Role Discovery-Guided Network Embedding Based on Autoencoder and Attention Mechanism

Pengfei Jiao, Qiang Tian, Wang Zhang, Xuan Guo, Di Jin, and Huaming Wu, *Member, IEEE*

***Abstract*—Recently, network embedding (NE) is an amazing research point in complex networks and devoted to a variety of tasks. Nearly, all the methods and models of NE are based on the local, high-order, or global similarity of the networks, and few studies have focused on the role discovery or structural similarity, which is of great significance in spreading dynamics and network theory. Meanwhile, existing NE models for role discovery suffer from two limitations, that is: 1) they fail to model the varying dependencies between each node and its neighbor nodes and 2) they cannot capture the effective node features which are helpful to role discovery, which makes these methods ineffective when applied to the role discovery task. To solve the above problems of NE for role discovery or structural similarity, we propose a unified deep learning framework, called RDAA, which can effectively represent features of nodes and benefit the Role Discovery-guided NE with a deep autoencoder, while modeling the local links with an Attention mechanism. In addition, we design an elaborately binding technique to combine both parts and optimize the framework in a unified way. We conduct different experiments, including visualization, role classification, role discovery, and running time compared to popular NE methods for both proximity and structural similarity. The RDAA has better performance on all the datasets and achieves good tradeoffs.**

***Index Terms*—Attention mechanism, autoencoder, network representation, role discovery, structural similarity.**

I. Introduction

THE EXPLOSIVE growth of social media promotes the rapid development of social network analysis. Due to the sparsity of large-scale networks in the real world, how to effectively represent node information is extremely important for many network tasks [1]. Network embedding (NE), a recently most popular tool for representing and analyzing large-scale networks, aims to embed network nodes into low dimension space while preserving important characteristics of the network topology [2]. Then, the learned node representations that preserve the structure information can be applied to downstream tasks of network analysis, for example, node classification [3], community detection [4], recommendation systems [5], and link prediction [6].

A variety of methods and models has been proposed for NE, which can generally be divided into methods based on random walks, matrix factorization, statistical network models, and machine learning. Inspired by word2vec [7], DeepWalk [8] was first devoted to learning the node embedding based on the classical random walks on graphs and optimized via Hierarchical Softmax, and then some other methods are also developed with different types of random walks [6]. Singular value decomposition (SVD) and non-negative matrix factorization are widely applied in NE because of their optimality for low-rank characteristics. In addition, NE approaches based on neural networks, especially deep learning, have been continuously proposed [1], [9]. For instance, SDNE [10] can jointly exploit the first-order and second-order proximities for node embedding based on the autoencoder, DVNE [11] can maintain the network structure by integrating the uncertainty of node embedding based on variational autoencoders (VAEs), GraphSAGE [12] can leverage node feature information to efficiently generate node embeddings for previously unseen data, and besides, a theoretical framework for analyzing graph neural networks for embedding was proposed in [13], more efficient robust optimization [14] and community enhancement [15] methods are also proposed. More detailed reviews can be seen in [16]–[18]. Essentially, most of these works attempt to preserve the local (first and second similarity), high-order (motif and community similarity), or global proximity of the network structure [19] or a combination.

In a variety of real-world networks, however, there are some cases where two nodes have the same role or occupy the same position even though they do not share a connection or even are far apart [20], which is also known as the structural similarity problem. Our goal is to learn embeddings that can assign node roles. As shown in Fig. 1, although node f and node d are far apart in the network, they play the same role as both of them are surrounded by red nodes. This usually corresponds to the connotation of role discovery or structural similarity in complex networks, which can be used in online advertising campaigns of online social networks (e.g., Facebook, Twitter, and Yelp), and is of great significance in spreading, network theory, and network analysis [16]. Taking into account the particularity of this problem, however, all these methods have failed without exception.

Manuscript received 28 December 2020; revised 15 April 2021; accepted 29 June 2021. Date of publication 18 August 2021; date of current version 21 December 2022. This work was supported by the National Natural Science Foundation of China under Grant 61902278, Grant 61801325, and Grant 62071327. This article was recommended by Associate Editor J. Cao. *(Corresponding author: Huaming Wu.)*

Pengfei Jiao is with the College of Intelligence and Computing and the Center of Biosafety Research and Strategy, Law School, Tianjin University, Tianjin 300350, China (e-mail: pjiao@tju.edu.cn).

Qiang Tian, Wang Zhang, Xuan Guo, and Di Jin are with the College of Intelligence and Computing, Tianjin University, Tianjin 300350, China (e-mail: tianqiang@tju.edu.cn; wangzhang@tju.edu.cn; guoxuan@tju.edu.cn; jindi@tju.edu.cn).

Huaming Wu is with the Center for Applied Mathematics, Tianjin University, Tianjin 300072, China (e-mail: whming@tju.edu.cn).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TCYB.2021.3094893.

Digital Object Identifier 10.1109/TCYB.2021.3094893

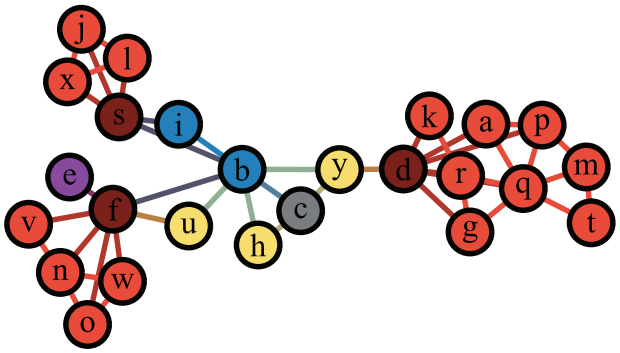


Fig. 1. Example of co-authorship networks (nodes represent the authors and links denote the cooperative relationships) illustrates the challenges role-based NE faced.

Specifically, role discovery or structural similarity [21] is usually expressed as searching a set of nodes with similar connection patterns in the network. In principle, there are indeed some different types of traditional ways that can solve this problem [22], for example, graph-based, feature-based, and hybrid methods. For all these methods, the most important step for role discovery is to extract node features either from complex networks or computing the similarity according to the adjacency matrix of the network. However, these traditional role discovery works, based on feature engineering [23] or matrix decomposition [24], are unable to deal with large-scale social networks. In the trend of NE, because of the essential difference between this problem and the previous node embedding, how to learn role discovery-guided NE is still an open and important issue [25].

Recently, a few schemes that focus on learning embedding for role discovery or structural similarity have been developed in [20] and [26]–[29]. Node embedding for the role or structural similarity, however, still faces many challenges as follows.

1) Nodes that play the same role are generally disconnected [30]. As depicted in Fig. 1, nodes with structurally similar neighbors, such as node d and node f, usually are far away in the network.
2) The role of a node depends on its neighbor nodes and has different dependencies with its neighbor nodes. As shown in Fig. 1, node f and node s have the same role because they are both connected to several red nodes, while the role of node f is independent of the roles of node e and node u. However, the current work like DRNE [20] ignores the varying dependencies between each node and its neighbor nodes. How to learn embeddings focusing on the most relevant dependencies between each node and its neighbors has not been resolved.
3) Extracting implicit role features from the network to guide role-based NE is an interesting but difficult task. Although node2vec [6] can extract role features to some extent through structural similarity [31], it is not enough to leverage the breadth-first search to obtain local features for role discovery. It is essential to study a role discovery-guided NE method to address these challenges.

On the one hand, for challenges 1 and 2, we design a role attention mechanism, which cannot only drive recursive similarity [21] through recursive learning steps (for challenge 1) but also naturally aggregate its neighbor nodes embedding (for challenge 2). On the other hand, for challenge 3, we take an effective feature extraction method to extract high-dimensional feature representations of nodes, which can reveal their roles in a given network. Although the two parts are essential to role discovery, it is unrealistic and inefficient to preserve varying dependencies between each node and its neighbors and features. Then, the most important problem is how to learn a node embedding with the role attention and the role feature matrix via a unified model, which makes the embeddings of nodes reveal the role characteristics more effectively.

In order to solve the aforementioned challenges, we design a unified deep learning framework, called RDAA, for role discovery guided NE. First, we use a recursive feature extraction method to obtain the high-dimensional role features of nodes, which may explore potential role characteristics that the observed structural information cannot achieve. Then, we leverage the autoencoder framework to encode that into a vector space and then derive a role attention mechanism to effectively describe the varying dependencies between each node with its neighbor nodes. Finally, we design an elaborately binding technique to combine the two parts together. In detail, through integrating the autoencoder and role attention based on the hidden feature representation for the attention mechanism, we design a joint loss function for training so that the above two parts can be mutually enhanced. In addition, extensive experiments are conducted both quantitatively and qualitatively to verify the effectiveness of the proposed approach.

The major contributions can be summarized as follows.

1) We design the RDAA, a unified deep learning framework for role discovery guided NE, which integrates the autoencoder and role attention mechanism.
2) The proposed deep learning framework can extract the features of the role, effectively depict the varying dependencies between each node and its neighbor nodes information and cleverly integrate the above two parts in principle.
3) The experimental results on role-based tasks, including structural role classification, role visualization, and a case study of role discovery on several real-world networks illustrate that the RDAA model can achieve significant improvements compared to other state-of-the-art methods.

II. Related Work

Here, we introduce some important works on role discovery, NE guided by role discovery or structural similarity, and the autoencoder framework.

A. Role Discovery

In network science, role discovery, which aims to discriminate the functions or behaviors of nodes [21], has been researched for decades [32]–[36]. It is considered that nodes in

similar roles have similar connectivity patterns (or higher order structure) [37]. Lorrain and White [32] first defined the roles of nodes based on structural equivalence. This criterion states that nodes connected to the same neighbors play the same role. It is so strict that loses effectiveness for nodes far from each other in the network. To relax this criterion, White and Reitz proposed regular equivalence [34] that nodes in the same role have role-equivalent neighbors. Nowicki and Snijders [36] proposed stochastic equivalence, which elucidates that the role of a node is determined by the distribution of neighbors' roles. No matter what later approaches are leveraged to mining node roles, they all follow or try to approximate these criteria. For example, RoleSim [35], a role similarity metric satisfying Axiomatic Role Similarity Properties, meets regular equivalence criteria via an iteratively computing process.

There are some other methods for role mining by using graph-based or feature-based structural similarities. ReFeX [23] is a recursive feature aggregation method that can obtain rich structural information. RolX [24] and GLRD [38] capture structural similarity by decomposing the ReFeX-based feature matrix. RC-Joint [39] aims to simultaneously detect communities and roles. While detecting roles, it leverages minwise hash signatures to effectively approximate the role similarity. REACT [40] and struc2gauss [41] take advantage of RoleSim and map the similarity between nodes into vector space. RIDεRs parties a network via ε-Equitable partitions and mines roles based on global features.

B. NE for Structural Similarity

Recently, motivated by NE, several methods have been proposed to preserve the structural similarity of the network and used for role discovery.

