Role-based Multiplex Network Embedding

Hegui Zhang [1] **Gang Kou** [1]

Abstract

In recent years, multiplex network embedding has received great attention from researchers. However, existing multiplex network embedding methods neglect structural role information, which can be used to determine the structural similarity between nodes. To overcome this shortcoming, this work proposes a simple, effective, role-based embedding method for multiplex networks, called RMNE. The RMNE uses the structural role information of nodes to preserve the structural similarity between nodes in the entire multiplex network. Specifically, a role-modified random walk is designed to generate node sequences of each node, which can capture both the within-layer neighbors, structural role members, and cross-layer structural role members of a node. Additionally, the variant of RMNE extends the existing collaborative embedding method by unifying the structural role information into our method to obtain the role-based node representations. Finally, the proposed methods are evaluated on the network reconstruction, node classification, link prediction, and multi-class edge classification tasks. The experimental results on eight public, real-world multiplex networks demonstrate that the proposed methods outperform state-of-the-art baseline methods.

1. Introduction

The wide applicability of networks and their success in describing real-world complex systems, for example social (Zhang et al., 2022), traffic(Dapeng & Xiao, 2021), biological (Seninge et al., 2021), and financial (Acemoglu et al., 2015; Zha et al., 2020; Guo et al., 2021; Stolbov et al., 2021) systems, have thus garnered considerable research attention in many fields. Motivated by the demands of network analysis tasks, such as node classification, link prediction, node clustering, and community detection, many network embedding methods (Cui et al., 2018) have been proposed to learn low-dimensional vector representations of nodes. Unsupervised network representation learning embeds a node representation vector without any external supervision labels. The basic strategy to extract node embedding is leveraging the node proximity or structural similarity in the network structure to learn dense low dimension representations. For instance, DeepWalk (Perozzi et al., 2014) and Node2vec (Grover & Leskovec, 2016) capture a node's proximity by maximizing the probability of its neighbors generated by a random walk. Role2vec (Ahmed et al., 2020) learns role-based embeddings that capture structural roles based on an attributed random walk. However, most existing popular methods predominately focus on the single-layer networks (Khosla et al., 2019), but real-world systems are highly complex; so, using multiplex networks to model these systems tends to be more realistic (Mucha et al., 2010; Osat et al., 2017).

Most real networks have the nature of multiplicity, and there are typically different types of connectivity between two entities. For instance, in a social network, users are usually connected through different relationships, such as friends, colleagues, or relatives. Recent research has demonstrated that ignoring the multiplicity of multiplex networks can be potentially dangerous, leading to severe consequences in system analysis (Osat et al., 2017). Therefore, multiplex (also known as multi-layer, multi-view or multi-relation) networks are more suitable for modeling complex systems, in which layers share the same set of individuals but may have different connectivity between two individuals in different layers (a toy example is shown in Figure 1).

Compared with a single-layer network, the embedding of a multiplex network is more challenging due to several reasons. First, the embedding method should be able to capture the structural information of the whole multiplex network rather than a single network, that is, the embedding method should make full use of the structural information of each layer of the multiplex network to improve the embedding quality. Second, the embedding method should be able to capture node proximity and structural similarity both in within- and cross-layer nodes. The existing research has denoted that the concept of structural roles is very suitable

[1]School of Business Administration, Faculty of Business Administration, Southwestern University of Finance and Economics, Chengdu 611130, China. Correspondence to: Gang Kou <kougang@swufe.edu.cn;kougang@yahoo.com>.

Proceedings of the 39^{th} International Conference on Machine Learning, Baltimore, Maryland, USA, PMLR 162, 2022.

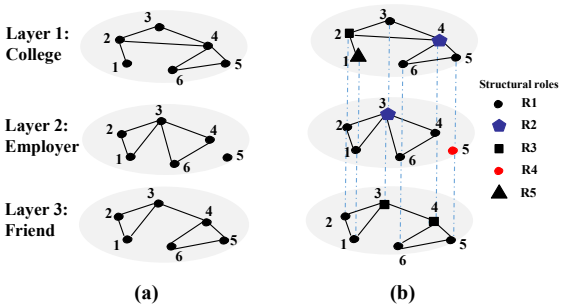


Figure 1. A toy example of a three-layer undirected multiplex network; and the simplest example of structural roles in a multiplex network. Here, when two nodes have the same degree value, they belong to the same structural role.

to denote the structural similarity between nodes (Rossi & Ahmed, 2014). By definition, two nodes belong to the same structural role if they are structurally similar, which allows them to be neither directly connected nor in the same layer network (Ahmed et al., 2020; Rossi & Ahmed, 2014; Zhang et al., 2021). As shown in Figure 1, all black squares belong to the same structural role (R3) since they have the same degree value, and thus, their vector representations should be close in a unified embedding space.

Recently, representation learning algorithms based on the single-layer network embedding methods have been proposed to learn the representation of multiplex network nodes. For instance, certain methods extend the random walk to entire multiplex network to generate the walks for embedding (Ning et al., 2018; Wilson et al., 2021). Other methods learn a base layer vector and combine it with the learned node vector into the final node vector representation (Matsuno & Murata, 2018; Zhang et al., 2018; Chu et al., 2019). The graph neural networks have also been applied to learn multiplex network embeddings (Berger-Wolf & Chawla, 2019). However, none of the above-mentioned methods can capture the structural role information in a multiplex network. Recently, several role-based embedding methods of single-layer networks have been proposed to capture structural similarity (Ahmed et al., 2020; Ma et al., 2019; Javari et al., 2020; Zhang et al., 2021). These methods have shown that the structural role information of nodes is beneficial to improve the node embeddings quality of the single-layer networks.

In this work, a simple, effective network embedding method, named the RMNE, is proposed to learn the role-based node embedding for multiplex networks. Additionally, in order to unify the different characteristics of a multiplex network into one framework, the RMNE$^+$ is further developed. The main contributions of this work are summarized as follows:

- A simple but effective role-based network embedding method, called RMNE, is proposed for multiplex networks. The learned embeddings of the RMNE can preserve the structural similarity both in within- and cross-layer nodes. Unlike the previously proposed methods, our methods derive the role-based embedding, which can capture the structural role information on nodes.
- A role-modified random walk is designed to sample the neighborhood of each node in a multiplex network. This method can sample across layers and capture the rich connection information on nodes. In addition, it can sample the structural role members of a node within and between network layers.
- Further, the RMNE$^+$ is developed to unify the collaborative and structural role characteristics into one framework.
- The proposed methods are empirically evaluated on the network reconstruction, node classification, link prediction, and multi-class edge classification tasks using several real-world multiplex networks. The experimental results demonstrate the superiority of our methods over the baseline methods.

2. Proposed methods

In this section, the concept of the multiplex network and the definition of the structural roles of multiplex networks are first presented. Then, we introduce the architectures of the proposed methods, followed by the details of each part.

2.1. Notations

This work focuses on multiplex networks, which are defined by Definition 2.1.

Definition 2.1. An L-layer multiplex network represents a collection of L networks $\mathcal{G} = \{G_1, \ldots, G_L\}$, where $G_l = (V_l, E_l)$ denotes the l-th layer in $\mathcal{G}$. V_l denotes the set of nodes of the l-th layer, and $E_l \in V_l \times V_l$ denotes the set of edges. For nodes $u, v \in V_l$, $e_l(u, v) \in E_l$ represents the edge from node u to node v; $w_l(u, v)$ denotes the weight of edge $e_l(u, v)$in the l-th layer. There is a one-to-one correspondence between the nodes in different layers, but links of different layers may differ from each other. Denoting the set of unique nodes in the multiplex network $\mathcal{G} = \{G_1, \ldots, G_L\}$ by $\mathcal{N}$; then, $N = |\mathcal{N}|$ is the number of nodes of $\mathcal{G}$. Thus, a multiplex network with L layers and node set $\mathcal{N}$ can be expressed as $\mathcal{G}(\mathcal{N}, \mathcal{E}, L)$, where $\mathcal{E}$ denotes the set of edges for the multiplex network.

The structural role is defined by Definition 2.2.

Definition 2.2. Structural roles define sets of nodes; the nodes inside the same set have higher structural similarity

than those from different sets (Rossi et al., 2020). The structurally similar indicates that nodes have similar structural properties.

The simplest example of structural similarity is that two nodes have the same degree; so, they are structurally similar, as shown in Figure 1. This definition allows nodes to be neither directly connected nor even in the same layer network.

2.2. The RMNE Algorithm

The overall framework of the proposed RMNE is shown in Figure 2, where it can be seen that the RMNE includes three parts: the role discovery process, which divides the nodes of the input multiplex network into classes of structurally similar nodes; role-modified random walk, which generates the node sequence as the training samples of SkipGram model (Mikolov et al., 2013) for each node $u \in \mathcal{N}$; the Skip-Gram model, which is used to obtain the node's d-dimensional vector representations. The pseudo-code of the RMNE method is presented in Algorithm 1.

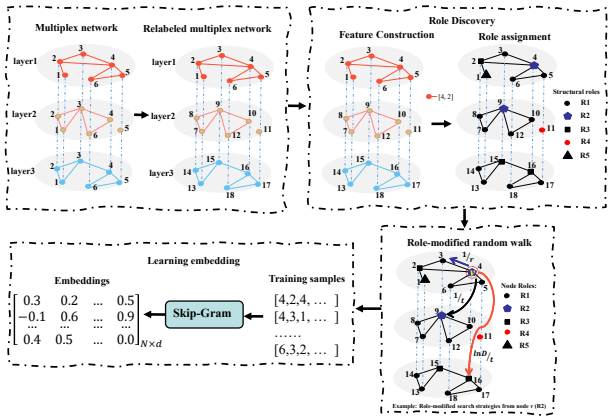


Figure 2. An overview of the proposed RMNE method: (1) the role discovery process; (2) role-modified random walk; (3) generate embeddings for all nodes.

