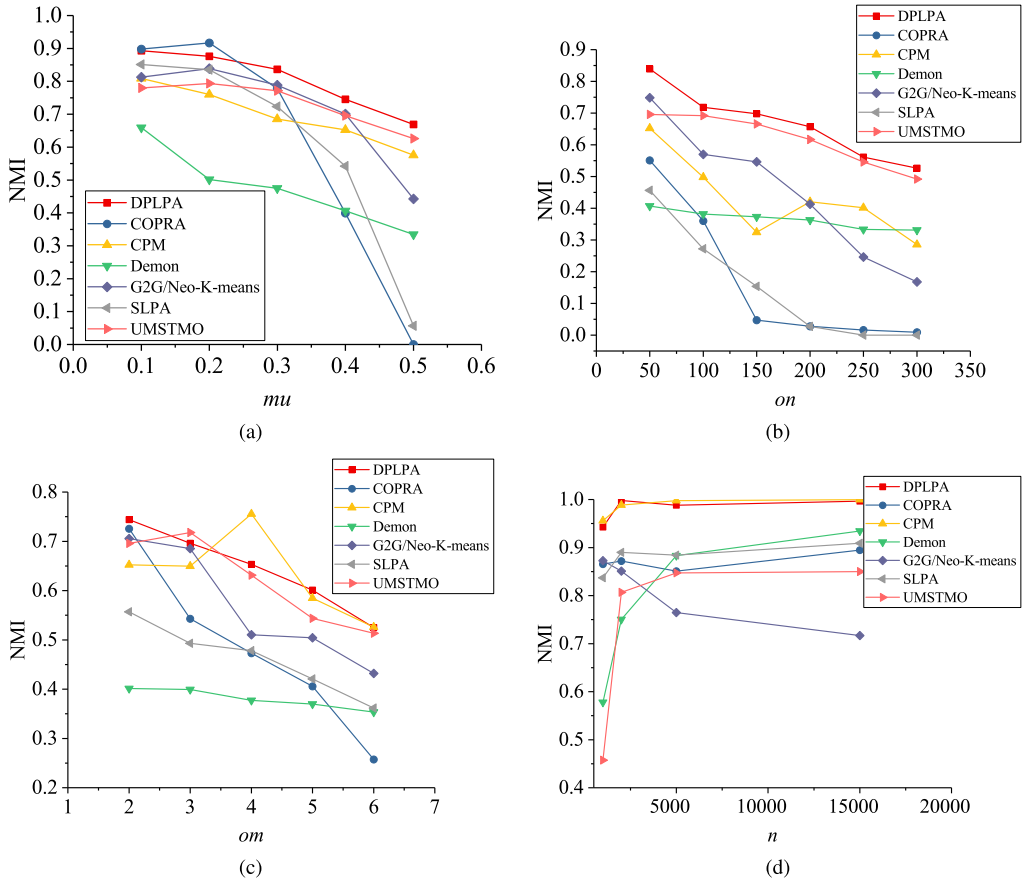


Fig. 3. Comparison of the NMI for seven methods on four synthetic datasets. (a) LFR1. (b) LFR2. (c) LFR3. (d) LFR4.