SNS [26] was first proposed to perform NE using structural information to enhance its quality, and essentially increases the stability of embeddings by means of structural similarity information. Struc2vec [30] transforms neighbor-degree-based similarity to edge weights in a hierarchy of complete graphs, and language models are utilized for embedding. Role2vec [42] replaces the input of language models with motif-count-based indicators. DRNE [20] learns a representation of each node by aggregating the embeddings of its neighborhoods in a recursive way. HONE [27] is a higher order network representation learning method for node embedding, which leverages multiple weighted motif adjacency matrices to capture the notion of structural roles. NRDR [28] learns the role-based embeddings based on two objectives: one is the cascade modeling objective, which aims to maximize the possibility of observed cascades, and the other is the matrix factorization objective, which aims to reconstruct structural proximities. GraphWave [43] leverages heat-wavelet diffusion patterns to represent nodes and treat them as distributions. SEGK [44] computes the structural similarity between nodes by applying graph kernel methods on node-centric subgraphs and generates embeddings by matrix factorization. SPaE [29] is a hybrid NE method based on the concept of the graphlet degree vector, which unifies both the structural proximity and similarity, simultaneously.

C. Autoencoder for Complex Networks

For its ability to model the high-order nonlinear characteristics of the network, the autoencoder [10], [45] framework and its extensions have been developed for NE.

SDNE [10] was first designed to incorporate the first- and second-order proximity of the network into the encoder–decoder framework and capture the network structure as semisupervised and unsupervised forms, respectively. The triplet enhanced autoencoder [46] can capture the topological structure and preserve the discriminative information based on metric learning. AAANE [47] consists of an attention-based autoencoder that constrains the posterior distribution of the latent embeddings and an adversarial regularization that constrains the prior distribution to be consistent and, thus, it can promote the collaboration of different scales for robust embeddings. Inspired by VAEs, Zhu *et al.* [11] proposed DVNE for NE in the Wasserstein spaces, which learned a Gaussian distribution for each node and preserved the transitivity of the network. Chen *et al.* [48] proposed variational graph embedding and clustering with the Laplacian eigenmaps, which was a generative model based on the VAE. The adversarial learning principle was introduced into the NE [49] and then the adversarially regularized graph autoencoder and autoencoder were proposed in [50] and [51], respectively. A general framework was proposed in [52] to scale the autoencoder and VAE to graphs with millions of nodes and edges by optimizing the reconstruction loss from nodes and propagating embeddings in the entire graph.

However, almost all the aforementioned methods are mainly designed for capturing the proximity in the network and cannot model the structural similarity and role discovery.

III. Methodology

In this section, we will introduce some notations and definitions about NE of role discovery, and then give the detailed construction process of the proposed framework, including the feature extraction module, encoder–decoder process, and role attention mechanism. Finally, we show how to optimize the loss function and its computational complexity.

A. Notations and Definitions

Given an undirected and unweighted network $G = (V, E)$, we denote V as the set of nodes and $E \subseteq V \times V$ as the set of edges. We define the neighbors of node i as $\mathcal{N}(i) = \{j|(i, j) \in E\}$, and also use $\mathbf{A}$ to represent the adjacency matrix, where $\mathbf{A}_{i,j} = 1$ means $\{(i, j) \in E\}$ and $\mathbf{A}_{i,j} = 0$ otherwise. NE is usually denoted to learn a mapping $\mathcal{F} : V \to \mathbb{R}^d$, where $d \ll |V|$ is the dimension of embedding and $n = |V|$ is the number of nodes. The symbols and their definitions are listed in Table I, which will be used in the following sections. Noting that our method can be easily extended to directed and weighted networks.

Definition 1 (Network Embedding): Given the topology information (such as the adjacency matrix) of the network G, the NE aims to learn a continuous vector $\mathbf{z}_i$ for each node i of G, such that the vectors can reveal the statistical structural characteristics of the network.

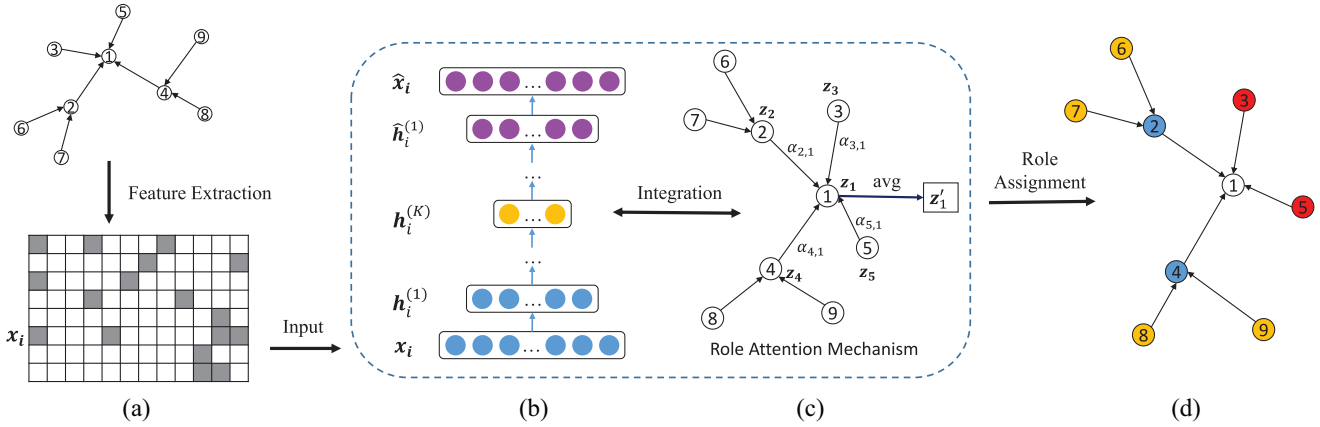


Fig. 2. Illustration of the proposed framework RDAA. It includes the following parts: (a) Extracting high-dimensional role-based features from the network. (b) Encoding these features into a low-dimensional embedding space and reconstructing the structural features. (c) Leveraging the mechanism of role attention to describe the varying dependencies based on the autoencoder process. (d) Applying to downstream machine-learning tasks, for example, role discovery or classification.

TABLE I
SYMBOLS AND DEFINITIONS

Symbol	Definition
n	The number of nodes
K	The number of layers
d	The number of embedding dimension
$\mathbf{z}_i \in R^d$	The embedding vector of node i
$\mathbf{x}_i$	The input role features of node i
$\hat{\mathbf{x}}_i$	The reconstructed feature for node i
$\mathbf{h}_i^{(k)}$	The k-th encoder layer's hidden representation of node i
$\hat{\mathbf{h}}_i^{(k)}$	The k-th decoder layer's hidden representation of node i
$\mathbf{W}^{(k)}$, $\mathbf{b}^{(k)}$	The weight matrix and biases of k-th encoder layer
$\hat{\mathbf{W}}^{(k)}$, $\hat{\mathbf{b}}^{(k)}$	The weight matrix and biases of k-th decoder layer
$\boldsymbol{\beta}_i$	The balance parameter vector on $\mathbf{x}_i$
$\mathbf{q}$	The query vector in attention mechanism

Definition 2 (Role Discovery Guided Network Embedding): Role discovery or structural similarity-based NE can be generally defined as the process of dividing nodes into equivalent classes. Let $r(i)$ be the role of node i, the NE for role discovery apply oneself to learn embedding vector $\mathbf{z_j}$ such that

$$r(i) = r(j) \Leftrightarrow S(\mathbf{z_i}, \mathbf{z_j}) \approx 1 \tag{1}$$

where $S(\cdot)$ is a function to measure the similarity of embedding $\mathbf{z_i}$ and $\mathbf{z_j}$. If the two nodes have the same roles, their embedding vectors should be more similar.

According to the above definitions and problem formulation, given any network G, our purpose is to learn embedding vectors of each node based on the set of edges. Meanwhile, we need to preserve the similarity between nodes i and j with the same role or the identical structural in the embedding space, that is, $\mathbf{z}_i$ and $\mathbf{z}_j$ should be close. It should be emphasized that our problem is different from the classic NE, in which the node embeddings are nearly only if they are proximity. Based on these embeddings, we can discover the role of nodes in the networks by any clustering method, for example, the k-means clustering.

B. Overall Framework

The illustration of our framework RDAA is shown in Fig. 2. First of all, we adopt a recursive feature extraction algorithm to extract high-dimensional features that can reveal role information. Then, we leverage the autoencoder framework to encode the network into a latent space and design an attention mechanism to effectively depict the varying dependencies between each node and its neighbors. Finally, by designing an ingenious technique, we jointly optimize the autoencoder and role attention mechanism to obtain the embeddings. In fact, we use the information of the network topology from two perspectives, that is: 1) the feature extraction and 2) the attention mechanism.

C. Role Feature Extraction

Since nodes having the same role are usually not in proximity, that is, they are far away in the network, so the adjacency matrix (a high-dimensional feature describing network topology structure), is very weak in extracting role-based features. So, we introduce a way to extract high-dimensional structural features, the dimension of which is less than $|V|$. Among countless feature extraction methods, we have adopted the recursive feature extraction method as described in [23]. This method aggregates local and neighborhood features in a recursive way to capture global structural information, which allows sufficient role features to be extracted. Specifically, for each node, it first generates primary features by counting its adjacent edges and that in the egonet of the node. Then, it aggregates the primary features recursively by applying some simple aggregation operators on the egonets, for example, sum and mean operators. Apparently, as the number of recursions increases, higher order structural properties could be captured. We report the process of it in Algorithm 1.

D. Encoder–Decoder

Although the RDAA model is flexible in feature extraction, in order to embed various features into a low-dimensional vector space and integrate the role attention mechanism described

Algorithm 1 Extraction Process of Role Features

Input: Adjacency matrix $\mathbf{A}$ of G; the number of iterations T.
Output: The role feature vector x_i for each node $i \in V$.

```
Compute the vector x_i^(0) for each node including node degree, egonet edge number and clustering coefficient;
x_i ← x_i^(0);
for t = 1 to T do
    for i ∈ V do
        Aggregate neighborhood features by concatenating their sum and mean values as x_i^(t) ← Σ_{j∈N(i)} x_j ∘ Σ_{j∈N(i)} x_j/|N(i)|;
        x_i ← x_i ∘ x_i^(t).
    end for
end for
return The role features {x_i}_{i=1}^n.
```

in the next section, here we introduce and extend the deep autoencoder for role feature reconstruction.

The encoder consists of a multilayer perceptron, which encodes the features into a low-dimensional representation. To be specific, given feature $\mathbf{x}_i$, the hidden representations for each layer can be expressed as

$$\mathbf{h}_i^{(1)} = \tanh\left(\mathbf{W}^{(1)}\mathbf{x}_i + \mathbf{b}^{(1)}\right) \tag{2}$$

$$\mathbf{h}_i^{(k)} = \tanh\left(\mathbf{W}^{(k)}\mathbf{h}_i^{(k-1)} + \mathbf{b}^{(k)}\right),\ k = 2, \ldots, K \tag{3}$$

where $\mathbf{x}_i$ represents the high-dimensional feature of node i.