Specifically, Line 3 relabels the nodes of a multiplex network, as shown in Figure 2; Line 4 performs the role discovery process and divides the nodes into different role classes. The details of role discovery will be introduced later. Lines 5 and 6 preprocess the transition probability for the role-modified random walk. Lines 7-14 generate the node sequence by using the role-modified random walk for all nodes; the process will also be described later. Line 15 learns the embeddings for all nodes in the multiplex network, as explained below.

Algorithm 1 The RMNE method

1: **Input:** Multiplex network $\mathcal{G}(\mathcal{N}, \mathcal{E}, L)$, number of random walks per node K, random walk maximum length Le, the window size of the SkipGram W, the representation size dimension d, random variable selected from neighbors r, random variable selected from the same role members t.
2: **Output:** Node representation matrix $f \in R^{(|\mathcal{N}| \times d)}$.
3: $\mathcal{G}_r(\mathcal{N}_r, \mathcal{E}_r, L)$ = Relabeled $\mathcal{G}(\mathcal{N}, \mathcal{E}, L)$
4: RoleList = RoleDiscovery $(\mathcal{G}_r)$
5: π = PropocessModifiedWeights $(G_r, RoleList, r, t)$
6: $\mathcal{G}_r^{'} = \mathcal{G}_r(\mathcal{N}_r, \mathcal{E}_r, L, \pi)$
7: Initialize $walks$ to Empty.
8: **while** $(i < K)$: **do**
9: $S = shuffle(\mathcal{N}_r)$
10: **for** all nodes $u \in \mathcal{N}_r$ **do**
11: $walk$ = role-modified random walk $(\mathcal{G}_r^{'}, u, Le)$
12: append $walk$ to $walks$
13: **end for**
14: **end while**
15: $SkipGram(W, d, walks)$

2.2.1. Role Discovery in RMNE

The role discovery aims to group the network nodes into classes of structurally similar nodes (Rossi & Ahmed, 2014). In this study, the role discovery process is performed over a multiplex network, that is, nodes may differ in the structural role between different layers since their structural properties may be different in different layers. Therefore, the network nodes are relabeled before the role discovery process so that the same node has different IDs in different layers. The relabeled multiplex network is denoted as $\mathcal{G}_r(\mathcal{N}_r, \mathcal{E}_r, L)$. As shown in Figure 2, node 4 (role: R2) of layer 1, node 10 (role: R1) of layer 2, and node 16 (role: R3) of layer 3 denote the same individual.

The role discovery methods have been reviewed in(Rossi & Ahmed, 2014). The Role2vec(Ahmed et al., 2020) includes three role discovery methods, namely, the motif-count-based, degree-based, and Weisfeiler-Lehman (WL) feature extraction methods (Shervashidze & Borgwardt, 2009; Ma et al., 2019). For the RMNE method, the role discovery process is performed based on the Weisfeiler-Lehman feature extraction method. The Weisfeiler-Lehman method decomposes each layer of a multiplex network into a tree-like pattern (called 'subtree'). Specifically, it first relabels each node with a compressed multi-label that consists of the original label of the node and the labels of its neighbors, and then, the feature vectors of nodes are extracted by repeating this process multiple times. The labeling iterations of the WL method are usually set to two, to capture node-local structural patterns. If the method uses the degree of a node as its feature, we can bin the degrees of all nodes so that

nodes belonging to the same bin belong to the same role. Finally, the structural role of each node is obtained through the role assignment process(Rossi & Ahmed, 2014).

After completing role feature construction (degree-based, motif-count-based, and WL method in this paper), role assignment aims to assign nodes with similar features to the same roles. There are several options for this process, the first is to divide the nodes into k types of structural roles through a clustering method such as k-means. Additionally, roles can be assigned through a simple equivalence rule (i.e.two nodes belong to the same structural role if their feature vectors are the same). In this work, we use the simple equivalence rule.

2.2.2. Role-Modified Random Walk

Random walk is at the core of many network embedding methods. However, a multiplex network poses novel challenges to the traditional random walk. First, a random walk should be able to sample across layers to capture the rich connection information on nodes. Second, a random walk should be able to sample the structural role members of nodes within and between layers, so that the embedding can preserve the structural similarity. In this work, a role-modified random walk is proposed to handle the two mentioned challenges.

Consider a random walk maximum length Le and a source node $u \in \mathcal{N}_r$. If the algorithm is at a node $u_{(i-1)}$ of a layer l at time $(i-1)$, it moves with a certain probability to the next node u_i of a layer $l^{'}$ at the time i. The node u_i is a neighbor or the structural role member of the node $u_{(i-1)}$. Formally, the u_i denotes the i-th node in the walk, and it is obtained as follows:

$$\begin{aligned} &\mathbb{P}\left(u_i = x, l_i = l' \mid u_{i-1} = v, l_{i-1} = l\right) \\ &= \begin{cases} \pi_{v,x,l,l'}/z \text{ if } (v,x) \in \mathcal{E}_r \text{ or } \mathrm{R}(x) = \mathrm{R}(v) \\ 0 \quad \text{otherwise} \end{cases} \end{aligned} \tag{1}$$

where $\pi_{v,x,l,l'}$ represents the transition probability between nodes v and x, Z is the normalization constant, and R denotes node's role. In this work, two hyperparameters denoted as r and t are designed to guide the sampling strategy of the role-modified random walk. Thus, the transition probability is expressed as $\pi_{x,v,l,l'} = a_{tr}\left(v,x,l,l'\right).w_{l,l'}(v,x)$, where the $w_{l,l'}(v,x)$ denotes the edge weight between nodes v and x, and it is defined as follows:

$$w_{l,l'}(v,x) = \begin{cases} w_{l,l'}(v,x) & \text{if } l = l', \\ 1 & \text{if } l \neq l'. \end{cases} \tag{2}$$

For an unweighted network, $w_{(l,l')}(v,x) = 1$. In addition,

Algorithm 2 The role-modified random walk method

1: **Input:** $\mathcal{G}_r^{'}$, start node u, random walk maximum length Le.
2: **Output:** A node list with the maximum length of Le.
3: Initialize $walk$ with node u.
4: **while** (len $(walk) < Le$) **do**
5: cur = $walk[-1]$
6: $\mathcal{T}_{cur}$ = set of the neighbors and role members for node cur
7: $w = AliasSample(\mathcal{T}_{cur}, \pi)$
8: Append w to $walk$
9: **end while**
10: r_walk = recover($walk$)

$a_{tr}(v,x,l,l^{'})$ is defined as follows:

$$\begin{aligned} &a_{tr}\left(v,x,l,l'\right) \\ &= \begin{cases} \delta_r(v,x) & \text{if } l = l' \text{ and } e_{l,l'}(v,x) \in \varepsilon_r, \\ \beta_t(v,x) & \text{if } \mathrm{R}(v) = \mathrm{R}(x) \text{ and } v \neq x, \\ \beta_t(v,x) * \ln(D(v)) & \text{if } l = l' \text{ and } v = x, \\ \beta_t(v,x) * \ln(D(x)) & \text{if } l \neq l' \text{ and } v = x, \end{cases} \end{aligned} \tag{3}$$

where $\delta_r(v,x)$ and $\beta_t(v,x)$ are respectively defined by:

$$\delta_r(v,x) = \frac{1}{r} \tag{4}$$

$$\beta_t(v,x) = \frac{1}{t} \tag{5}$$

where, $v = x$ indicates that nodes v and x denote the same individual; for instance, nodes 4 and 16 in Figure 2. As given by Eq. (3), the hyperparameter r guides the sampling of directly connected neighbors, while the hyperparameter t controls the sampling of structural role members within and between layers; if $v = x$, the node degree at different layers is used to distinguish the node importance in these layers. Thus, if $u_{(i-1)} = v$, the role-modified random walk samples a node v itself with the probability of $\beta_t(v,x) * \ln(D(v))$at time i, and samples a one-to-one corresponding node (node x) of v in other layers with the probability of $\beta_t(v,x) * \ln(D(v))$, where $D(\cdot)$ denotes the node degree, and $\ln(\cdot)$ is the natural logarithm function, which is used here to avoid the deviation caused by a large node degree value. If the degree of a node is zero, we set it equal to 1.

After calculating the random walk transition probability by using Eqs. (1)-(5), the RMNE performs the role-modified random walk algorithm, the pseudo-code of which is given in Algorithm 2. Particularly, line 10 performs the recovering of the relabeled nodes ID. For instance, a $walk = [4,3,13,18]$ is recovered as $r\_walk = [4,3,1,6]$.

2.2.3. LEARN EMBEDDINGS

According to Algorithm 1, after generating of the role-modified random walks in a multiplex network, the Skip-Gram architecture is used to learn embeddings for all nodes. Specifically, for a node $u \in \mathcal{N}$, $N_r(u)$ is defined as a neighborhood of node u generated by the role-modified sampling strategy. The objective function is given by:

$$\max \prod_{u \in \mathcal{N}} \mathbb{P}\left(N_r(u) \mid f(u)\right) \tag{6}$$

where $f(\cdot)$ denotes the vector representation. In this work, two standard assumptions are adopted (Grover & Leskovec, 2016). The first assumption corresponds to the conditional independence assumption, whereas the second assumption characterizes the symmetric effect of neighboring nodes in their feature space. Accordingly, the softmax function can be used to approximate $\mathbb{P}(v|f(u))$ as follows:

$$\mathbb{P}(v \mid f(u)) = \frac{\exp(f(v) \cdot f(u))}{\sum_{w \in \mathcal{N}} \exp(f(w) \cdot f(u))} \tag{7}$$

Thus, the objective function of Eq. (6) can be simplified to:

$$\min - \prod_{u \in \mathcal{N}} \prod_{v \in N_r(u)} \frac{\exp(f(v) \cdot f(u))}{\sum_{w \in \mathcal{N}} \exp(f(w) \cdot f(u))} \tag{8}$$

Next, Eq. (8) can be rewritten in the log-likelihood form as follows:

$$\min - \sum_{u \in \mathcal{N}} \sum_{v \in N_r(u)} \left[-\log Z_u + f(v) \cdot f(u)\right], \tag{9}$$

where $Z_u = \sum_{w \in \mathcal{N}} \exp(f(w) \cdot f(u))$, and negative sampling is used to approximate its value (Mikolov et al., 2013).