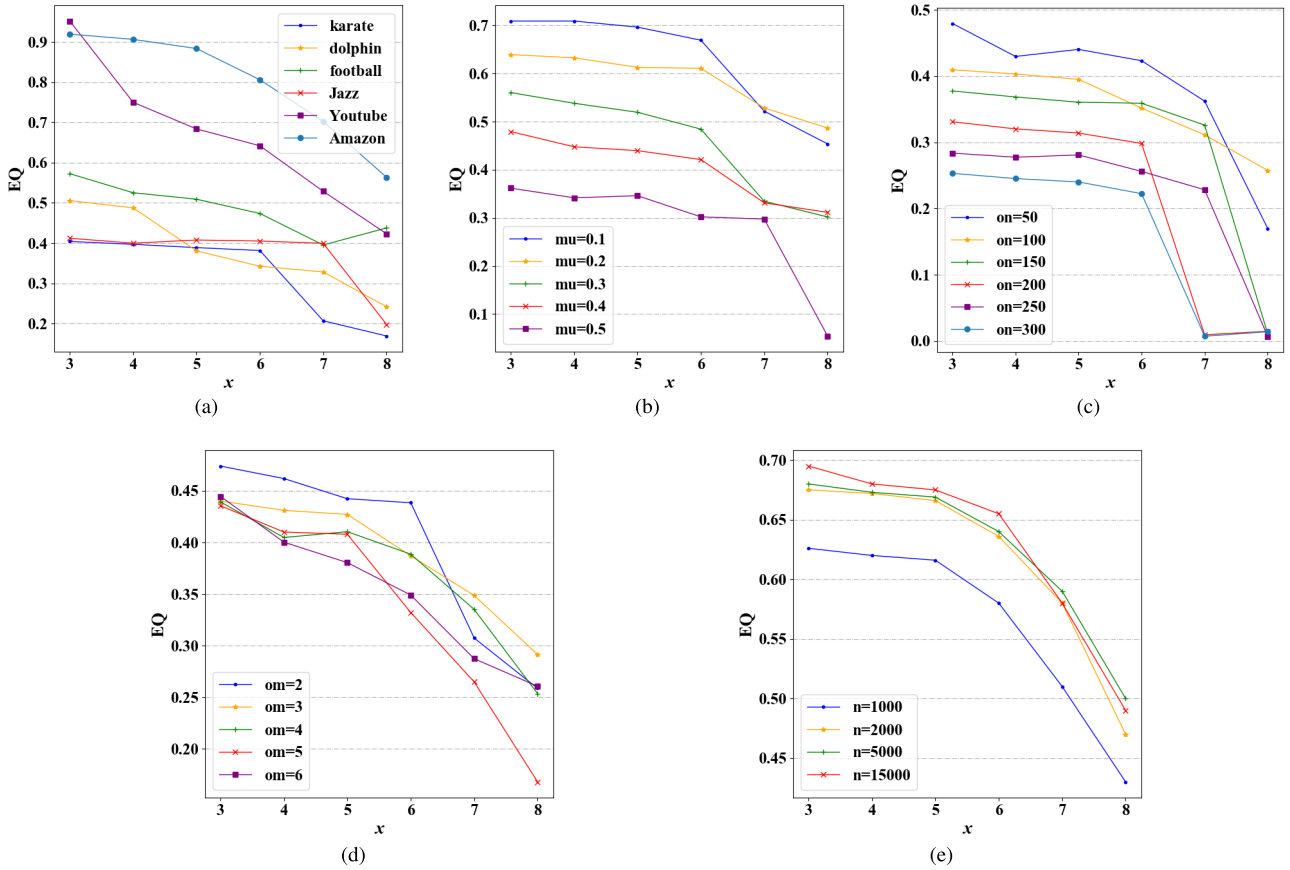


Fig. 4. Influence of parameter x on the EQ of DPLPA. (a) Real-world dataset. (b) LFR1. (c) LFR2. (d) LFR3. (e) LFR4.

from 1000 to 15 000, which means that DPLPA has good performance with the increase of network size. The NMI of DPLPA on the LFR4 is lower than that of CPM when n is equal to 1000 and 5000, but the NMI of DPLPA and CPM is basically the same. On the whole, DPLPA performs better than CPM on the LFR1, LFR2, and LFR3 datasets. The NMI of COPRA and SLPA is lower than that of DPLPA because the label propagation of COPRA and SLPA randomly selects neighbor nodes, while the label propagation of DPLPA selects neighbor nodes with the maximum preference probability to exchange labels. Demon and G2G/Neo-K-means also show poor performance because the structure of network is not obvious when mu is equal to 0.3 on the LFR4.

Based on the above analyses, it can be concluded that the performance of DPLPA is overall better than that of COPRA, CPM, Demon, G2G/Neo-K-means, SLPA, and UMSTMO on four synthetic datasets.

3) Parameter Analysis: Fig. 4 shows the influence of parameter x on the EQ of DPLPA on the real-world and synthetic datasets.