With the part of an encoder, we can generate the node embedding $\mathbf{z}_i = \mathbf{h}_i^{(K)}$ of i. Then, the reconstructed data $\hat{\mathbf{x}}_i$ can be obtained by reversing the process of the encoder as follows:

$$\hat{\mathbf{h}}_i^{(k-1)} = \tanh\left(\hat{\mathbf{W}}^{(k)}\hat{\mathbf{h}}_i^{(k)} + \hat{\mathbf{b}}^{(k)}\right),\ k = K, \ldots, 2 \tag{4}$$

$$\hat{\mathbf{x}}_i = \tanh\left(\hat{\mathbf{W}}^{(1)}\hat{\mathbf{h}}^{(1)} + \hat{\mathbf{b}}^{(1)}\right) \tag{5}$$

where the node embedding $\mathbf{z}_i$ is denoted as $\hat{\mathbf{h}}_i^{(K)}$ for convenience.

The goal of the structure of encoder–decoder is to obtain the low-dimensional representations by minimizing the error between input features and the reconstructed data. The loss function of reconstruction is expressed as follows:

$$\mathcal{L}_{AE} = \sum_{i=1}^{n} \left\|\mathbf{x}_i - \hat{\mathbf{x}}_i\right\|_2^2. \tag{6}$$

Minimizing the reconstruction loss causes the representation of the node to smoothly capture the data manifolds and maintain the role features of the nodes in the low-dimensional space. However, considering that if there are a large number of zero elements in the features, our model would easily reconstruct the zero elements in $\mathbf{x}_i$, which is not desirable. To solve this issue, we introduce coefficients to correct the reconstruction loss function as follows:

$$\mathcal{L}_{AE} = \sum_{i=1}^{n} \left\|\left(\mathbf{x}_i - \hat{\mathbf{x}}_i\right) \odot \boldsymbol{\beta}_i\right\|_2^2 \tag{7}$$

where $\odot$ means the Hadamard product. Let x_{il} denote the lth element in $\mathbf{x}_i$ and $\boldsymbol{\beta}_{il}$ denotes the lth element in $\boldsymbol{\beta}_i$, then if $\mathbf{x}_{il} = 0$, $\boldsymbol{\beta}_{il} = 1$, else $\boldsymbol{\beta}_{il} = \beta > 1$, where β is a hyperparameter to control the contribution of nonzero elements. In fact, by introducing $\boldsymbol{\beta}_i$ for each node i, we can capture the different importance of different nodes.

E. Role Attention Mechanism

As mentioned in Section I, we have defined NE guided by role discovery or structural similarity. According to Definition 2, we constraint the node embedding in a recursive way by integrating its neighbor embeddings. Then, we can design a loss function based on the structural similarity constraint as follows:

$$\mathcal{L}_{\text{role}} = \sum_{i=1}^{n}\left(\left\|\mathbf{z}_i - \mathcal{G}\left(\left\{\mathbf{z}_j | j \in \mathcal{N}(i)\right\}\right)\right\|_2^2\right) \tag{8}$$

where $\mathcal{G} : \{\mathbf{z}_j | (j) \in V\} \to \mathbb{R}^d$ is an aggregate function used to aggregate the embeddings of neighbors. Obviously, node embedding can preserve local role information through $\mathcal{G}$ or obtain global role similarity information by updating the node embedding iteratively, which is consistent with the structural similarity for NE.

As we all known, the role of a node is closely related to the part of its neighborhood. As shown in Fig. 1, the role of node f depends on the red nodes surrounding it, while the roles of nodes e and u have little to do with the role of node f. In order to depict the varying dependencies between each node and its neighbors, we design an attention mechanism that allows for dealing with variably sized inputs and focusing on the most relevant parts of the input. We perform self-attention [53] on the nodes to compute attention-relevant coefficients

$$e_{ij} = a\left(\mathbf{z}_i, \mathbf{z}_j\right) \tag{9}$$

where $a : \mathbb{R}^d \times \mathbb{R}^d \to \mathbb{R}$ is a shared attention mechanism.

Unlike traditional attention mechanisms, we only focus on the neighbor nodes. Therefore, we only calculate e_{ij} on the neighbors, and then normalize them across all choices of j leveraging the softmax function as follows:

$$\begin{aligned}\alpha_{ij} &= \text{softmax}_j\left(e_{ij}\right) \\ &= \frac{\exp\left(\text{ReLU}\left(\mathbf{q}\left[\mathbf{z}_i \oplus \mathbf{z}_j\right]\right)\right)}{\sum_{k\in\mathcal{N}(i)}\exp(\text{ReLU}(\mathbf{q}[\mathbf{z}_i \oplus \mathbf{z}_k]))}\end{aligned} \tag{10}$$

where query vector $\mathbf{q} \in \mathbb{R}^{2d}$ is a weight vector and $\oplus$ represents the concatenation operation. Then, we define $\mathcal{G}$ by leveraging the normalized coefficient to calculate the linear combination of neighbor's node representations as follows:

$$\mathbf{z}_i' = \mathcal{G}\left(\left\{\mathbf{z}_j | j \in \mathcal{N}(i)\right\}\right) = \sigma\left(\sum_{j\in\mathcal{N}(i)} \alpha_{ij}\mathbf{z}_j\right) \tag{11}$$

where σ is the sigmoid activation function. Then, the loss function based on the structural similarity constraint of the role attention mechanism can be expressed as

$$\mathcal{L}_{\text{role}} = \sum_{i=1}^{n}\left(\left\|\mathbf{z}_i - \mathbf{z}_i'\right\|_2^2\right). \tag{12}$$

Algorithm 2 Optimization Algorithm for the RDAA

Input: $G = (V, E)$: the network;
K: the layer number of encoder and decoder;
β: the balance parameter of AE's reconstruction loss;
λ: the weight of loss function $\mathcal{L}_{role}$;
γ: the weight of loss function $\mathcal{L}_{reg}$.
Output: Embeddings $\{z_i\}_{i=1}^n$.

```
Extract structural features {x_i}_{i=1}^n via Algorithm 1;
Initialize the parameters of AE & attention mechanism;
for each epoch do
    Encode features {x_i}_{i=1}^n into embeddings {z_i}_{i=1}^n.
    Decode {z_i}_{i=1}^n into reconstructed features {x̂_i}_{i=1}^n;
    Combine attentional neighbor embeddings as {z'_i}_{i=1}^n;
    Compute losses L_AE, L_role and L_reg;
    Compute combined loss L and update all the parameters
    through backpropagation by minimizing it;
end for
Apply the optimized parameters to generate final embeddings through the encoder;
return embeddings {z_i}_{i=1}^n.
```

F. Joint Training

Through the two parts of loss functions, we can jointly optimize the unified deep learning framework consisting of the autoencoder and role attention mechanism. In order to prevent overfitting and further improve the robustness of RDAA, we define a regularizer as follows:

$$\mathcal{L}_{\text{reg}} = \sum_{k=1}^{K}\left(\left\|\mathbf{W}^{(k)}\right\|_F^2 + \left\|\hat{\mathbf{W}}^{(k)}\right\|_F^2 + \left\|\mathbf{b}^{(k)}\right\|_2^2 + \left\|\hat{\mathbf{b}}^{(k)}\right\|_2^2\right) + \|\mathbf{q}\|_2^2. \tag{13}$$

Then, the final joint loss function is designed as

$$\mathcal{L} = \mathcal{L}_{AE} + \gamma\mathcal{L}_{\text{role}} + \lambda\mathcal{L}_{\text{reg}} \tag{14}$$

where γ and λ are the weights of $\mathcal{L}_{\text{role}}$ and $\mathcal{L}_{\text{reg}}$, respectively.

We first generate feature matrix $\mathbf{X}$ through Algorithm 1 as the input of our model RDAA, and then randomly initialize all parameters of the model ($\mathbf{W}^{(k)}$, $\mathbf{b}^{(k)}$, $\hat{\mathbf{W}}^{(k)}$, $\hat{\mathbf{b}}^{(k)}$, and $\mathbf{q}$). For each node v_i, we can generate its embedding z_i as the last layer of the encoder through (3), which is used for reconstructing its feature x_i. To leverage the attention mechanism, we assign weight α_{ij} of its neighbor v_j through (10), and then obtain intermediate results to compute $\mathcal{L}_{\text{loss}}$. Through the joint training of (14), we cleverly integrate the autoencoder and role attention mechanisms. After obtaining the final loss $\mathcal{L}$, we can calculate the gradients of other parameters and use the backpropagation mechanism to update them to minimize it. We repeat this process many times until the loss converges. Finally, we apply the trained parameters to generate node embeddings.

G. Computational Complexity Analysis

The algorithm to optimize the objective function can be seen in Algorithm 2. For each node, the highest of time complexity is to calculate the normalized attention coefficient α, which is $O(Dd)$, where D is the maximum degree of nodes and d is the dimension of embedding, and then training and updating parameters take $O(Dd^2)$. Thus, the overall time complexity is denoted as $O(|V|Dd^2)$. Since the dimension d of node embedding is usually set to a small number and the degrees of most nodes are small, the time complexity of our algorithm is $O(|V|)$, which is linear with the number of nodes $|V|$ for large-scale networks.

IV. Performance Evaluation

In this section, we first introduce the commonly used datasets and baselines in our experiments. Then, we conduct visualization, structural role classification, and role discovery experiments, respectively. Furthermore, we analyze its sensitivity across two parameters.

A. Datasets and Metrics

Extensive experiments are conducted on several widely used real-world networks, which are listed as follows.

1) *Barbell Graph [20]:* It consists of two complete graphs C_1 and C_2 that connected by a path P. In our experiments, we use a barbell graph in which C_1, C_2, and P are all composed of ten nodes, as shown in Fig. 3(a).
2) *American, European, and Brazilian Air-Traffic Network [20]:* They are three air-traffic networks in which nodes refer to airports labeled according to the number of airplanes or personnel, and edges represent the existing flights between airports.
3) *Actor Co-Occurrence Network [25]:* In this network, nodes are actors, and edges between them mean that there is a co-occurrence of two actors appear on the same Wiki page. These actors are classified according to the word count on their pages.
4) *Ca-Netscience Network [54], [55]:* It describes the co-authorship of scientists engaged in the network theory and experiments, where nodes and links denote authors and cooperative relationships, respectively.