Finally, Eq. (9) is optimized by the stochastic gradient descent (SGD) algorithm (Bottou et al., 1991). The optimization result represents a d-dimensional vector representation of all nodes in the multiplex network.

2.3. The RMNE$^+$ Algorithm

The overall framework of the RMNE$^+$ is shown in Figure 3, which unifies the structural role information based on the existing MANE (Ata et al., 2021) method. The key view of the MANE method is that "if two nodes are associated in one layer, they are more likely to be connected in another layer as well", which it call second-order collaboration. Apparently, the method does not take into account the structural similarity between nodes but is only based on 'connectivity'.

Recall the Skip-gram method, a node pair (u, v) consists of the center node u and a context node v, the task is to maximize the co-occurrence probability of node pairs. Given an

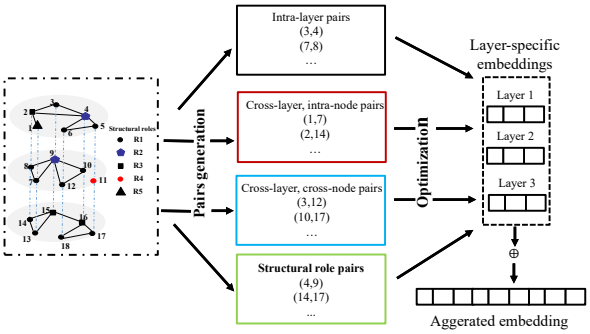


Figure 3. An overview of the RMNE$^+$ method, which unify the intra-layer, cross-layer, and structural role node pairs into one framework based on (Ata et al., 2021).

L-layer multiplex network and the generated node pairs, the corresponding loss of each characteristic in the RMNE$^+$ is shown as follows:
Intra-layer pairs:

$$L_0 = -\sum_{l \in L} \sum_{u \in \mathcal{N}} \sum_{v \in Ne(u)} \log \left(\frac{\exp\left(f_v^T f_u\right)}{\sum_{w \in \mathcal{N}} \exp\left(f_w^T f_u\right)} \right) \tag{10}$$

For each layer, the intra-layer pairs are generated from random walks following the Deepwalk method. The $Ne(u)$ is the context node set of node u under a specific window size, and center node u and node $v \in Ne(u)$ form the intra-layer node pair (u,v), such as node pairs (3,4) and (7,8) in Figure 3.

Cross-layer, intra-node pairs:

$$L_1 = -\sum_{l \in L} \sum_{u \in \mathcal{N}} \sum_{v \in Intra(u)} \log \left(\frac{\exp\left(f_v^T f_u\right)}{\sum_{w \in \mathcal{N}} \exp\left(f_w^T f_u\right)} \right) \tag{11}$$

The cross-layer, intra-node pairs are generated across different layers for the same node. The $Intra(u)$ denotes the one-to-one corresponding nodes set for node u. For example, as shown in Figure 3, $Intra(1) = \{7, 13\}$.

Cross-layer, cross-node pairs:

$$L_2 = -\sum_{l \in L} \sum_{u \in \mathcal{N}} \sum_{v \in Cross(u)} \log \left(\frac{\exp\left(f_v^T f_u\right)}{\sum_{w \in \mathcal{N}} \exp\left(f_w^T f_u\right)} \right) \tag{12}$$

The $Cross(u)$ is the second-order collaboration node set of the center node u. As shown in Figure 3, nodes 9 and 12 are connected at the second layer of the multiplex network and are therefore more likely connected at the first layer (i.e. nodes 3 and 6 are more likely to be connected), thus forming the cross-layer, cross-node pair (3,12).

Structural role pairs:

$$L_3 = -\sum_{l\in L}\sum_{u\in\mathcal{N}}\sum_{v\in Role(u)}\log\left(\frac{\exp\left(f_v^T f_u\right)}{\sum_{w\in\mathcal{N}}\exp\left(f_w^T f_u\right)}\right) \tag{13}$$

The $Role(u)$ denotes the structural role members of node u, and two nodes with the same role form a structural role pair (such as node pair (4,9) in Figure 3).

Hence, the overall loss for a multiplex network is obtained by combining Eq.(10)-(13):

$$L = L_0 + \alpha L_1 + \beta L_2 + \gamma L_3 \tag{14}$$

Where α, β, and $\gamma(\geqslant 0)$are hyperparameters that control the contribution of each component to the overall loss. When $\alpha = \beta = \gamma = 0$, the RMNE$^+$ is reduced to Skip-gram models on individual layers of a multiplex network.

To implement the RMNE$^+$, we follow the MANE to generate the collaborative node pairs, and the structural role pairs are obtained by combining pairs of nodes in the role class set. The negative sampling and the Adam optimizer are also adopted. A final node representation is obtained by concatenating the layer-specific embeddings.

3. Experiments

The proposed methods were evaluated on the network reconstruction, node classification, link prediction, and multi-class edge classification tasks. The network reconstruction problem was defined as follows. Given an L-layer multiplex network $\mathcal{G}(\mathcal{N}, \mathcal{E}, L)$ and the node embeddings learned from the $L-1$ layers of $\mathcal{G}$, the network reconstruction task aimed to predict the edges and non-edges of the entire L-th layer.

The network reconstruction task is a supervised learning problem. In the experiment, the test samples were constructed as follows. The edges set off the L-th layer were the positive samples; an equal number of randomly generated non-edges were the negative samples. It was ensured that the non-edges in the negative samples did not belong to the edges of the input multiplex network.

Further, an edge (u, v) embedding was constructed based on the node embeddings of nodes u and v using different operators, including Hadamard, Average, Weighted-L1, and Weighted-L2 (as shown in Appendix B). Then, a logistic regressor was trained to conduct the prediction task. The ROC-AUC was used as an evaluation metric of model performance and it is calculated by averaging the results of 10 runs.

The node classification, link prediction, and multi-class edge classification tasks are performed the same as the (Ata et al., 2021). In addition, the effect of different parameters of the proposed methods' performance was analyzed.

Table 1. A brief introduction to the networks used in this work.

DATASET	#LAYERS	#NODES	#EDGES
CKM	3	246	1551
LLF	3	71	2571
VICKERS	3	29	740
KTS	4	39	1018
SACCHCERE	4	6570	97482
ALZHEIMER'S	2	12901	204353
LINKEDIN	3	10196	3323042
YOUTUBE	3	7558	4469222

3.1. Datasets

Eight real-world multiplex networks were used to validate the performance of the proposed methods. The basic statistics for the multiplex networks used are given in Table 1.

The **CKM** (Coleman et al., 1957) physician-innovative multiplex network contains three different layers. The **LLF** (Snijders et al., 2006) multiplex social network consists of three types of relationships (co-work, friendship, and advice) between the partners and associates of a corporate law partnership. The **Vickers** (Vickers & Chan, 1981) denotes a three-layer multiplex social network collected by Vickers from 29 seventh-grade students in a school in Victoria, Australia. The **KTS** (Kapferer, 1972) describes the interactions in a tailor shop in Zambia over a period of 10 months. The **Sacchcere** (De Domenico et al., 2015) represents a multiplex genetic and protein interaction network of the Saccharomyces Cerevisiae. In this study, the physical association, direct interaction, association, and colocalization layers were extracted. The **Alzheimer's** (Ata et al., 2021) is a two-layers protein network and we conduct the node classification task on this labeled network. The **LinkedIn** (Ata et al., 2021) captures three different relationships among a social network,we perform the multi-class edge classification on this network. The **YouTube** (Ata et al., 2021) is a three-layers video-sharing network. We conduct the link prediction task on it.

3.2. Baseline Methods

For the RMNE method, the comparison methods include **DeepWalk** (Perozzi et al., 2014), **Node2vec** (Grover & Leskovec, 2016), **Ohmnet** (Zitnik & Leskovec, 2017), **PMNE** (Liu et al., 2017), **MNE** (Zhang et al., 2018),**MANE** (Ata et al., 2021), **FFME** (Ning et al., 2019), and **GATNE-T** (Cen et al., 2019) methods. For RMNE$^+$ method, the baseline methods are the same as the MANE (Ata et al., 2021), which include **HIN2Vec** (Fu et al., 2017) and **HeGAN** (Hu et al., 2019) two heterogeneous network embedding methods, and five multiplex network embedding methods, $i.e.$ **MVE** (Qu et al., 2017), **mvn2vec** (Shi et al., 2018), **MNE**

(Zhang et al., 2018), **DMNE** (Ni et al., 2018), and **GATNE** (Cen et al., 2019). Two representative role-based single-layer network embedding methods were also compared, namely **Struc2vec** (Ribeiro et al., 2017) and **Role2vec** (Ahmed et al., 2020). Theses methods are explained briefly in Appendix A.

3.3. Performance Comparison

For the network reconstruction task, the experimental results are shown in Table 2, where it can be seen that the RMNE achieved the optimal result compared to the baseline methods. According to the results in Table 2, the single-layer network embedding method without considering the multiplicity of a multiplex network showed poor performance in the network reconstruction task on most datasets. Moreover, the MANE, a multi-view network embedding method, had a relatively low AUC score compared to the other multiplex network embedding methods. Generally, the RMNE combined with the Weighted-L2 operator outperformed all baseline methods for all datasets, except for the CKM. These results suggest that embedding with preserving the structural role information is beneficial to the network reconstruction task of multiplex networks.

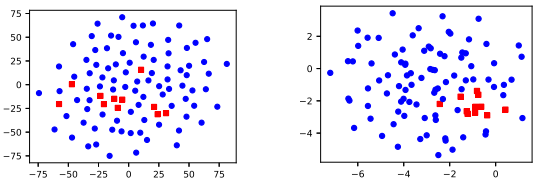
Table 2. AUC scores of different embedding methods combined with different operators in the network reconstruction task on five real-world multiplex networks. The best result for each dataset is denoted in bold; for each operator, the best scores are underlined. See Appendix B for the complete results (Sac: Sacchcere, OP: operators, HD: Hadamard, L2: Weighted-L2).