As shown in Fig. 4(a), on the real-world datasets, the EQ of DPLPA drops as x increases from 3 to 8. When $x = 3$, the EQ of DPLPA is the best. This is not difficult to explain. When $x = 3$, each node retains more important labels, which is much easier to find the correct communities. When x value is greater than 3, the set of labels of nodes has more inessential labels, resulting in nodes being divided into the incorrect communities. Therefore, parameter x is set to 3 on the real-world datasets.

As shown in Fig. 4(b), most EQ values of DPLPA decrease gradually with the increase in x. However, the EQ of DPLPA shows a trend of increasing first and then decreasing with the increase in x when $mu = 0.1$. When $x = 4$, the EQ is the best. As shown in Fig. 4(c)–(e), the EQ of DPLPA on the LFR2, LFR3, and LFR4 datasets shows a trend of decreasing as x increases from 3 to 8. When x is equal to 3, the EQ of DPLPA reaches the best. Therefore, the parameter x is set to 4 when $mu = 0.1$ on the LFR1 dataset; in other cases, x is set to 3 on the synthetic datasets.

VI. Conclusion

The detection of overlapping community plays a vital role in the analysis of complex network structure. In this article, we propose an overlapping community detection method based on DeepWalk and the improved label propagation. The findings we have achieved include the following.

1) We have proposed a method to calculate the weights between network nodes. The weights are calculated through the dot operation of vectors obtained by the DeepWalk model, which can further distinguish the influence of nodes in the network.
2) We have proposed an improved label propagation strategy. This strategy can select neighbors with the maximum preference probability to exchange labels, which can avoid the randomness in choosing neighbor nodes.
3) We have proposed an improved label update rule. This label update rule not only learns the label information of the node's neighbors but also reserves the node's own label information, which can save more information about nodes. The experimental results on the real-world datasets and synthetic datasets illustrate the effectiveness of the proposed approach in detecting overlapping communities.

In this work, we focus only on detecting overlapping communities in complex networks without attribute information. Overlapping community detection in attribute complex networks is also important to analyze the structure of networks. In our future work, we will investigate the problem of overlapping community detection in attribute complex networks.