All these data are dedicated to role discovery and structural similarity, and the ground truth of these data comes from its original paper. In our quantitative experiments for classification, we use F1-micro and F1-macro as evaluation indicators, which are expressed as follows:

$$\text{F1-micro} = \frac{2 \times \text{P-micro} \times \text{R-micro}}{\text{P-micro} + \text{R-micro}} \tag{15}$$

$$\text{F1-macro} = \frac{2 \times \text{P-macro} \times \text{R-macro}}{\text{P-macro} + \text{R-macro}} \tag{16}$$

where P-micro, R-micro, P-macro, and R-macro are the macroscopic precision, macroscopic recall, microcosmic precision, and microcosmic recall [56], respectively, which are calculated based on the confusion matrix of classification.

B. Experimental Settings

To empirically evaluate the effectiveness of the RDAA model, we compare it with widely used embedding algorithms, including five proximity-based methods and eight role-based embedding methods. Their mechanisms are introduced here.

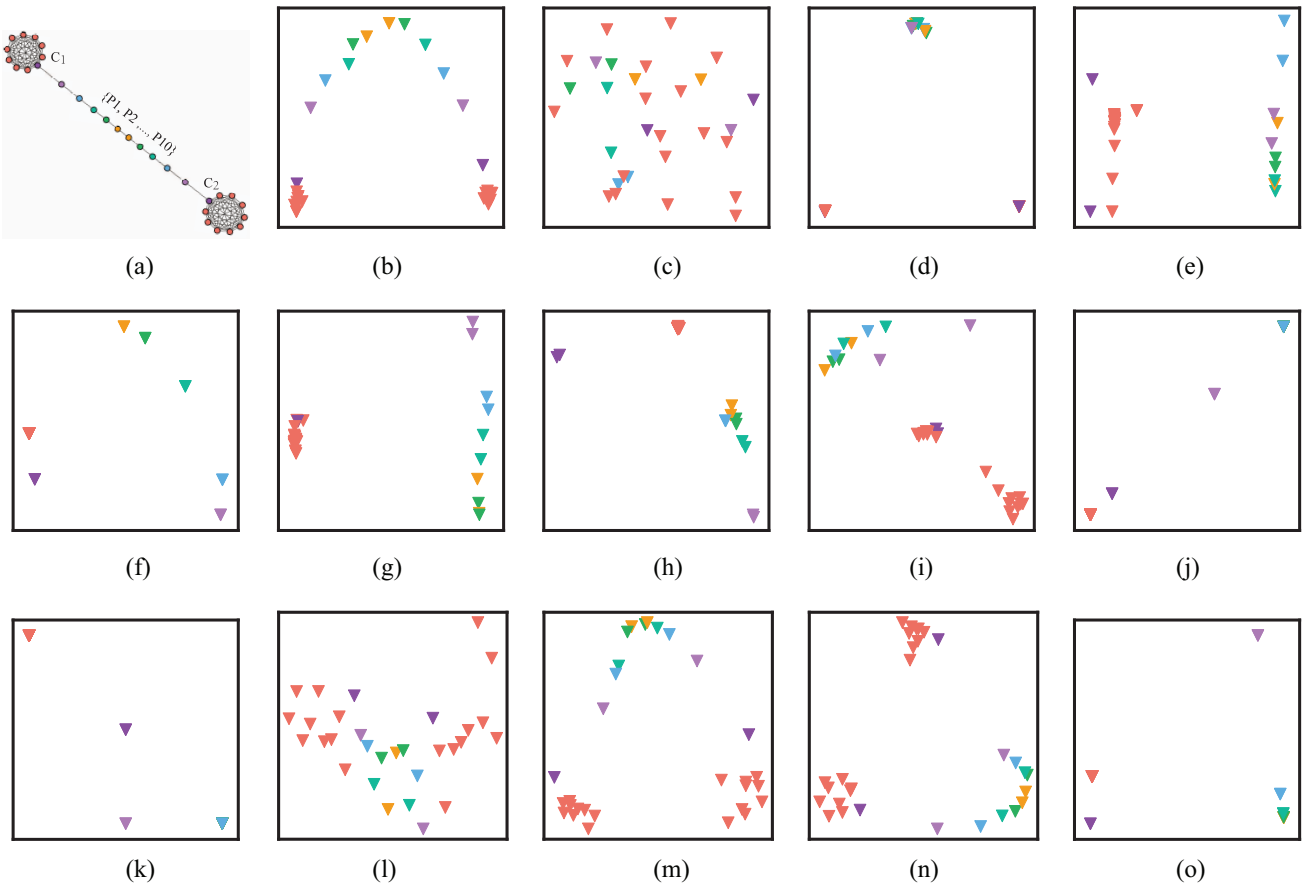


Fig. 3. Visualization results on the Barbell network. Nodes with the same color indicate that their roles are the same. (a) Topology structure of Barbell graph. (b)–(o) Visualization based on different NE methods. For it is a small network, here we set $d = 2$.

1) *Node2vec [6]:* It samples DFS-biased and BFS-biased random walks to preserve the proximity information. The proximity is mapped into vector space via the Skip-Gram model with nodes in walks treated as words in sentences.
2) *LINE [3]:* This method obtains representations by learning two kinds of relations: a) adjacence and b) triads. In specific, it uses empirical probabilities called first and second-order proximities to measure these relations and reconstructs them via the inner product of embeddings.
3) *SDNE [10]:* This method reconstructs the first-order and second-order proximities as LINE does, but via an autoencoder and Laplacian regularization.
4) *GraLSP [57]:* This method utilizes anonymous walks to express local structure. The anonymous walks are encoded into vector space for enhancing the aggregation part of graph neural networks via an attention mechanism. In addition, an objective for preserving walk-based similarity is proposed in GraLSP.
5) *GraphSTONE [58]:* GraphSTONE introduces topic models into graphs where it treats nodes as words and treats anonymous walks as documents. The structural topics are embedded via NMF and integrated with the GNN aggregator.
6) *Struc2vec [30]:* For struc2vec, a multilayer graph is first built based on degree-based distances to preserve structural similarities. Then, it applies random walks and a language model like node2vec.
7) *DRNE [20]:* DRNE aims to learn embeddings meeting regular equivalence, that is, equivalent nodes have equivalent neighbors, which leads to a recursive architecture of layer-normalized LSTMs. In addition, a degree-based regularizer is designed for avoiding undesired solutions.
8) *GraphWave [43]:* GraphWave computes the empirical characteristic function of wavelet distributions as structural embeddings, as the wavelet distributions indicate how energy spreads on the node-centered subgraphs.
9) *RolX [24]:* RolX is a classic method that factorizes an extracted structural feature matrix. The features of one node are attained by recursively aggregating its neighbors' features. As these features can finely characterize the local structure, we introduce them into our method.
10) *Struc2gauss [41]:* This method applies an algorithm, called RoleSim, which is designed for capturing regular equivalence, to compute pairwise similarity. It optimizes a KL-energy function empirically to learn the Gaussian embeddings.
11) *SEGK [44]:* SEGK introduces graph kernels into pairwise similarity computation. Similarities between different hops of neighborhoods are fused. It decomposes the fused similarity matrix into an embedding matrix.

TABLE II
STATISTICAL CHARACTERISTICS AND NEURAL-NETWORK STRUCTURES FOR EACH DATA

Dataset	#nodes	#edges	#classes	#the largest class	#the smallest class	#size of each layer
Barbell	30	101	7	18	2	$35-2$
American air-traffic network	1,190	13,599	4	299	297	$1,235-128$
European air-traffic network	399	5,993	4	35	32	$503-128$
Brazilian air-traffic network	131	1,003	4	102	99	$188-128$
Actor co-occurrence network	7,758	26,646	4	1,798	1,782	$132-128$
ca-netscience	379	914	9	102	9	$144-128$

12) *RIDεRs [22]:* This method combines the idea of epsilon equitable partitions with which it generates structural features. Embeddings are obtained via matrix factorization.
13) *RESD [59]:* RESD leverages the VAE on structural features to decrease noises and adds a regularizer to preserve the node degree, which is lost in the encoding process.

In the following experiments on role classification and discovery, the embedding dimension of all the baseline methods is set to 128 except for GraphWave and RIDεRs. GraphWave sets the default dimension to 100 while the dimension of RIDεRs is unfixed. To make the random walks of node2vec more BFS-biased, its two hyperparameters p and q are set to 1 and 2, respectively. Other parameters stay the same as the default values mentioned in their original paper. From the perspective of embedding mechanism, node2vec, struc2vec, and struc2gauss are based on the random walk, LINE, GraphWave, SEGK, RIDεRs, and RolX are from the low-rank matrix factorization, SDNE, DRNE, GraLSP, GraphSTONE, and RESD are motivated by the deep learning. The RDAA framework includes a multilayer deep neural network, and its settings on different datasets are listed in Table II. For the proposed RDAA model, we set the hyperparameter $\beta = 3$, the weight of loss function $\mathcal{L}_{\text{role}}$ $\gamma = 15$, and the weight of loss function $\mathcal{L}_{\text{reg}}$ $\lambda = 0.02$. The Adadelta optimization algorithm [60] is adopted to train the model. We have shared the code on https://github.com/cspjiao/RDAA.

C. Visualization

To illustrate the beneficial performance of the RDAA model when compared to some baselines on role discovery, we conduct several visualization experiments on the Barbell graph as shown in Fig. 3, from which we can draw important observations as follows.

1) The RDAA model performs much better than LINE, node2vec, SDNE, GraLSP, and GraphSTONE, because the role discovery guided NE can make the nodes with the same role closer in the embedding space. LINE cannot explore the similarity of networks very well since it only considers the pairwise proximities. Although node2vec considers structural similarity, it still cannot capture global structural similarity information regardless of different combinations of model parameters p and q. SDNE is designed to model the nonlinearity of the network and it seems to project the nodes with the same roles into nearby points, but it assigns almost all the nodes on the path P together, this is obviously wrong. GraLSP and GraphSTONE can effectively capture proximities in the networks, but fail to embed role-oriented networks in this field.
2) Although DRNE, RolX, struc2gauss, and RESD attempt to learn the structural features of nodes belonging to the same roles, they all fail to capture the characteristics of the nodes of two regiments. DRNE focuses on the influence of the node degree and ignores high-order properties and the nodes are obviously divided into two parts: a) the nodes on the path and b) the two cliques. RolX is based on feature extraction and non-negative matrix factorization, ignoring the nonlinear structure of the network. struc2guass computes role similarities via RoleSim, which is hard to be discriminated the delicate role differences of the nodes in the path. RESD focuses on reconstructing the features while ignoring the dependency between nodes
3) Relatively speaking, the SEGK, RIDεRs, and RDAA models are more powerful to discriminate the nodes in the path P. They all discover the nodes of the same role, especially the two regiments, as shown in Fig. 3. This is because these methods can effectively model the similarity characteristics of nodes and learn the embedding guided by role discovery. However, the other two methods will more or less lose some node information because they are overfitting the network.
4) It should be emphasized that the struc2vec and GraphWave are the closest to our method, however, their vectors of cyan nodes are inferior to our method. In addition, these two methods have higher complexity.