OP	METHOD	LLF	VICKERS	KTS	SAC	CKM
HD	DEEPWALK	0.5371	0.6321	0.6203	0.7903	0.7508
	NODE2VEC	0.5926	0.6306	0.6159	0.7884	0.7391
	STRUC2VEC	0.6645	0.7596	0.7860	0.7853	0.5918
	ROLE2VEC	0.6353	0.6482	0.6977	0.7574	0.9115
	MANE-1	0.5585	0.5936	0.6653	0.6334	0.6533
	MANE-2	0.6028	0.6009	0.7153	0.5342	0.7245
	OHMNET	0.7320	0.8035	0.8461	0.9112	0.9073
	PMNE(N)	0.7196	0.8524	0.8837	0.8827	0.8925
	PMNE(R)	0.7124	0.8443	0.8456	0.8766	0.8905
	PMNE(C)	0.7050	0.8501	0.8489	0.8927	0.8869
	MNE	0.7043	0.7024	0.7706	0.9138	0.9139
	GATNE-A	0.7563	0.7833	0.8271	0.8967	0.9087
	GATNE-C	0.7525	0.8239	0.8093	0.8997	0.9056
	FFME	0.7458	0.8076	0.7606	0.9121	0.9115
	RMNE	0.7606	0.8246	0.8465	0.8783	**0.9208**
L2	DEEPWALK	0.5485	0.6475	0.6321	0.7012	0.6855
	NODE2VEC	0.5834	0.6195	0.5952	0.7060	0.6855
	STRUC2VEC	0.6509	0.7301	0.7367	0.7865	0.5856
	ROLE2VEC	0.6400	0.6770	0.7074	0.7980	0.8203
	MANE-1	0.5625	0.6062	0.6773	0.8579	0.6527
	MANE-2	0.6021	0.6094	0.7373	0.8614	0.6844
	OHMNET	0.7066	0.7817	0.7963	0.9130	0.8890
	PMNE (N)	0.7322	0.8511	0.8654	0.8907	0.8793
	PMNE (R)	0.7202	0.8519	0.8610	0.8536	0.8654
	PMNE (C)	0.7009	0.8470	0.8606	0.5322	0.8768
	MNE	0.7053	0.7290	0.7424	0.8352	0.9022
	GATNE-A	0.7578	0.8199	0.8171	0.8271	0.9029
	GATNE-C	0.7564	0.8559	0.8406	0.8850	0.9014
	FFME	0.7733	0.8499	0.8289	0.7054	0.8979
	RMNE	**0.8017**	**0.8957**	**0.9109**	**0.9274**	0.9151

For the node classification, link prediction, and multi-class edge classification tasks, the results are shown in Table 3: the RMNE$^+$ obtained the optimal results compared to these baselines. The visualization of embeddings of the Alzheimer's network obtained by RMNE$^+$ and MANE is shown in Figure 4. We only sampled 100 nodes which included all positives for clearer visualization since the Alzheimer's dataset is highly imbalanced with fewer than 1% positive samples. The result denoted that the positives of the RMNE$^+$ method clustered well and were relatively dense. Here, the role discovery method for Alzheimer's dataset is Weisfeiler-Lehman-based, while LinkedIn and YouTube are based on degree and motif-count methods respectively. We further include the results of various role discovery methods in Appendix B. In addition, we also present the role distributions for different role discovery methods in Appendix B. Note that, to make the RMNE$^+$ method suitable for large-scale networks, we only count the degree and the number of triangles as node features for the motif-count method.

Table 3. Performance evaluation for node classification, multi-class edge classification and link prediction tasks, the best result for each dataset is denoted in bold. The results of these baseline methods (except Struc2vec and Role2vec methods) are obtained from (Ata et al., 2021).

	ALZHEIMER'S		LINKEDIN		YOUTUBE	
METRIC	ROC-AUC	PR-AUC	MICRO-F	MACRO-F	ROC-AUC	PR-AUC
SINGLE	0.6968	0.0221	0.4001	0.3468	0.6334	0.1565
DECOUPLED	0.8200	0.0735	0.4341	0.3739	0.6700	0.1649
MERGED	0.7305	0.0075	0.4197	0.3724	0.6565	0.1763
STRUC2VEC	0.5807	0.0010	0.4033	0.3371	0.6517	0.1770
ROLE2VEC	0.9330	0.1759	0.4355	0.3780	0.6398	0.1571
HIN2VEC	0.5734	0.0030	0.3014	0.2175	0.6264	0.1258
HEGAN	0.6967	0.0104	0.3937	0.3467	0.6322	0.1520
MVE	0.6543	0.0161	0.4113	0.2867	0.6276	0.1650
MVN2VEC-C	0.8617	0.0890	0.4326	0.3791	0.6633	0.1621
MVN2VEC-R	0.8756	0.0275	0.4439	0.3717	0.6703	0.1669
MNE	0.9195	0.1676	0.4334	0.3538	0.6749	0.1604
DMNE	0.9357	0.0603	0.3473	0.2253	0.6056	0.1394
GATNE	0.9190	0.1227	0.3202	0.1619	0.6838	0.1624
MANE	0.9660	0.2277	0.4446	0.3865	0.6917	0.2039
RMNE$^+$	**0.9761**	**0.4264**	**0.4602**	**0.3922**	**0.7017**	**0.2194**



(a) MANE (b) RMNE$^+$.

Figure 4. Visualization of Alzheimer's proteins.

3.4. Ablation and Parameter Sensitivity Analysis

For the RMNE method, the parameter sensitivity experiment was performed on various real-world datasets. In the experiment, to evaluate the effect of a particular parameter on the network reconstruction performance, that parameter was changed while the other parameters were kept fixed. The results are shown in Figure 5. Compared with the other parameters, the embedding dimension d and window size W had little effect on the AUC score. For the Sacchcere dataset, all parameters of the RMNE had little effect on the network reconstruction, indicating the RMNE could stably reconstruct the L-layer, regardless of the changes in the parameters' values. For the LLF dataset, the optimal values of Le and K were 15 and 20, respectively. The best values of K and Le for the Vickers dataset were 50 and 30, respectively.

Intuitively, the parameter r controls the likelihood of sampling the within-layer neighbors, while the parameter t guides the sampling of the cross-layer and role-level. If $r < 1$, this sampling strategy encourages the walk to preserve more node's within-layer neighbors. Conversely, if $t < 1$, the walk is more likely to sample a node's role members and encourages exploration across layers. The effects of parameters r and t are shown in Figure 5 (a) and (b), respectively. For the LLF, Vickers, and KTS datasets, the RMNE performed better when $r > 1$, while for datasets Sacchcere and CKM, its performance decreased slightly as r increased. The best values of t for LLF, Vickers, and KTS datasets were 0.5, 0.25, and 1.0, respectively. For datasets Sacchcere and CKM, the RMNE's performance increased slightly as t increased.

For the RMNE$^+$ method, we examined the contribution of structural role pairs in Eq.14. Specifically, we evaluated (1) Without role pairs, $i.e., \gamma = 0$; (2) Only role pairs, $i.e., \alpha = \beta = 0$. As Figure 6 shows, RMNE$^+$ outperforms the other cases, indicating that the structural role information of nodes is beneficial to derive higher-quality node representations. Interestingly, when we only use the structural role pair for optimization, $i.e. \alpha = \beta = 0$ in Eq.14, our method still achieves better performance than MANE (Ata et al., 2021) in datasets YouTube and LinkedIn. While in Alzheimer's network, the ROC-AUC metric has declined due to the large influence of the second-order collaboration (Ata et al., 2021). We further vary γ in Figure 7 (a, b, c), the results denote $\gamma = 0.5$ exhibits the optimal performance. Figure 7 (d) shows the fast convergence of the RMNE$^+$ method, typically less than 8 epochs. The impact of hyperparameters α and β are shown in Appendix B. Particularly, when $\alpha = 0.5$, the RMNE$^+$ achieves the best performance on Alzheimer's dataset, $i.e.$ ROC-AUC: 0.9918, PR-AUC: 0.5192.

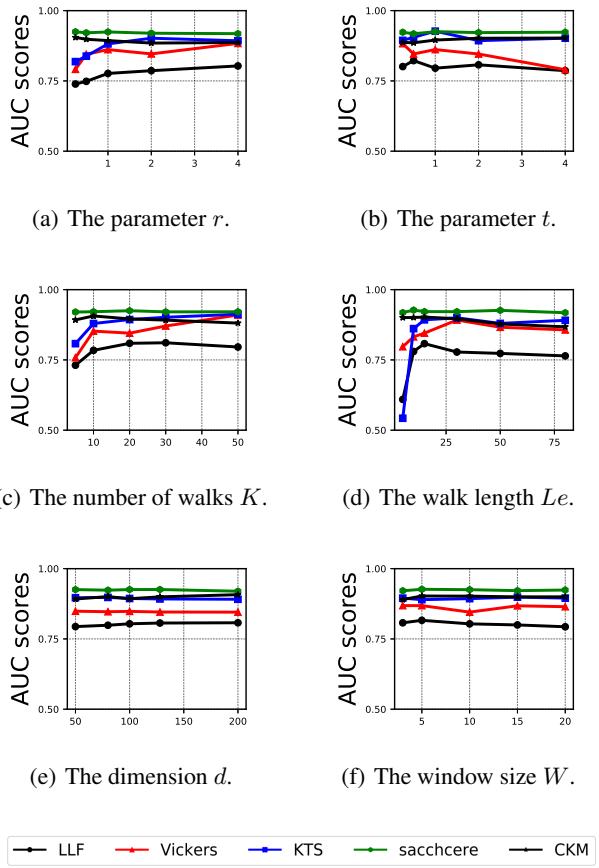
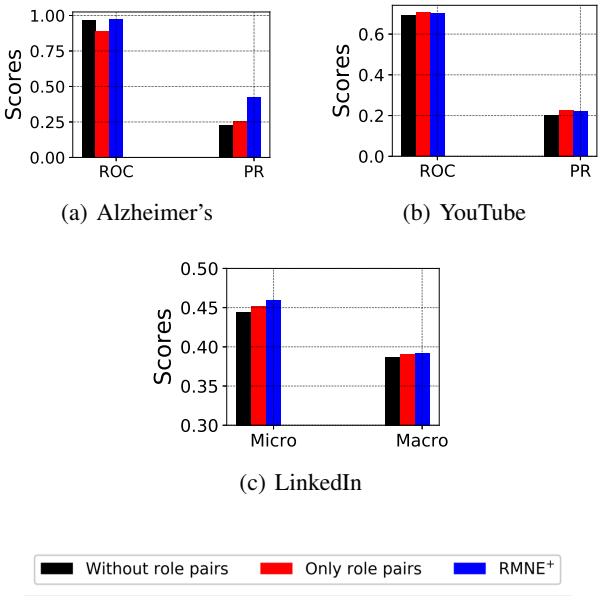


Figure 5. The effects of different parameters on the RMNE performance.