References

[1] M. R. Shahmoradi, M. Ebrahimi, Z. Heshmati, and M. Salehi, "Multilayer overlapping community detection using multi-objective optimization," *Future Gener. Comput. Syst.*, vol. 101, pp. 221–235, Dec. 2019.
[2] X. Teng, J. Liu, and M. Li, "Overlapping community detection in directed and undirected attributed networks using a multiobjective evolutionary algorithm," *IEEE Trans. Cybern.*, vol. 51, no. 1, pp. 138–150, Jan. 2021.
[3] D. Li, M. Q. Chen, Y. Xu, F. Yuan, Y. N. Chen, and Y. Q. Fu, "Optimal community detection method based on average mutual information," *Sci. China Inf. Sci.*, vol. 49, no. 5, pp. 111–127, May 2019.
[4] J. F. Pan, Y. H. Dong, H. H. Chen, J. B. Qian, and M. Y. Dai, "The overlapping community discovery algorithm based on compact structure," *Tien Tzu Hsueh Pao.*, vol. 47, no. 1, pp. 145–152, Jan. 2019.
[5] A. Lancichinetti, S. Fortunato, and J. Kertész, "Detecting the overlapping and hierarchical community structure in complex networks," *New J. Phys.*, vol. 11, no. 3, Mar. 2009, Art. no. 033015, doi: 10.1088/1367-2630/11/3/033015.
[6] C. Lee, F. Reid, A. Mcdaid, and N. Hurley, "Detecting the overlapping and hierarchical community structure in complex networks," 2009, *arXiv:1002.1827*.
[7] T. H. Ma *et al.*, "LED: A fast overlapping communities detection algorithm based on structural clustering," *Neurocomputing*, vol. 207, pp. 2227–2249, Oct. 2021.
[8] C. Gui, R. Zhang, R. Hu, G. Huang, and J. Wei, "Overlapping communities detection based on spectral analysis of line graphs," *Phys. A, Stat. Mech. Appl.*, vol. 498, pp. 50–65, May 2018.
[9] L. Hao, "Overlapping community detection with least replicas in complex networks," *Inf Sci.*, vol. 453, pp. 216–226, Jul. 2018.
[10] G. Palla, I. Derényi, I. Farkas, and T. Vicsek, "Uncovering the overlapping community structure of complex networks in nature and society," *Nature*, vol. 435, no. 7043, pp. 814–818, Jun. 2005.
[11] I. Farkas and D. Ábel, G. Palla, and T. Vicsek, "Weighted network modules," *New J. Phys.*, vol. 9, no. 6, p. 180, Jun. 2007.
[12] J. M. Kumpula, M. Kivelä, K. Kaski, and J. Saramäki, "Sequential algorithm for fast clique percolation," *Phys. Rev. E, Stat. Phys. Plasmas Fluids Relat. Interdiscip. Top.*, vol. 78, no. 2, Aug. 2008, Art. no. 026109, doi: 10.1103/physreve.78.026109.
[13] B. Perozzi, R. Al-Rfou, and S. Skiena, "Deepwalk: Online learning of social representations," in *Proc. 20th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, New York, NY, USA, 2014, pp. 701–710.
[14] A. Grover and J. Leskovec, "Node2vec: Scalable feature learning for networks," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, New York, NY, USA, 2016, pp. 855–864.
[15] J. Tang *et al.*, "LINE: Large-scale information network embedding," in *Proc. Int. Conf. World Wide Web*, San Francisco, CA, USA, 2015, pp. 1067–1077.
[16] D. X. Wang, P. Cui, and W. W. Zhu, "Structural deep network embedding," in *Proc. ACM SIGKDD Int. Conf. Knowl. Discov. Data Mining*, San Francisco, CA, USA, 2016, pp. 1225–1234.
[17] S. Gregory, "Finding overlapping communities in networks by label propagation," *New J. Phys.*, vol. 12, Oct. 2010, Art. no. 103018, doi: 10.1088/1367-2630/12/10/103018.
[18] J. Xie and B. K. Szymanski, "Towards linear time overlapping community detection in social networks," in *Advances in Knowledge Discovery and Data Mining* (Lecture Notes in Computer Science), vol. 7302. Berlin, Germany: Springer, 2012, pp. 25–36.
[19] S. W. Robert and J. Eugene, "A method for the analysis of the structure of complex organizations," *Amer. Sociol. Rev.*, vol. 20, no. 6, pp. 661–668, Dec. 1955.

[20] 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. Computat. Soc. Syst.*, vol. 5, no. 4, pp. 1021–1033, Dec. 2018.

[21] K. Berahmand *et al.*, "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.

[22] 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.

[23] J. L. Zhang, Y. L. Ghang, and W. Shi, "Overview on label propagation algorithm and applications," *Appl. Res. Comput.*, vol. 76, no. 3, pp. 36–106, Sep. 2007.

[24] L. Katzir and S. J. Hardiman, "Estimating clustering coefficients and size of social networks via random walk," *ACM Trans. Web*, vol. 9, no. 4, p. 19, Sep. 2015.

[25] J. Sheng, K. Wang, Z. Sun, and B. Wang, "Overlapping community detection via preferential learning model," *Phys. A, Stat. Mech. Appl.*, vol. 527, Aug. 2019, Art. no. 121265, doi: 10.1016/j.physa.2019.121265.

[26] J. Q. Sun, R. G. Fan, M. Luo, Y. Q. Zhang, and L. L. Dong, "The evolution of cooperation in spatial prisoner's dilemma game with dynamic relationship-based preferential learning," *Phys. A, Stat. Mech. Appl.*, vol. 512, pp. 598–611, Dec. 2018.

[27] Z. L. Ding, X. Y. Zhang, D. D. Sun, and B. Luo, "Overlapping community detection based on network decomposition," *Sci. Rep.*, vol. 6, Apr. 2016, Art. no. 24115.

[28] W. W. Zachary, "An information flow model for conflict and fission in small groups," *J. Anthropol. Res.*, vol. 33, no. 4, pp. 452–473, Dec. 1977.