D. Experiments on Role Classification

To evaluate the role-based embedding quantitatively, we conduct experiments on structural role classification, which is the most important and intuitive task. We evaluate the ability and performance of the RDAA model on the American, European, and Brazilian air-traffic networks and the Actor co-occurrence network based on predicting the roles of nodes. For each of the baselines and the RDAA model, we randomly sample 10%–90% of the learned embeddings as the training set for training a simple logistic regression classifier and regard the other part as the test set for evaluating the performance. We take ten-fold cross-validation and report the mean value in the same strategy as the other methods. Finally, we evaluate

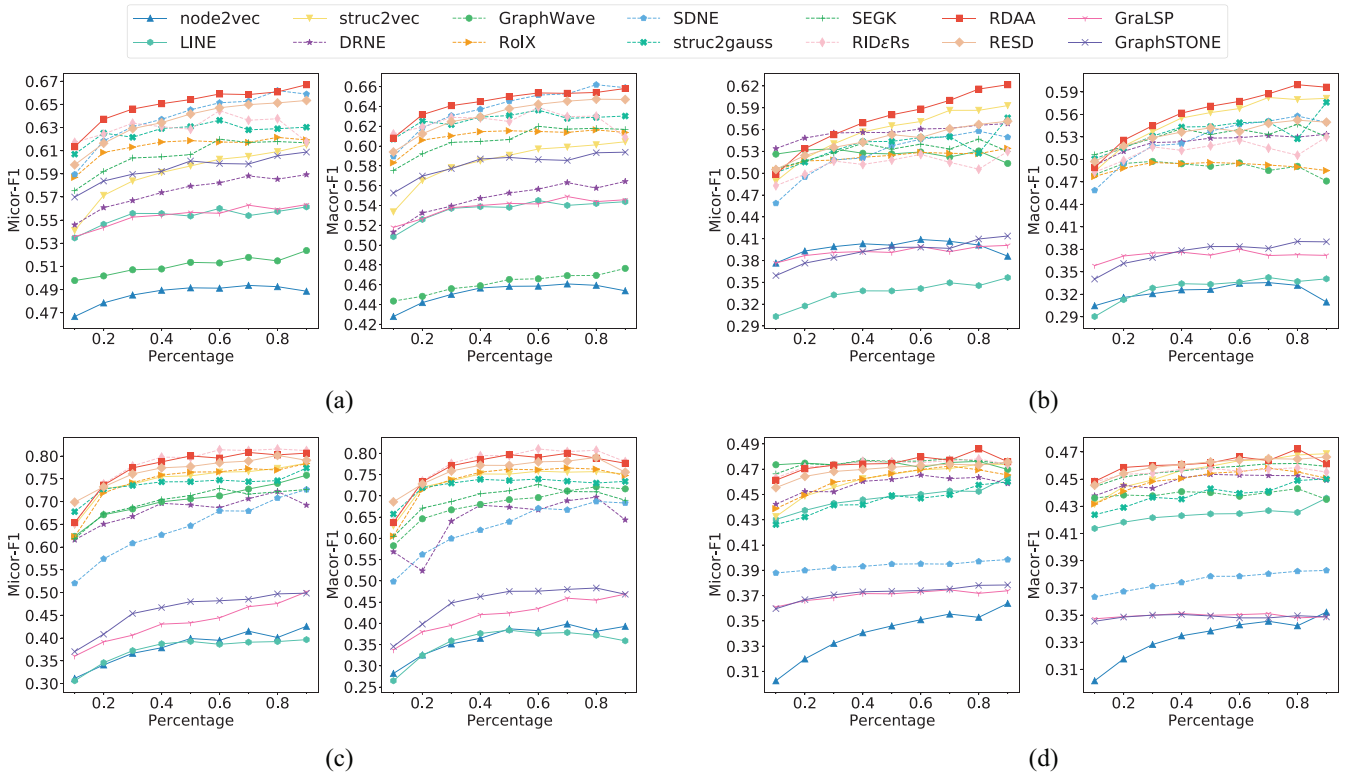


Fig. 4. Performance of different embedding methods on structural role classification in four different real-world networks. The results are computed based on the F1-micro and F1-macro via means of tenfold cross-validation. (a) American air-traffic network. (b) European air-traffic network. (c) Brazilian air-traffic network. (d) Actor co-occurrence network.

the performance with F1-micro and F1-macro. The detailed experimental results are shown in Fig. 4, from which we have the following observations.

Generally speaking, our proposed RDAA is superior to almost all the baselines on the four networks, indicating that RDAA performs effectively on structural role classification and can predict the role very well. Only the RID$\boldsymbol{\varepsilon}$Rs has a slight advantage over the Brazilian air-traffic network. RESD has competitive results on this task on these networks. The node2vec and LINE models have poor performance for that they focus on the proximity and ignore the structural similarity of nodes. SDNE has poor performance on the Brazilian air-traffic and Actor co-occurrence networks. All other methods for role-based embedding are second-best level and stable on the role classification. Specifically, all the methods show greater F1 scores on the American air-traffic network than the European air-traffic network. Simultaneously, the RDAA model achieves greater performance improvements on the American air-traffic network. Due to the complex structure of the American air-traffic network, our role attention mechanism is able to reveal the varying dependencies between the node and its neighbors more precisely.

Furthermore, we employ statistical tests to see the significance and stability of the RDAA model and we compare it with ten other baselines on the American, European, and Brazilian air-traffic networks and the Actor co-occurrence network. Here, we set the proportion of training set as 0.7 with 20 random runs. The performance on the F1-micro and F1-macro is shown in Table III, in which the means and the standard deviations of the results for each method on different datasets are listed. It is easy to know that the standard deviations of all the methods are relatively small and there is no obvious difference.

Besides, we also compare the results with several variants of RDAA (some ablation experiments). Here, RDAE denotes the model without $\mathcal{L}_{\text{role}}$, where we have deleted the part of attention. On the other hand, we evaluate another type of loss function of $\mathcal{L}_{AE}$ in RDMAA, in which we replace it with $\mathcal{L}_{AE} = \sum_{i=1}^{n} |(\mathbf{x}_i - \hat{\mathbf{x}}_i) \odot \boldsymbol{\beta}_i|$. Based on the two ablation models and the results in Table III, we know that RDAA has obvious advantages and shows that every part of it is effective. Furthermore, we also report the running time of all these methods in Table IV, our method has a compromise level.

E. Parameters Sensitivity

In this part, we investigate whether and how the parameters affect performance. Specifically, we evaluate the effectiveness of the dimensions d of the node representations, and the hyperparameters β and γ. We apply the structural role classification task on the American, European, and Brazilian air-traffic networks, respectively.

From Fig. 5(a), we observe that the performance is significantly improved initially while the dimension d increases. Then, the effect tends to be stable when the dimension is greater than 16, which proves the robustness and effectiveness of the RDAA model.

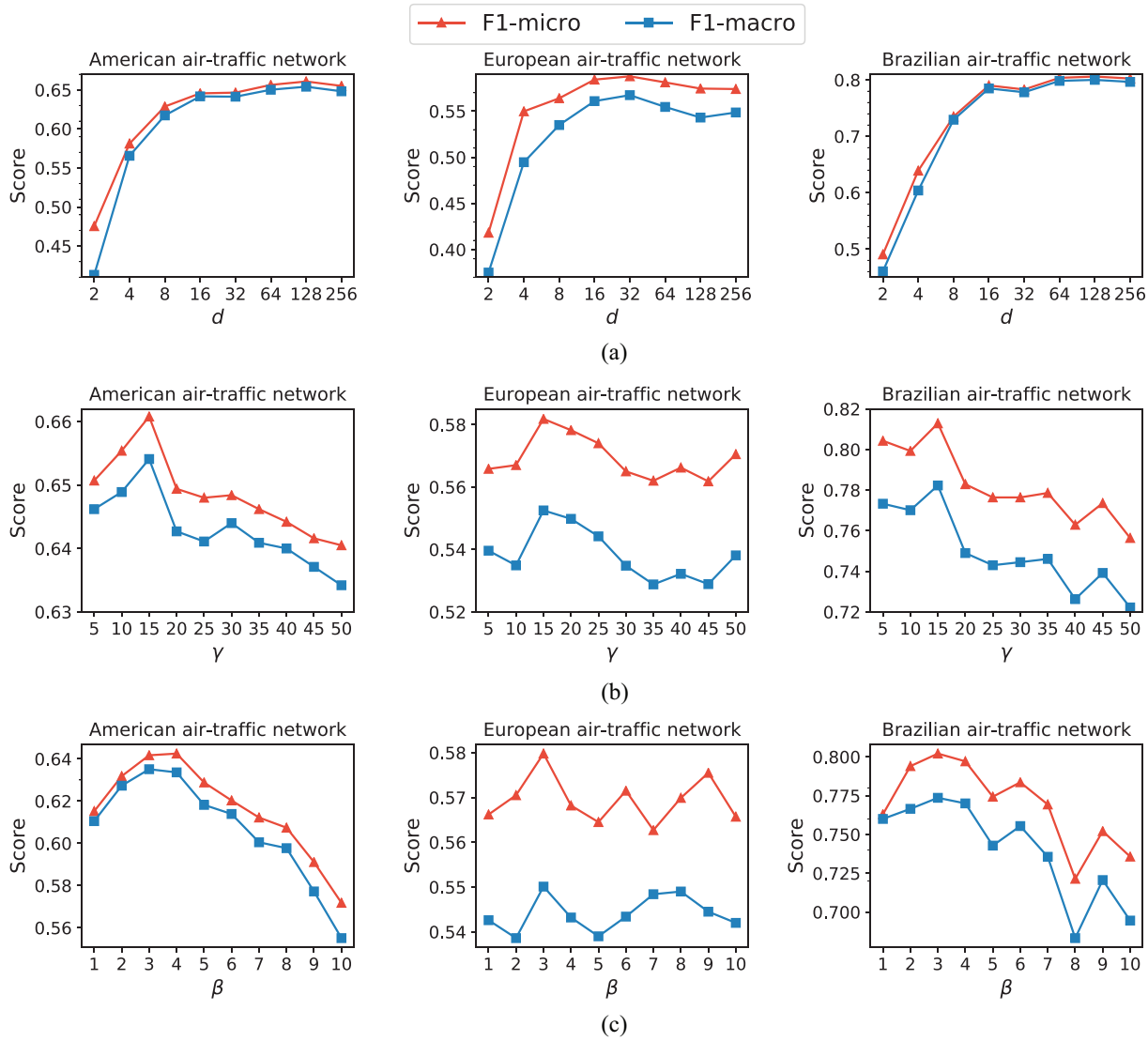


Fig. 5. Sensitivity analysis based on the F1-micro and F1-macro of parameters with respect to the number of dimension d, and hyperparameters γ and β, respectively. (a) Number of dimension d. (b) Hyperparameter γ. (c) Hyperparameter β.