3.5. Scalability Analysis

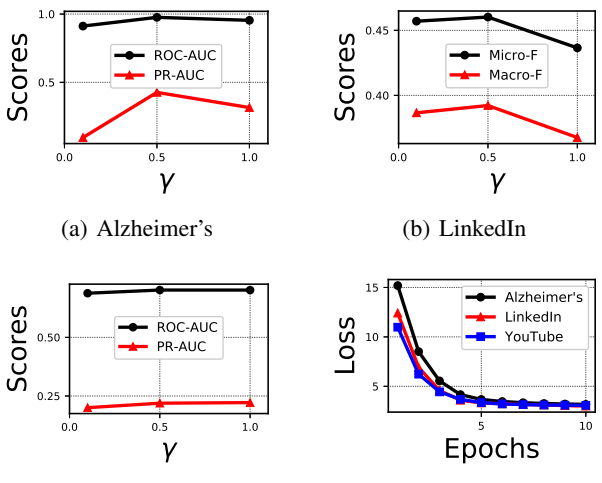
To evaluate the scalability of the proposed RMNE method, the node embeddings of the Barabasi-Albert scale-free networks were learned by increasing the network size from 100 to 100000. For each network size case, a two-layer multiplex network with an average node degree of 10 for each layer was generated. The results are shown in Figure 8. The sampling procedure included the role discovery process and the role-modified random walk simulation process. The optimization procedure was the Skip-gram model training with negative sampling. As shown in Figure 8, the proposed RMNE method was to be nearly as fast as node2vec, and it scaled linearly as the network size increased. For Node2vec, the two-layer multiplex network was merged as a single-layer network and used to learn the node embeddings. The scalability analysis of the RMNE$^+$ method is similar to the MANE (Ata et al., 2021) method.



(a) Alzheimer's (b) YouTube

(c) LinkedIn

Figure 6. Impact of structural role on the RMNE$^+$ performance.



(a) Alzheimer's (b) LinkedIn

(c) YouTube (d) Loss of RMNE$^+$

Figure 7. (a), (b), (c): Impact of hyperparameter γ on the RMNE$^+$ performance; (d): The convergence analysis of RMNE$^+$.

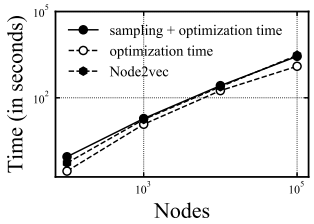


Figure 8. Runtime comparison of the Barabasi-Albert scale-free networks with an average degree of 10 for each layer.

4. Conclusion

In this work, a simple but effective role-based network embedding method for multiplex networks, named RMNE, is proposed. The contextual information of a node is obtained by a role-modified random walk approach. The proposed method samples both the within-layer neighbors and structural role members and the cross-layer structural role members of a node during the walk. Thus, the learned node embeddings can preserve the node proximity and the structural similarity in multiplex networks. Further, the RMNE$^+$ is developed to unify the different multiplex network characteristic, including collaborative and structural role information, to one framework. The proposed methods are evaluated on the network reconstruction, node classification, multi-class edge classfiic, and link prediction tasks. The experimental results on different real-world multiplex networks demonstrate the superiority of our methods over the baseline methods.

5. Acknowledgements

We thank anonymous reviewers for their constructive comments and suggestions. This research has been partially supported by grants from the National Natural Science Foundation of China (#71725001, #U1811462, and #71910107002), State key R & D Program of China (#2020YFC0832702), and Major Project of the National Social Science Foundation of China (#19ZDA092).

References

Acemoglu, D., Ozdaglar, A., and Tahbaz-Salehi, A. Systemic risk and stability in financial networks. *American Economic Review*, 105(2):564–608, 2015.

Ahmed, N., Rossi, R. A., Lee, J., Willke, T., Zhou, R., Kong, X., and Eldardiry, H. Role-based graph embeddings. *IEEE Transactions on Knowledge and Data Engineering*, 2020.

Ata, S. K., Fang, Y., Wu, M., Shi, J., Kwoh, C. K., and

Li, X. Multi-view collaborative network embedding. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 15(3):1–18, 2021.

Berger-Wolf, T. and Chawla, N. *Proceedings of the 2019 SIAM International Conference on Data Mining (SDM)*. SIAM, 2019.

Bottou, L. et al. Stochastic gradient learning in neural networks. *Proceedings of Neuro-Nımes*, 91(8):12, 1991.

Cen, Y., Zou, X., Zhang, J., Yang, H., Zhou, J., and Tang, J. Representation learning for attributed multiplex heterogeneous network. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 1358–1368, 2019.

Chu, X., Fan, X., Yao, D., Zhu, Z., Huang, J., and Bi, J. Cross-network embedding for multi-network alignment. In *The world wide web conference*, pp. 273–284, 2019.

Coleman, J., Katz, E., and Menzel, H. The diffusion of an innovation among physicians. *Sociometry*, 20(4):253–270, 1957.

Cui, P., Wang, X., Pei, J., and Zhu, W. A survey on network embedding. *IEEE Transactions on Knowledge and Data Engineering*, 31(5):833–852, 2018.

Dapeng, Z. and Xiao, F. Dynamic auto-structuring graph neural network: A joint learning framework for origin-destination demand prediction. *IEEE Transactions on Knowledge and Data Engineering*, 2021.

De Domenico, M., Nicosia, V., Arenas, A., and Latora, V. Structural reducibility of multilayer networks. *Nature communications*, 6(1):1–9, 2015.

Fu, T.-y., Lee, W.-C., and Lei, Z. Hin2vec: Explore meta-paths in heterogeneous information networks for representation learning. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, pp. 1797–1806, 2017.

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

Guo, L., Wang, W., Wu, Y. J., and Goh, M. How much do social connections matter in fundraising outcomes? *Financial Innovation*, 7(1):1–23, 2021.

Hu, B., Fang, Y., and Shi, C. Adversarial learning on heterogeneous information networks. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 120–129, 2019.

Javari, A., Derr, T., Esmailian, P., Tang, J., and Chang, K. C.-C. Rose: Role-based signed network embedding. In *Proceedings of The Web Conference 2020*, pp. 2782–2788, 2020.

Jing, B., Park, C., and Tong, H. Hdmi: High-order deep multiplex infomax. In *Proceedings of the Web Conference 2021*, pp. 2414–2424, 2021.

Kapferer, B. *Strategy and transaction in an African factory: African workers and Indian management in a Zambian town*. Manchester University Press, 1972.

Keikha, M. M., Rahgozar, M., and Asadpour, M. Community aware random walk for network embedding. *Knowledge-Based Systems*, 148:47–54, 2018.

Khosla, M., Setty, V., and Anand, A. A comparative study for unsupervised network representation learning. *IEEE Transactions on Knowledge and Data Engineering*, 2019.

Li, J., Chen, C., Tong, H., and Liu, H. Multi-layered network embedding. In *Proceedings of the 2018 SIAM International Conference on Data Mining*, pp. 684–692. SIAM, 2018.

Liu, W., Chen, P.-Y., Yeung, S., Suzumura, T., and Chen, L. Principled multilayer network embedding. In *2017 IEEE International Conference on Data Mining Workshops (ICDMW)*, pp. 134–141. IEEE, 2017.

Ma, X., Qin, G., Qiu, Z., Zheng, M., and Wang, Z. Riwalk: Fast structural node embedding via role identification. In *2019 IEEE International Conference on Data Mining (ICDM)*, pp. 478–487. IEEE, 2019.

Matsuno, R. and Murata, T. Mell: effective embedding method for multiplex networks. In *Companion Proceedings of the The Web Conference 2018*, pp. 1261–1268, 2018.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., and Dean, J. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pp. 3111–3119, 2013.

Mucha, P. J., Richardson, T., Macon, K., Porter, M. A., and Onnela, J.-P. Community structure in time-dependent, multiscale, and multiplex networks. *science*, 328(5980): 876–878, 2010.

Ni, J., Chang, S., Liu, X., Cheng, W., Chen, H., Xu, D., and Zhang, X. Co-regularized deep multi-network embedding. In *Proceedings of the 2018 World Wide Web Conference*, pp. 469–478, 2018.

Ning, N., Wu, B., and Peng, C. Representation learning based on influence of node for multiplex network. In *2018*

IEEE Third International Conference on Data Science in Cyberspace (DSC), pp. 865–872. IEEE, 2018.

Ning, N., Song, C., Zhou, P., Zhang, Y., and Wu, B. An adaptive cross-layer sampling-based node embedding for multiplex networks. In *2019 IEEE 31st International Conference on Tools with Artificial Intelligence (ICTAI)*, pp. 1515–1519. IEEE, 2019.

Ning, N., Li, Q., Zhao, K., and Wu, B. Multiplex network embedding model with high-order node dependence. *Complexity*, 2021, 2021.