[29] D. Lusseau, K. Schneider, O. J. Boisseau, P. Haase, E. Slooten, and S. M. Dawson, "The bottlenose dolphin community of doubtful sound features a large proportion of long-lasting associations," *Behav. Ecol. Sociobiol.*, vol. 54, no. 4, pp. 396–405, Sep. 2003.

[30] M. Girvan and M. E. J. Newman, "Community structure in social and biological networks," *Proc. Nat. Acad. Sci. USA*, vol. 99, no. 12, pp. 7821–7826, Jun. 2002.

[31] P. M. Gleiser and L. Danon, "Community structure in jazz," *Adv Complex Syst.*, vol. 6, no. 4, pp. 565–573, 2003.

[32] *Overlapping Community Detection Implementation for the Algorithm From the Paper: Efficient Identification of Overlapping Communities (Intelligence and Security Informatics. 2005)*. Accessed: 2017. [Online]. Available: https://github.com/kritishrivastava/CommunityDetection-Project2GDM

[33] A. Lancichinetti and S. Fortunato, "Benchmarks for testing community detection algorithms on directed and weighted graphs with overlapping communities," *Phys. Rev. E, Stat. Phys. Plasmas Fluids Relat. Interdiscip. Top. Stat. Nonlinear Soft Matter Phys.*, vol. 80, no. 1, Aug. 2009, Art. no. 016118, doi: 10.1103/PhysRevE.80.016118.

[34] H. W. Shen, X. Q. Chen, K. Cai, and M. B. Hu, "Detect overlapping and hierarchical community structure in networks," *Phys. A, Stat. Mech. Appl.*, vol. 388, no. 8, pp. 1706–1712, Apr. 2009.

[35] M. Coscia, G. Rossetti, F. Giannotti, and D. Pedreschi, "Uncovering hierarchical and overlapping communities with a local-first approach," *ACM Trans. Knowl. Discov. Data.*, vol. 9, no. 1, pp. 1–27, Sep. 2014, doi: 10.1145/2629511.

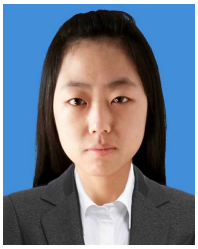
[36] A. Bojchevski and S. Gunnemann, "Deep Gaussian embedding of graphs: Unsupervised inductive learning via ranking," in *Proc. Int. Conf. Learn. Represent.*, Vancouver, BC, Canada, 2018, pp. 1–13.

[37] J. J. Whang, Y. Y. Hou, D. F. Gleich, and I. S. Dhillon, "Non-exhaustive, overlapping clustering," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 41, no. 11, pp. 2644–2659, Nov. 2019.

[38] K. Asmi, D. Hou, D. F. Lotfi, and M. E. Marraki, "Overlapping community detection based on the union of all maximum spanning trees," *Library Hi Tech.*, vol. 38, no. 2, pp. 276–292, Jan. 2020.

Hongtao Yu received the bachelor's degree in computer application from the Hefei University of Technology, Hefei, China, in 1987, and the master's degree in computer application technology from Yanshan University, Qinhuangdao, China, in 1993.

He is currently an Associate Professor with the School of Information Science and Engineering, Yanshan University. His main research interests include recommender systems, computer networks, and service-oriented computing.



Ru Ma received the master's degree in computer technology from Yanshan University, Qinhuangdao, China, in 2021, where she is currently pursuing the Ph.D. degree with the School of Information Science and Engineering.

Her main research interests include intelligent network information processing and information security.

Jinbo Chao received the bachelor's and master's degrees in computer application technology from Yanshan University, Qinhuangdao, China, in 2000 and 2004, respectively, where she is currently pursuing the Ph.D. degree in computer science and technology.

She is currently a Lecturer with Yanshan University. Her main research interests include recommender systems and information security.

Fuzhi Zhang received the Ph.D. degree in computer application technology from the Beijing Institute of Technology, Beijing, China, in 2004.

Since 1986, he has been working with Yanshan University, Qinhuangdao, China, where he is currently a Professor and a Ph.D. Supervisor. His main research interests include intelligent network information processing, information security, and service-oriented computing.