Fig. 5(b) indicates that with the increase of the hyperparameter γ that controls the contribution of $\mathcal{L}_{\text{role}}$, the performance first increases and then gradually flattens out. The increase in performance demonstrates the effectiveness of the role attention mechanism in the RDAA model. When $\gamma = 15$, our method has the best performance on all the networks with regard to both F1-micro and F1-macro metrics.

Fig. 5(c) shows the impact of the hyperparameter β, which is used to penalize the nonzero elements harder than zero elements on the reconstruction error. On the whole, our method can achieve almost the best performance when $\beta = 3$.

F. Efficient versus Effective

As depicted in Fig. 6, we provide a comparison of embedding time and classification results of different methods on the European and American air-traffic network. It can be observed that there exists a tradeoff between the effectiveness and efficiency of these methods. On the one hand, our RDAA model has the best classification results among all the methods, which demonstrate the superiority effectiveness of RDAA for role discovery. On the other hand, the embedding time for RDAA is slightly longer than that of SEGK and struc2gauss, which means the efficiency of RDAA is not as good as SEGK and struc2gauss; however, it is within an acceptable range. The RESD has competitive performance and slightly more efficient compared to the RDAA.

On the whole, the RDAA model achieves the best performance in terms of the F1-micro score with only a small portion of embedding time sacrificed.

G. Case Study: Role Discovery

A case study on the ca-netscience network is conducted to show the effectiveness of the RDAA model for role discovery. We first train the model on this network and generate a representation for each node. Then, we assign roles for them as the task of clustering through the K-Means model. By doing this, the performance of these methods can be evaluated

TABLE III
STRUCTURAL ROLE CLASSIFICATION THAT IDENTIFIES STRUCTURALLY SIMILAR NODES. THE PERFORMANCE REPORTED IS THE AVERAGE AND STANDARD DEVIATION OF F1-MICRO AND F1-MACRO SCORES FOR 20 RUNS WITH 70 PERCENT OF TRAINING SAMPLES

Method	Indicator	America	Europe	Brazil	Actor
node2vec	F1-micro	0.4958 ± 0.0560	0.3808 ± 0.0608	0.4800 ± 0.0950	0.3594 ± 0.0108
	F1-macro	0.4923 ± 0.0483	0.4071 ± 0.0625	0.4526 ± 0.1166	0.3462 ± 0.0232
LINE	F1-micro	0.6322 ± 0.0513	0.4967 ± 0.0700	0.4138 ± 0.1113	0.4493 ± 0.0144
	F1-macro	0.6140 ± 0.0411	0.4586 ± 0.0637	0.2104 ± 0.0938	0.4242 ± 0.0142
struc2vec	F1-micro	0.6207 ± 0.0319	0.5721 ± 0.0612	0.7638 ± 0.0862	0.4658 ± 0.0105
	F1-macro	0.6105 ± 0.0562	0.5679 ± 0.0502	0.7581 ± 0.1063	0.4579 ± 0.0165
DRNE	F1-micro	0.5710 ± 0.0368	0.5721 ± 0.0779	0.6712 ± 0.1288	0.4632 ± 0.0112
	F1-macro	0.5606 ± 0.0535	0.5470 ± 0.0552	0.6613 ± 0.0872	0.4466 ± 0.0126
GraphWave	F1-micro	0.5148 ± 0.0398	0.5367 ± 0.0300	0.7537 ± 0.0713	0.4744 ± 0.0205
	F1-macro	0.4691 ± 0.0231	0.4923 ± 0.0706	0.7564 ± 0.0760	0.4460 ± 0.0136
RolX	F1-micro	0.6146 ± 0.0493	0.5621 ± 0.0463	0.7762 ± 0.0738	0.4695 ± 0.0193
	F1-macro	0.6191 ± 0.0198	0.5348 ± 0.0683	0.7490 ± 0.1016	0.4561 ± 0.0138
SDNE	F1-micro	0.6422 ± 0.0413	0.5704 ± 0.0796	0.7163 ± 0.1087	0.3973 ± 0.0199
	F1-macro	0.6227 ± 0.0515	0.5332 ± 0.0584	0.6396 ± 0.1164	0.3757 ± 0.0138
struc2gauss	F1-micro	0.6290 ± 0.0349	0.5579 ± 0.0837	0.7263 ± 0.0987	0.4493 ± 0.0093
	F1-macro	0.6249 ± 0.0353	0.5563 ± 0.0393	0.7302 ± 0.1836	0.4436 ± 0.0201
SEGK	F1-micro	0.6175 ± 0.0548	0.5208 ± 0.0625	0.7362 ± 0.1138	0.4772 ± 0.0130
	F1-macro	0.6185 ± 0.0541	0.5155 ± 0.0700	0.7146 ± 0.1282	0.4592 ± 0.0117
RIDεRs	F1-micro	0.6360 ± 0.0447	0.5283 ± 0.0633	$\mathbf{0.7925 \pm 0.1075}$	0.4767 ± 0.0145
	F1-macro	0.6226 ± 0.0400	0.4719 ± 0.0601	$\mathbf{0.7793 \pm 0.0893}$	0.4578 ± 0.0138
RESD	F1-micro	0.6506 ± 0.0023	0.5587 ± 0.0048	0.7858 ± 0.0062	0.4736 ± 0.0010
	F1-macro	0.6459 ± 0.0024	0.5447 ± 0.0047	0.7779 ± 0.0065	0.4649 ± 0.0010
GraLSP	F1-micro	0.5610 ± 0.0022	0.3972 ± 0.0037	0.4680 ± 0.0067	0.3734 ± 0.0008
	F1-macro	0.5466 ± 0.0020	0.3763 ± 0.0035	0.4557 ± 0.0071	0.3498 ± 0.0007
GraphSTONE	F1-micro	0.6039 ± 0.0016	0.4013 ± 0.0042	0.4867 ± 0.0056	0.3752 ± 0.0010
	F1-macro	0.5918 ± 0.0015	0.3854 ± 0.0042	0.4785 ± 0.0055	0.3481 ± 0.0009
RDAE	F1-micro	0.6291 ± 0.0207	0.5658 ± 0.0508	0.7775 ± 0.1225	0.4687 ± 0.0150
	F1-macro	0.6208 ± 0.0308	0.5620 ± 0.0606	0.7383 ± 0.1090	0.4584 ± 0.0193
RDMAA	F1-micro	0.6143 ± 0.0272	0.5617 ± 0.0550	0.6500 ± 0.0750	0.4734 ± 0.0127
	F1-macro	0.6091 ± 0.0311	0.5411 ± 0.0561	0.6435 ± 0.1693	0.4411 ± 0.0103
RDAA	F1-micro	$\mathbf{0.6487 \pm 0.0599}$	$\mathbf{0.6092 \pm 0.0658}$	0.7912 ± 0.1588	$\mathbf{0.4807 \pm 0.0156}$
	F1-macro	$\mathbf{0.6384 \pm 0.0523}$	$\mathbf{0.5815 \pm 0.0677}$	0.7729 ± 0.1068	$\mathbf{0.4662 \pm 0.0201}$

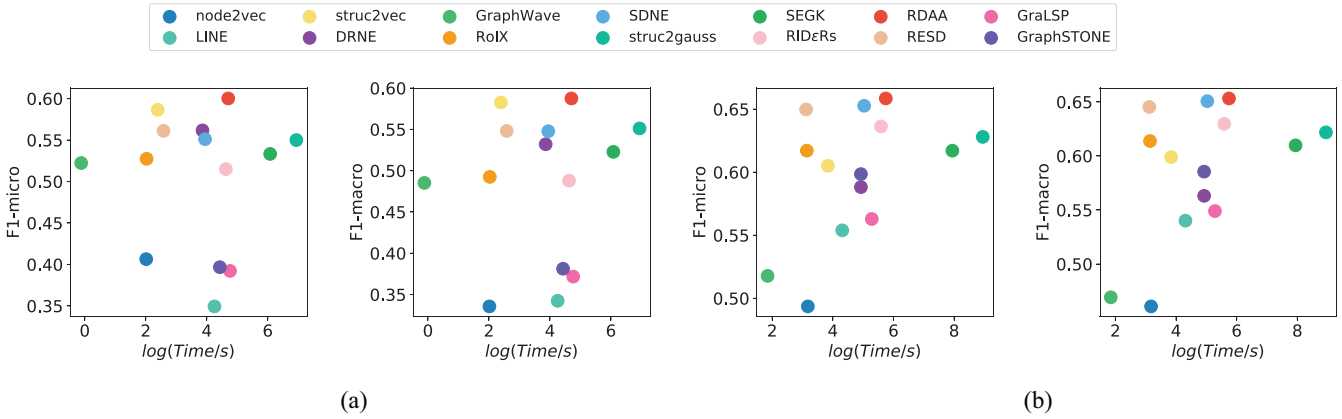


Fig. 6. Comparison of embedding time and classification results of different methods. (a) European air-traffic network. (b) American air-traffic network.

by analyzing whether the embeddings reflect the structural similarities.

Intuitively, we can observe from Fig. 7 that node2vec, LINE, and SDNE cannot conduct role mining through clustering. SDNE is obviously a community-based embedding method since the network is partitioned into several subgraphs in which nodes are densely connected. Since LINE can only preserve the first-order and second-order proximities in a shallow manner, the nodes in the same colors usually occur as neighbors or have the same neighbors. Both GraLSP and GraphSTONE learn proximity-oriented results for the local propagation mechanism in graph neural networks. Due to our