Osat, S., Faqeeh, A., and Radicchi, F. Optimal percolation on multiplex networks. *Nature communications*, 8(1): 1–7, 2017.

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

Qu, M., Tang, J., Shang, J., Ren, X., Zhang, M., and Han, J. An attention-based collaboration framework for multi-view network representation learning. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, pp. 1767–1776, 2017.

Ribeiro, L. F., Saverese, P. H., and Figueiredo, D. R. struc2vec: Learning node representations from structural identity. In *Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining*, pp. 385–394, 2017.

Rossi, R. A. and Ahmed, N. K. Role discovery in networks. *IEEE Transactions on Knowledge and Data Engineering*, 27(4):1112–1131, 2014.

Rossi, R. A., Jin, D., Kim, S., Ahmed, N. K., Koutra, D., and Lee, J. B. On proximity and structural role-based embeddings in networks: Misconceptions, techniques, and applications. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 14(5):1–37, 2020.

Seninge, L., Anastopoulos, I., Ding, H., and Stuart, J. Vega is an interpretable generative model for inferring biological network activity in single-cell transcriptomics. *Nature communications*, 12(1):1–9, 2021.

Shervashidze, N. and Borgwardt, K. M. Fast subtree kernels on graphs. In *NIPS*, pp. 1660–1668, 2009.

Shi, B., Zhong, J., Qiu, H., Bao, Q., Liu, K., and Liu, J. Hybrid embedding via cross-layer random walks on multiplex networks. *IEEE Transactions on Network Science and Engineering*, 8(2):1815–1827, 2021.

Shi, Y., Han, F., He, X., He, X., Yang, C., Luo, J., and Han, J. mvn2vec: Preservation and collaboration in multi-view network embedding. *arXiv preprint arXiv:1801.06597*, 2018.

Snijders, T. A., Pattison, P. E., Robins, G. L., and Handcock, M. S. New specifications for exponential random graph models. *Sociological methodology*, 36(1):99–153, 2006.

Stolbov, M. I., Shchepeleva, M. A., and Karminsky, A. M. A global perspective on macroprudential policy interaction with systemic risk, real economic activity, and monetary intervention. *Financial Innovation*, 7(1):1–25, 2021.

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

Vickers, M. and Chan, S. Representing classroom social structure. *Victoria Institute of Secondary Education, Melbourne*, 1981.

Wilson, J. D., Baybay, M., Sankar, R., Stillman, P., and Popa, A. M. Analysis of population functional connectivity data via multilayer network embeddings. *Network Science*, 9 (1):99–122, 2021.

Zha, Q., Kou, G., Zhang, H., Liang, H., Chen, X., Li, C.-C., and Dong, Y. Opinion dynamics in finance and business: a literature review and research opportunities. *Financial Innovation*, 6(1):1–22, 2020.

Zhang, H., Qiu, L., Yi, L., and Song, Y. Scalable multiplex network embedding. In *IJCAI*, volume 18, pp. 3082–3088, 2018.

Zhang, H., Chen, X., Peng, Y., Kou, G., and Wang, R. The interaction of multiple information on multiplex social networks. *Information Sciences*, 605:366–380, 2022.

Zhang, W., Guo, X., Wang, W., Tian, Q., Pan, L., and Jiao, P. Role-based network embedding via structural features reconstruction with degree-regularized constraint. *Knowledge-Based Systems*, 218:106872, 2021.

Zitnik, M. and Leskovec, J. Predicting multicellular function through multi-layer tissue networks. *Bioinformatics*, 33 (14):i190–i198, 2017.

6. Appendix

6.1. Related works

In this section, network embedding methods for single-layer networks are briefly reviewed, and then several embedding methods proposed for multiplex networks are introduced.

6.1.1. Network embedding methods for single-layer networks

Inspired by the success of natural language processing, DeepWalk (Perozzi et al., 2014) was the first method to introduce deep learning into the field of node embeddings. Subsequently, Node2vec (Grover & Leskovec, 2016) extended DeepWalk by proposing a biased random walk. However, the above-mentioned methods, as well as LINE (Tang et al., 2015) and CARE (Keikha et al., 2018), derive community-based embedding (Rossi et al., 2020) while ignoring the structural role information on nodes. Conversely, Role2vec (Ahmed et al., 2020) was developed to learn structural role-based embeddings for single-layer networks. Thereafter, Rose (Javari et al., 2020), RiWalk (Ma et al., 2019), and RESD (Zhang et al., 2021) were proposed to learn the role-based embedding of nodes. However, all the listed embedding methods are intended for single-layer networks, and the multiplicity of networks is not considered.

6.1.2. Network embedding methods for multiplex networks

Recently, multiplex network embedding methods have received considerable attention. PMNE (Liu et al., 2017) proposed three methods—network aggregation, result aggregation, and layer co-analysis—to project a multi-layer network onto a continuous vector space. OhmNet (Zitnik & Leskovec, 2017), which represents a hierarchy-aware unsupervised node embedding method for multiplex networks, uses a hierarchy to model dependencies between network layers. MNE (Zhang et al., 2018) learns a high-dimensional node embedding and a lower-dimensional layer embedding simultaneously for each node and then combines these two embeddings to obtain the final node embedding. CrossMNA (Chu et al., 2019) adopts the same idea as the MNE. MELL (Matsuno & Murata, 2018) uses the layer vector to capture and characterize a layer's connectivity; the final node embeddings are obtained by combining the node embedding in each layer with layer embedding. GATNE (Cen et al., 2019), designed for attributed multiplex heterogeneous networks, learns the node representation embedding by combining the base, edge, and attribute embeddings. Moreover, GATNE-T performs only base and edge embeddings. HDMI (Jing et al., 2021) uses a high-order deep infomax to optimize a joint supervision signal, and then, an attention mechanism is adopted to combine node embeddings from different layers. HMNE (Ning et al., 2021) embeds node embeddings by considering high-order node dependence. Multi-node2vec (Wilson et al., 2021) introduces a third hyperparameter based on the node2vec to control the random walk between layers and then uses the Skip-gram (Mikolov et al., 2013) neural network model to learn the final node representation. The literature (Shi et al., 2021) also extends the random walk method to multiplex networks. In contrast, FFME (Ning et al., 2019) adopts an adaptive cross-layer forest fire sampling (FFS) for multiplex networks to address the bias problem of the traditional random walk. DMNE (Ni et al., 2018) is a multi-network embedding method, which can form many-to-many node mappings between different networks. MVN2VEC (Shi et al., 2018) identifies preservation and collaboration as two objectives for multi-view network representation learning. The MVE (Qu et al., 2017) method promotes the collaboration of different views by voting among them. The work (Li et al., 2018) work considers both within-layer connections and cross-layer dependencies. In addition, a multi-view collaborative network embedding method named MANE (Ata et al., 2021) considers the second-order collaboration to learn node embedding.

In summary, single-layer network embedding methods cannot capture the multiplicity of networks. Moreover, multiplex network embedding methods neglect the structural role information in the final node embeddings so that the embeddings cannot capture the structural similarity of the network. To overcome these shortcomings, this paper proposes a structural role-based embedding method for multiplex networks, named RMNE, which can preserve the node proximity and structural similarity.

6.2. Appendix A

- **DeepWalk** (Perozzi et al., 2014): DeepWalk uses a random walk to generate a node sequence, and then feeds it to the Skip-Gram architecture to learn the embedding of each node.
- **Node2vec** (Grover & Leskovec, 2016): Node2vec extends the DeepWalk by designing two hyperparameters to guide the second-order random walk.
- **Struc2vec** (Ribeiro et al., 2017): Struc2vec learns node embeddings based on structural identities.
- **Role2vec** (Ahmed et al., 2020):Role2vec uses a mapping function to assign a role to each node and then uses the attributed random walks to generate node sequence that capture structural roles.

  It should be note that the DeepWalk, Node2vec, Struc2vec, and Role2vec are the single-layer network embedding methods. In the network reconstruction task, the DeepWalk and Node2vec were applied to each layer (i.e., to $(L-1)$ layers in total) to learns the node rep-resentations, and then independently reconstructed the Lth layer. Finally, the average performance was calculated. For the two role-based network embedding methods, i.e. Struc2vec and Role2vec, we merge multiple networks into a single-layer network and then use these two methods to learn node embeddings. For the network reconstruction task of a L-layer multiplex

network, we use the $L-1$ layers to learn the node embeddings, and then use the learned embeddings to predict the edges of the Lth layer.

- **Ohmnet** (Zitnik & Leskovec, 2017): Ohmnet is a multiplex network embedding method that uses hierarchy information to model dependencies between the tissues.
- **PMNE** (Liu et al., 2017): PMNE employs three methods to learn node embeddings in multiplex networks, which are network aggregation, results aggregation, and layer co-analysis, and they are denoted as PMNE(n), PMNE(r), and PMNE(c), respectively. The PMNE(n) and PMNE(r) are based on a single-layer network embedding without leveraging the interaction between the layers, while PMNE(c) considers the influence of interactions among layers.
- **MNE** (Zhang et al., 2018): For each node, the MNE learns a common embedding and an additional embedding for each type of relationship between nodes. Then, the multiple relationships are jointly learned by a unified network embedding model.
- **MANE** (Ata et al., 2021): MANE is a multi-view network (i.e., multiplex network) embedding method considering both first- and second-order collaborations. In this work, we use MANE-1 and MANE-2 to denote the first- and second-order collaboration modes, respectively.
- **FFME** (Ning et al., 2019): FFME uses the adaptive cross-layer forest fire sampling (FFS) to address the bias prob-lem of the random walk. It also uses a node metric called the neighbor partition coefficient (NPC) to supervise the node sequence generation process.
- **GATNE-T** (Cen et al., 2019): GATNE-T splits the overall node embedding into two parts: base embedding and edge embedding. The base embedding is shared among edges of different layers, while the edge embedding is computed by aggregating the edge embeddings of a node's neighbors using the self-attention mechanism. In this work, the heterogeneous skip-gram model was used for node representation learning, and the final node embedding was obtained by aggregating the node embeddings of different layers. The aggregate operators were concatenating and average, and the corresponding methods were denoted as GATNE-c and GATNE-a, respectively.