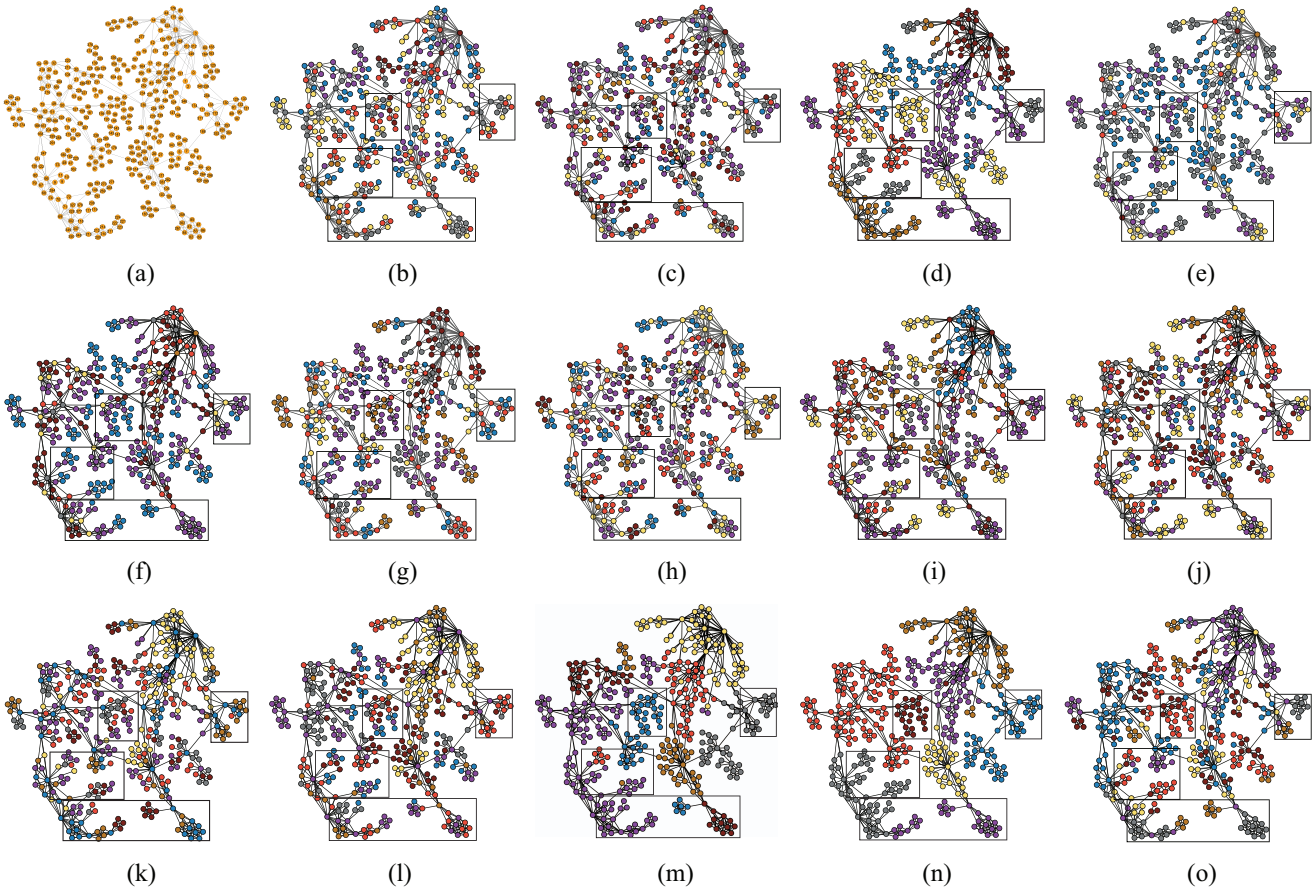


Fig. 7. Case study of role discovery on the ca-netscience network, where the color of the node represents the category of the role. (a) Topological structure of the network. (b)–(o) Results of role discovery based on different methods, and we highlight some results with a few rectangles.

TABLE IV
RUNNING TIME (S) OF DIFFERENT METHODS. THE OT MEANS IT IS BEYOND OUR TOLERANCE

Method	America	Europe	Brazil	Actor
node2vec	23.97	7.50	1.60	106.92
LINE	74.48	70.34	69.18	89.61
struc2vec	46.41	10.95	2.43	311.84
DRNE	137.93	47.45	8.34	504.91
GraphWave	6.35	0.89	0.13	222.92
RolX	23.17	7.60	2.14	154.90
SDNE	153.26	51.67	28.30	2492.59
struc2gauss	7663.78	1030.28	15.24	8445.69
SEGK	2813.18	436.58	12.80	*OT*
RIDεRs	267.46	102.09	26.17	312.46
RESD	22.63	13.25	11.79	95.23
GraLSP	197.16	117.03	83.38	1129.63
GraphSTONE	137.85	83.90	62.27	431.16
RDMAA	316.44	116.69	50.16	1917.78
RDAA	313.10	110.33	46.18	2000.44

BFS-biased parameter setting, node2vec shows limited ability to capture structural information: almost all colors appear in each local part; however, no significant patterns can be observed. The results of DRNE, struc2vec, and RIDεRs show a strong correlation with node degrees, which is mainly dependent on their degree-based processes. While DRNE also colors some nodes in a similar way to struc2gauss as they are both designed to capture regular equivalence. RolX, struc2vec, SEGK, and RIDεRs cluster nodes in a very hard way due to their shallow and hard structural information computation, which makes the visualization results look a little chaotic. RESD divides the nodes with the same degrees into one cluster for its direct regularization term. On the contrary, the performance of GraphWave and our RDAA is superior to other embedding methods, especially on the subgraphs highlighted by black rectangles. However, our RDAA also performs much better than GraphWave on distinguishing star and clique-like subgraphs. All these results demonstrate that the RDAA model can more effectively mine the roles of the nodes.

V. CONCLUSION

This article proposed a unified deep learning framework for role discovery and structural similarity guided NE, which integrates the autoencoder and role attention mechanisms. In particular, the proposed role attention mechanism effectively depicts the varying dependencies between each node and its neighbor nodes. We also designed an elaborately binding technique that combines the two parts together and optimizes the proposed framework in a unified way. The experimental

results demonstrated the effectiveness of the RDAA model since it does capture some unique information. Furthermore, the proposed RDAA model can easily be extended to different forms. First, the feature extraction can be replaced in any way that could indicate the role-based structural features. Then, the autoencoder process can be interchanged by the VAE, graph convolutional networks, or others. Moreover, we can expand RDAA to large-scale networks in a manner similar to the degeneracy framework [52].

One disadvantage is that the proposed model is a relatively independent module for feature extraction; however, how to design an end-to-end framework for the RDAA model is a potential research point. In the future, we will study how to improve RDAA so that it can be applied to role discovery for both nodes and edges, and we also plan to extend the proposed model to a multilayer network and dynamic network analysis. Moreover, quantitative evaluation is also a research direction for this problem.

REFERENCES

[1] X. Shen and F.-L. Chung, "Deep network embedding for graph representation learning in signed networks," *IEEE Trans. Cybern.*, vol. 50, no. 4, pp. 1556–1568, Apr. 2020.

[2] H. Cai, V. W. Zheng, and K. C.-C. Chang, "A comprehensive survey of graph embedding: Problems, techniques, and applications," *IEEE Trans. Knowl. Data Eng.*, vol. 30, no. 9, pp. 1616–1637, Sep. 2018.

[3] J. Tang, M. Qu, M. Wang, M. Zhang, J. Yan, and Q. Mei, "LINE: Large-scale information network embedding," in *Proc. World Wide Web Conf.*, 2015, pp. 1067–1077.

[4] 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. 2019.

[5] Z. Guo and H. Wang, "A deep graph neural network-based mechanism for social recommendations," *Proc. IEEE Trans. Ind. Informat.*, vol. 17, no. 4, pp. 2776–2783, Apr. 2020.

[6] A. Grover and J. Leskovec, "NODE2VEC: Scalable feature learning for networks," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Disc. Data Min.*, 2016, pp. 855–864.

[7] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Proc. Adv. Neural Inf. Process. Syst.* vol. 26, 2013, pp. 3111–3119.

[8] B. Perozzi, R. Al-Rfou, and S. Skiena, "DeepWalk: Online learning of social representations," in *Proc. 20th ACM SIGKDD Int. Conf. Knowl. Disc. Data Min.*, 2014, pp. 701–710.

[9] M. Lei, P. Quan, R. Ma, Y. Shi, and L. Niu, "DigGCN: Learning compact graph convolutional networks via diffusion aggregation," *IEEE Trans. Cybern.*, early access, May 18, 2020, doi: 10.1109/TCYB.2020.2988791.

[10] D. Wang, P. Cui, and W. Zhu, "Structural deep network embedding," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Disc. Data Min.*, 2016, pp. 1225–1234.

[11] D. Zhu, P. Cui, D. Wang, and W. Zhu, "Deep variational network embedding in Wasserstein space," in *Proc. 24th ACM SIGKDD Int. Conf. Knowl. Disc. Data Min.*, 2018, pp. 2827–2836.

[12] W. L. Hamilton, R. Ying, and J. Leskovec, "Inductive representation learning on large graphs," in *Proc. 31st Int. Conf. Neural Inf. Process. Syst.*, 2017, pp. 1025–1035.

[13] K. Xu, W. Hu, J. Leskovec, and S. Jegelka, "How powerful are graph neural networks?" in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2019, pp. 1–17.

[14] S. Wang, J. Liu, and Y. Jin, "Surrogate-assisted robust optimization of large-scale networks based on graph embedding," *IEEE Trans. Evol. Comput.*, vol. 24, no. 4, pp. 735–749, Aug. 2019.

[15] M. Li, J. Liu, P. Wu, and X. Teng, "Evolutionary network embedding preserving both local proximity and community structure," *IEEE Trans. Evol. Comput.*, vol. 24, no. 3, pp. 523–535, Jun. 2020.

[16] P. Cui, X. Wang, J. Pei, and W. Zhu, "A survey on network embedding," *IEEE Trans. Knowl. Data Eng.*, vol. 31, no. 5, pp. 833–852, May 2019.

[17] H. Chen, B. Perozzi, R. Al-Rfou, and S. Skiena, "A tutorial on network embeddings," 2018. [Online]. Available: arXiv:1808.02590.

[18] D. Zhang, J. Yin, X. Zhu, and C. Zhang, "Network representation learning: A survey," *IEEE Trans. Big Data*, vol. 6, no. 1, pp. 3–28, Mar. 2020.

[19] Z. Zhang, P. Cui, X. Wang, J. Pei, X. Yao, and W. Zhu, "Arbitrary-order proximity preserved network embedding," in *Proc. 24th ACM SIGKDD Int. Conf. Knowl. Disc. Data Min.*, 2018, pp. 2778–2786.

[20] K. Tu, P. Cui, X. Wang, P. S. Yu, and W. Zhu, "Deep recursive network embedding with regular equivalence," in *Proc. 24th ACM SIGKDD Int. Conf. Knowl. Disc. Data Min.*, 2018, pp. 2357–2366.

[21] R. A. Rossi and N. K. Ahmed, "Role discovery in networks," *IEEE Trans. Knowl. Data Eng.*, vol. 27, no. 4, pp. 1112–1131, Apr. 2015.

[22] P. V. Gupte, B. Ravindran, and S. Parthasarathy, "Role discovery in graphs using global features: Algorithms, applications and a novel evaluation strategy," in *Proc. IEEE 33rd Int. Conf. Data Eng. (ICDE)*, 2017, pp. 771–782.

[23] K. Henderson *et al.*, "It's who you know: Graph mining using recursive structural features," in *Proc. 17th ACM SIGKDD Int. Conf. Knowl. Disc. Data Min.*, 2011, pp. 663–671.

[24] K. Henderson *et al.*, "RolX: Structural role extraction & mining in large graphs," in *Proc. 18th ACM SIGKDD Int. Conf. Knowl. Disc. Data Min.*, 2012, pp. 1231–1239.