Parameter settings: In the comparison experiments, the embedding dimension d was set to 128 for all methods. The parameters of the comparison methods were set as follows. For the DeepWalk and Node2vec, the walk length was 80, the number of random walks per node was 10, and the window size was 10; For the Node2vec, p and q were empirically set to four and one, respectively. For the Ohmnet and PMNE, the number of walks was set to 15, and the walk length was 10; also, according to (Jing et al., 2021), $p = 2$ and $q = 1$. For the PMNE(r), the embedding dimension was set to 126 for the three-layer multiplex networks. For MANE-1, $\alpha = 1$ and $\beta = 0$, and for MANE-2, $\alpha = 1$ and $\beta = 1$. The other parameters of MANE-1 and MANE-2 were set the same as the RMNE. For the FFME, MNE and GATNE-T, the default parameters settings suggested by the original paper was used. The other parameters were set the same as those suggested in the original studies. For the RMNE, we set $K = 20$, $Le = 15$, and $W = 10$. The optimal values of r and t were chosen from a set of $\{0.25, 0.5, 1.0, 2.0, 4.0\}$. For RMNE$^+$, we set $\alpha = \beta = 1$, $\gamma = 0.5$, the other parameters were set the same as the MANE (Ata et al., 2021). In addition, the role discovery process settings were the same as that of the Role2vec (Ahmed et al., 2020).

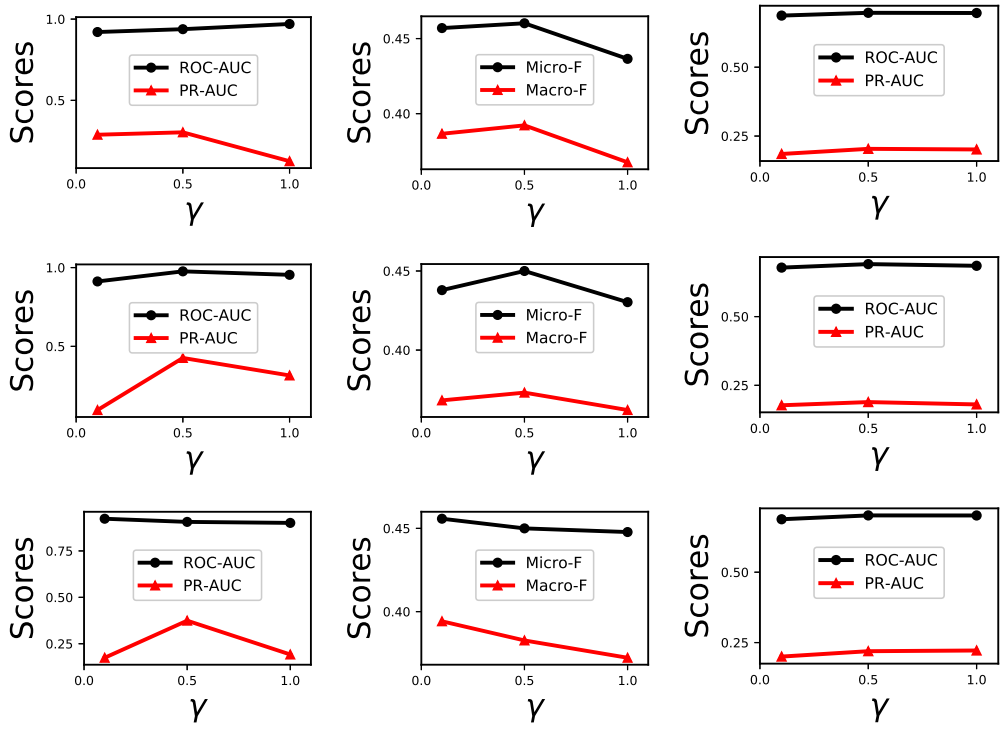
6.3. Appendix B

Table 4. Embedding operators used to obtain the representation of edges on the network reconstruction task (L1: Weighted-L1, L2: Weighted-L2).

OPERATORS	SYMBOL	DEFINITION
HADAMARD	$\odot$	$[f(u) \odot f(v)]_i = f_i(u) * f_i(v)$
AVERAGE	$\boxplus$	$[f(u) \boxplus f(v)]_i = \frac{f_i(u)+f_i(v)}{2}$
L1	$\|\cdot\|_{\overline{1}}i$	$\|f(u) \cdot f(v)\|_{\overline{1}}i = \|f_i(u) - f_i(v)\|$
L2	$\|\cdot\|_{\overline{2}}i$	$\|f(u) \cdot f(v)\|_{\overline{2}}i = \|f_i(u) - f_i(v)\|^2$

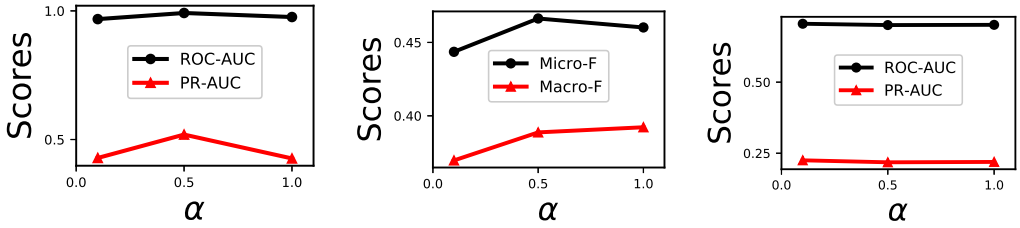
Table 5. Performance evaluation for node classification, multi-class edge classification, and link prediction tasks, the best result for each dataset is denoted in bold. The results of these baseline methods (except struc2vec and role2vec methods) are obtained from (Ata et al., 2021). The standard deviation is shown in parentheses. These methods of not showing standard deviation are due to the original paper does not provide this information.

	ALZHEIMER'S		LINKEDIN		YOUTUBE	
METRIC	ROC-AUC	PR-AUC	MICRO-F	MACRO-F	ROC-AUC	PR-AUC
SINGLE	0.6968	0.0221	0.4001	0.3468	0.6334	0.1565
DECOUPLED	0.8200	0.0735	0.4341	0.3739	0.6700	0.1649
MERGED	0.7305	0.0075	0.4197	0.3724	0.6565	0.1763
STRUC2VEC	0.5807(.198)	0.001(.000)	0.4033(.009)	0.3371(.013)	0.6517(.004)	0.1770(.001)
ROLE2VEC	0.9330(.046)	0.1759(.186)	0.4355(.001)	0.3780(.001)	0.6398(.002)	0.1571(.001)
HIN2VEC	0.5734	0.0030	0.3014	0.2175	0.6264	0.1258
HEGAN	0.6967	0.0104	0.3937	0.3467	0.6322	0.1520
MVE	0.6543	0.0161	0.4113	0.2867	0.6276	0.1650
MVN2VEC-C	0.8617	0.0890	0.4326	0.3791	0.6633	0.1621
MVN2VEC-R	0.8756	0.0275	0.4439	0.3717	0.6703	0.1669
MNE	0.9195	0.1676	0.4334	0.3538	0.6749	0.1604
DMNE	0.9357	0.0603	0.3473	0.2253	0.6056	0.1394
GATNE	0.9190	0.1227	0.3202	0.1619	0.6838	0.1624
MANE	0.9660	0.2277	0.4446	0.3865	0.6917	0.2039
RMNE$^+$	**0.9761(.006)**	**0.4264(.171)**	**0.4602(.022)**	**0.3922(.001)**	**0.7017(.003)**	**0.2194(.002)**



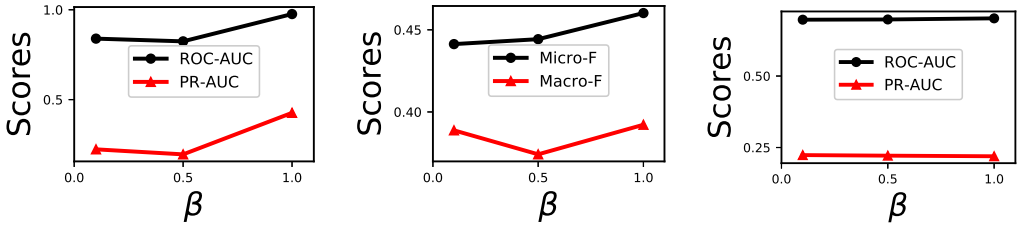
(a) Alzheimer's (b) LinkedIn (c) YouTube

Figure 9. Impact of hyperparameter γ and different role discovery methods. Top:degree-based; middle: Weisfeiler-Lehman-based; bottom: Motif-count based.