[25] X. Ma, G. Qin, Z. Qiu, M. Zheng, and Z. Wang, "RiWalk: Fast structural node embedding via role identification," in *Proc. IEEE Int. Conf. Data Min. (ICDM)*, Nov. 2019, pp. 478–487.

[26] T. Lyu, Y. Zhang, and Y. Zhang, "Enhancing the network embedding quality with structural similarity," in *Proc. ACM Conf. Inf. Knowl. Manag.*, 2017, pp. 147–156.

[27] R. A. Rossi, N. K. Ahmed, and E. Koh, "Higher-order network representation learning," in *Proc. World Wide Web Conf. (WWW)*, 2018, pp. 3–4.

[28] Z. Wang, C. Chen, and W. Li, "Information diffusion prediction with network regularized role-based user representation learning," *ACM Trans. Knowl. Disc. Data*, vol. 13, p. 29, Jul. 2019.

[29] B. Shi, C. Zhou, H. Qiu, X. Xu, and J. Liu, "Unifying structural proximity and equivalence for network embedding," *IEEE Access*, vol. 7, pp. 106124–106138, 2019.

[30] L. F. Ribeiro, P. H. Saverese, and D. R. Figueiredo, "Struc2vec: Learning node representations from structural identity," in *Proc. 23rd ACM SIGKDD Int. Conf. Knowl. Disc. Data Min.*, 2017, pp. 385–394.

[31] E. A. Leicht, P. Holme, and M. E. Newman, "Vertex similarity in networks," *Phys. Rev. E, Stat. Phys. Plasmas Fluids Relat. Interdiscip. Top.*, vol. 73, no. 2, 2006, Art. no. 026120.

[32] F. Lorrain and H. C. White, "Structural equivalence of individuals in social networks," *J. Math. Sociol.*, vol. 1, no. 1, pp. 49–80, 1971.

[33] P. W. Holland and S. Leinhardt, "An exponential family of probability distributions for directed graphs," *J. Amer. Stat. Assoc.*, vol. 76, no. 373, pp. 33–50, 1981.

[34] D. R. White and K. P. Reitz, "Graph and semigroup homomorphisms on networks of relations," *Soc. Netw.*, vol. 5, no. 2, pp. 193–234, 1983.

[35] R. Jin, V. E. Lee, and H. Hong, "Axiomatic ranking of network role similarity," in *Proc. 17th ACM SIGKDD Int. Conf. Knowl. Disc. Data Min.*, 2011, pp. 922–930.

[36] K. Nowicki and T. A. B. Snijders, "Estimation and prediction for stochastic blockstructures," *J. Amer. Stat. Assoc.*, vol. 96, no. 455, pp. 1077–1087, 2001.

[37] R. A. Rossi, D. Jin, S. Kim, N. K. Ahmed, D. Koutra, and J. B. Lee, "On proximity and structural role-based embeddings in networks: Misconceptions, techniques, and applications," *Trans. Knowl. Disc. Data*, vol. 14, no. 5, p. 36, 2020.

[38] S. Gilpin, T. Eliassi-Rad, and I. Davidson, "Guided learning for role discovery (GLRD) framework, algorithms, and applications," in *Proc. 19th ACM SIGKDD Int. Conf. Knowl. Disc. Data Min.*, 2013, pp. 113–121.

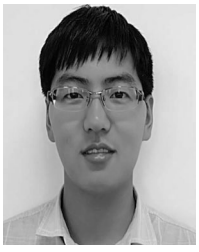
[39] T. Chen, L.-A. Tang, Y. Sun, Z. Chen, H. Chen, and G. Jiang, "Integrating community and role detection in information networks," in *Proc. SIAM Int. Conf. Data Min.*, 2016, pp. 72–80.

[40] Y. Pei, G. Fletcher, and M. Pechenizkiy, "Joint role and community detection in networks via L2, 1 norm regularized nonnegative matrix tri-factorization," in *Proc. IEEE/ACM Int. Conf. Adv. Social Netw. Anal. Min.*, 2019, pp. 168–175.

[41] Y. Pei, X. Du, J. Zhang, G. Fletcher, and M. Pechenizkiy, "Struc2gauss: Structural role preserving network embedding via Gaussian embedding," *Data Min. Knowl. Disc.*, vol. 34, no. 4, pp. 1072–1103, 2020.

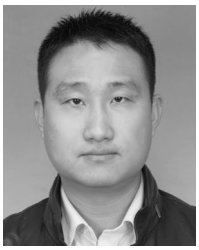
[42] N. Ahmed *et al.*, "Role-based graph embeddings," *IEEE Trans. Knowl. Data Eng.*, early access, Jul. 2, 2020, doi: 10.1109/TKDE.2020.3006475.

[43] C. Donnat, M. Zitnik, D. Hallac, and J. Leskovec, "Learning structural node embeddings via diffusion wavelets," in *Proc. 24th ACM SIGKDD Int. Conf. Knowl. Disc. Data Min.*, 2018, pp. 1320–1329.
[44] G. Nikolentzos and M. Vazirgiannis, "Learning structural node representations using graph kernels," *IEEE Trans. Knowl. Data Eng.*, vol. 33, no. 5, pp. 2045–2056, May 2021.
[45] K. Wu, J. Liu, P. Liu, and S. Yang, "Time series prediction using sparse autoencoder and high-order fuzzy cognitive maps," *IEEE Trans. Fuzzy Syst.*, vol. 28, no. 12, pp. 3110–3121, Dec. 2020.
[46] Y. Yang, H. Chen, and J. Shao, "Triplet enhanced autoencoder: Model-free discriminative network embedding," in *Proc. 28th Int. Joint Conf. Artif. Intell.*, 2019, pp. 5363–5369.
[47] L. Sang, M. Xu, S. Qian, and X. Wu, "AAANE: Attention-based adversarial autoencoder for multi-scale network embedding," in *Proc. Pac.-Asia Conf. Knowl. Disc. Data Min.*, 2019, pp. 3–14.
[48] Z. Chen, C. Chen, Z. Zhang, Z. Zheng, and Q. Zou, "Variational graph embedding and clustering with Laplacian eigenmaps," in *Proc. 28th Int. Joint Conf. Artif. Intell.*, 2019, pp. 2144–2150.
[49] Q. Dai, Q. Li, J. Tang, and D. Wang, "Adversarial network embedding," in *Proc. 32nd AAAI Conf. Artif. Intell.*, 2018, pp. 2167–2174.
[50] S. Pan, R. Hu, G. Long, J. Jiang, L. Yao, and C. Zhang, "Adversarially regularized graph autoencoder for graph embedding," in *Proc. 27th Int. Joint Conf. Artif. Intell.*, 2018, pp. 2609–2615.
[51] 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.
[52] G. Salha, R. Hennequin, V. A. Tran, and M. Vazirgiannis, "A degeneracy framework for scalable graph autoencoders," in *Proc. 28th Int. Joint Conf. Artif. Intell.*, 2019, pp. 3353–3359.
[53] A. Vaswani *et al.*, "Attention is all you need," in *Proc. 31st Int. Conf. Neural Inf. Process. Syst.*, 2017, pp. 6000–6010.
[54] M. E. Newman, "Finding community structure in networks using the eigenvectors of matrices," *Phys. Rev. E, Stat. Phys. Plasmas Fluids Relat. Interdiscip. Top.*, vol. 74, no. 3, 2006, Sep. 036104.
[55] R. A. Rossi and N. K. Ahmed, "The network data repository with interactive graph analytics and visualization," in *Proc. 29th AAAI Conf. Artif. Intell.*, 2015, pp. 4292–4293.
[56] Z. Zhou, *Machine Learning*. Beijing, China: Tsinghua Univ. Press, 2016.
[57] Y. Jin, G. Song, and C. Shi, "GralSP: Graph neural networks with local structural patterns," in *Proc. AAAI*, 2020, pp. 4361–4368.
[58] Q. Long, Y. Jin, G. Song, Y. Li, and W. Lin, "Graph structural-topic neural network," in *Proc. 26th ACM SIGKDD Int. Conf. Knowl. Disc. Data Min.*, 2020, pp. 1065–1073.
[59] W. Zhang, X. Guo, W. Wang, Q. Tian, L. Pan, and P. Jiao, "Role-based network embedding via structural features reconstruction with degree-regularized constraint," *Knowl. Based Syst.*, vol. 218, Apr. 2021, Art. no. 106872.
[60] J. Zhang, F. Hu, L. Li, X. Xu, Z. Yang, and Y. Chen, "An adaptive mechanism to achieve learning rate dynamically," *Neural Comput. Appl.*, vol. 31, pp. 6685–6698, Apr. 2018.



Pengfei Jiao received the Ph.D. degree in computer science from Tianjin University, Tianjin, China, in 2018.

He is a Lecturer with the Center of Biosafety Research and Strategy, Tianjin University. His current research interests include complex network analysis, data mining and graph neural networks, and is currently working on temporal community detection, link predication, network embedding, recommender systems, and applications of statistical network model.



Qiang Tian received the B.S. degree from Tianjin Normal University, Tianjin, China, in 2005, and the M.S. degree from Tianjin University, Tianjin, China, in 2014, where he is currently pursuing the Ph.D. degree with the School of College of Intelligence and Computing.

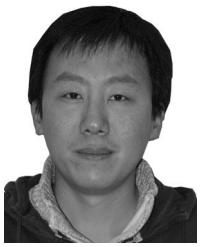
His research interests include dynamic complex network analysis, large-scale data mining, and machine learning.

Wang Zhang received the bachelor's degree from Tianjin University, Tianjin, China, in 2018, where he is currently pursuing the master's degree with the School of Computer Science and Technology.

His current research interests include complex network analysis and network embedding.

Xuan Guo received the bachelor's degree in computer science from Tianjin University, Tianjin, China, in 2018, where he is currently pursuing the master's degree with the School of Computer Science and Technology.

His current research interests include complex network analysis, network representation learning, graph neural networks, and role discovery.



Di Jin received the Ph.D. degree in computer science from Jilin University, Changchun, China, in 2012.

He was a Postdoctoral Research Fellow with the School of Design, Engineering, and Computing, Bournemouth University, Poole, U.K., from 2013 to 2014. He is currently an Associate Professor with the College of Intelligence and Computing, Tianjin University, Tianjin, China. He has published more than 50 papers in international journals and conferences in the areas of community detection, social network analysis, and machine learning.

Dr. Jin serves as an Invited Reviewer for journals, including IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING, and senior program committees for conferences, including AAAI.

Huaming Wu (Member, IEEE) received the B.E. and M.S. degrees in electrical engineering from the Harbin Institute of Technology, Harbin, China, in 2009 and 2011, respectively, and the Ph.D. degree (Highest Hons.) in computer science from Freie Universität Berlin, Berlin, Germany, in 2015.

He is currently an Associate Professor with the Center for Applied Mathematics, Tianjin University, Tianjin, China. His research interests include wireless networks, mobile-edge computing, Internet of Things, and complex networks.