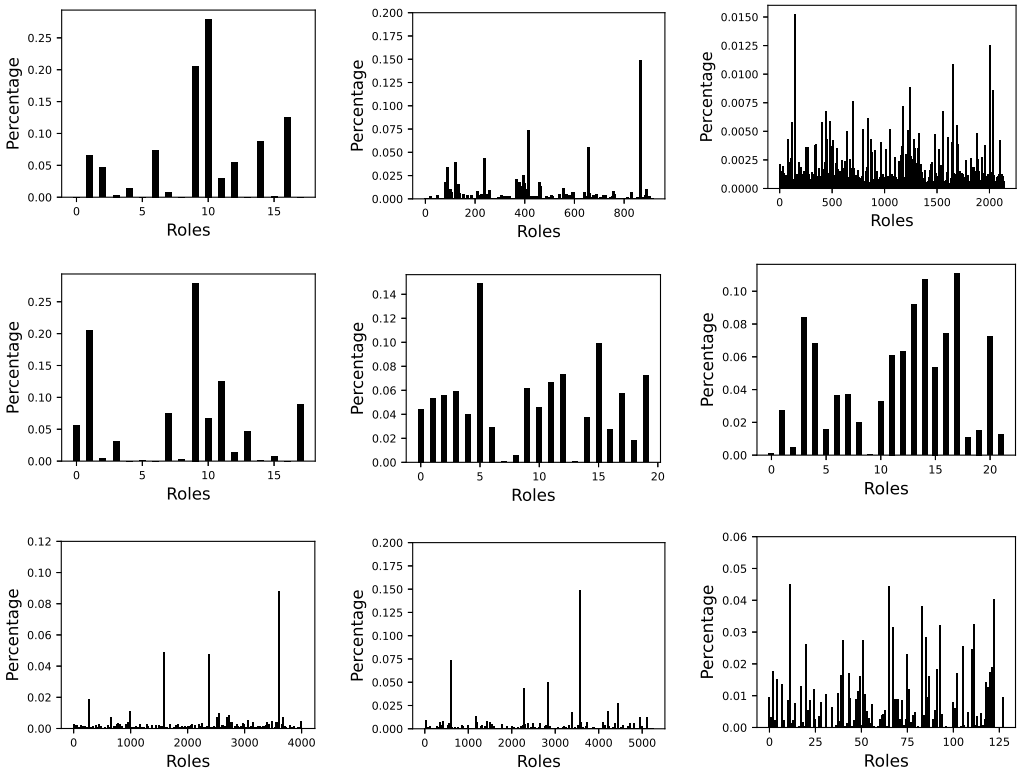
(a) Alzheimer's (b) LinkedIn (c) YouTube

Figure 10. Impact of hyperparameter α on the RMNE$^+$ performance.



(a) Alzheimer's (b) LinkedIn (c) YouTube

Figure 11. Impact of hyperparameter β on the RMNE$^+$ performance.



(a) Alzheimer's (b) LinkedIn (c) YouTube

Figure 12. The role distribution for different role discovery methods. Top: degree-based; middle: Weisfeiler-Lehman-based; bottom: Motif-count-based. In this figure, the role assignment process of the degree- and motif-count-based methods uses equivalence rules, while for the Weisfeiler-Lehman-based method, we assign roles based on the first element of nodes' feature vector (when using the entire feature vector, the number of roles of datasets Alzheimer's, LinkedIn, and YouTube are 22794, 12085, and 21945 respectively).

Table 6. AUC scores of different embedding methods combined with different operators in the network reconstruction task on five real-world multiplex networks. The best result for each dataset is denoted in bold; for each operator, the best scores are underlined (OP:operators, Sac: Sacchcere, HD:Hadamard, AG:Average). The standard deviation is shown in parentheses.

OP	METHOD	LLF	VICKERS	KTS	SAC	CKM
HD	DEEPWALK	0.5371(.013)	0.6321(.030)	0.6203(.052)	0.7903(.002)	0.7508(.001)
	NODE2VEC	0.5926(.013)	0.6306(.001)	0.6159(.004)	0.7884(.010)	0.7391(.001)
	STRUC2VEC	0.6645(.002)	0.7596(.007)	0.7860(.005)	0.7853(.009)	0.5918(.006)
	ROLE2VEC	0.6353(.018)	0.6482(.040)	0.6977(.043)	0.7574(.009)	0.9115(.004)
	MANE-1	0.5585(.020)	0.5936(.018)	0.6653(.018)	0.6334(.002)	0.6533(.010)
	MANE-2	0.6028(.012)	0.6009(.010)	0.7153(.020)	0.5342(.006)	0.7245(.009)
	OHMNET	0.7320(.010)	0.8035(.027)	0.8461(.023)	0.9112(.003)	0.9073(.006)
	PMNE (N)	0.7196(.011)	0.8524(.019)	0.8837(.018)	0.8827(.006)	0.8925(.008)
	PMNE (R)	0.7124(.010)	0.8443(.015)	0.8456(.011)	0.8766(.007)	0.8905(.010)
	PMNE (C)	0.7050(.006)	0.8501(.025)	0.8489(.014)	0.8927(.004)	0.8869(.010)
	MNE	0.7043(.034)	0.7024(.034)	0.7706(.012)	0.9138(.006)	0.9139(.008)
	GATNE-A	0.7563(.008)	0.7833(.005)	0.8271(.008)	0.8967(.006)	0.9087(.003)
	GATNE-C	0.7525(.011)	0.8239(.008)	0.8093(.007)	0.8997(.010)	0.9056(.012)
	FFME	0.7458(.008)	0.8076(.013)	0.7606(.022)	0.9121(.007)	0.9115(.005)
	RMNE	0.7606(.006)	0.8246(.006)	0.8465(.008)	0.8783(.013)	**0.9208(.002)**
AG	DEEPWALK	0.5384(.010)	0.6189(.049)	0.5996(.045)	0.7470(.006)	0.6559(.007)
	NODE2VEC	0.5911(.007)	0.6054(.001)	0.6236(.007)	0.7389(.012)	0.6522(.010)
	STRUC2VEC	0.6673(.003)	0.7620(.005)	0.7849(.005)	0.8198(.004)	0.5918(.006)
	ROLE2VEC	0.6100(.011)	0.6600(.031)	0.7092(.032)	0.7183(.008)	0.7272(.005)
	MANE-1	0.5852(.018)	0.6839(.014)	0.7428(.025)	0.7475(.007)	0.5511(.008)
	MANE-2	0.5954(.016)	0.6837(.013)	0.7407(.016)	0.7768(.006)	0.5603(.009)
	OHMNET	0.6049(.009)	0.6836(.011)	0.7662(.022)	0.8353(.004)	0.5668(.010)
	PMNE (N)	0.6003(.009)	0.7007(.020)	0.7427(.012)	0.8489(.006)	0.5720(.017)
	PMNE (R)	0.5895(.008)	0.6934(.024)	0.7443(.021)	0.7534(.010)	0.5546(.016)
	PMNE (C)	0.5889(.009)	0.6918(.014)	0.7359(.017)	0.8320(.008)	0.5667(.011)
	MNE	0.5893(.016)	0.6807(.032)	0.7499(.011)	0.7951(.011)	0.5658(.008)
	GATNE-A	0.5838(.009)	0.7336(.015)	0.7586(.012)	0.8246(.010)	0.5188(.007)
	GATNE-C	0.5897(.010)	0.7337(.009)	0.7391(.003)	0.8397(.006)	0.5357(.003)
	FFME	0.5859(.009)	0.6203(.022)	0.5445(.019)	0.8321(.004)	0.5624(.016)
	RMNE	0.6088(.003)	0.7093(.007)	0.7439(.004)	0.8371(.007)	0.5581(.003)
L1	DEEPWALK	0.5486(.014)	0.6475(.022)	0.6250(.020)	0.7212(.013)	0.6783(.002)
	NODE2VEC	0.5874(.010)	0.6137(.002)	0.6117(.005)	0.7218(.011)	0.6772(.002)
	STRUC2VEC	0.6568(.003)	0.7449(.008)	0.7600(.006)	0.8037(.004)	0.5949(.004)
	ROLE2VEC	0.6286(.008)	0.6734(.025)	0.7248(.023)	0.7627(.005)	0.7881(.004)
	MANE-1	0.5652(.010)	0.6209(.008)	0.6720(.009)	0.8612(.021)	0.6471(.004)
	MANE-2	0.5930(.005)	0.6209(.012)	0.7258(.005)	0.8638(.011)	0.6889(.002)
	OHMNET	0.6954(.008)	0.7675(.009)	0.8036(.016)	0.9121(.012)	0.8940(.007)
	PMNE (N)	0.7190(.008)	0.8430(.011)	0.8532(.009)	0.8897(.008)	0.8736(.007)
	PMNE (R)	0.7202(.007)	0.8260(.007)	0.8660(.006)	0.8241(.005)	0.8632(.006)
	PMNE (C)	0.6965(.007)	0.8394(.007)	0.8515(.007)	0.5285(.007)	0.8815(.006)
	MNE	0.6959(.004)	0.6909(.016)	0.7287(.018)	0.8361(.007)	0.9002(.007)
	GATNE-A	0.7409(.012)	0.8309(.012)	0.8540(.020)	0.8200(.008)	0.9126(.008)
	GATNE-C	0.7559(.010)	0.8501(.007)	0.8481(.002)	0.8787(.006)	0.9103(.011)
	FFME	0.7627(.005)	0.8492(.007)	0.8357(.008)	0.7069(.005)	0.8930(.005)
	RMNE	0.7945(.002)	0.8836(.005)	0.9041(.004)	0.9227(.005)	0.9022(.002)
L2	DEEPWALK	0.5485(.012)	0.6475(.021)	0.6321(.055)	0.7012(.009)	0.6855(.001)
	NODE2VEC	0.5834(.019)	0.6195(.003)	0.5952(.004)	0.7060(.010)	0.6855(.001)
	STRUC2VEC	0.6509(.003)	0.7301(.013)	0.7367(.006)	0.7865(.005)	0.5893(.004)
	ROLE2VEC	0.6400(.008)	0.6770(.024)	0.7307(.019)	0.7980(.004)	0.8203(.004)
	MANE-1	0.5625(.015)	0.6062(.021)	0.6773(.001)	0.8579(.009)	0.6527(.012)
	MANE-2	0.6021(.011)	0.6094(.010)	0.7373(.014)	0.8614(.010)	0.6844(.020)
	OHMNET	0.7066(.009)	0.7817(.010)	0.7963(.011)	0.9130(.008)	0.8890(.007)
	PMNE (N)	0.7322(.006)	0.8511(.008)	0.8654(.010)	0.8907(.002)	0.8793(.008)
	PMNE (R)	0.7202(.006)	0.8519(.010)	0.8610(.002)	0.8536(.004)	0.8654(.005)
	PMNE (C)	0.7009(.006)	0.8470(.008)	0.8606(.007)	0.5322(.008)	0.8768(.010)
	MNE	0.7053(.004)	0.7290(.017)	0.7424(.023)	0.8352(.006)	0.9022(.008)
	GATNE-A	0.7578(.008)	0.8199(.001)	0.8171(.008)	0.8271(.001)	0.9029(.010)
	GATNE-C	0.7564(.006)	0.8559(.006)	0.8406(.002)	0.8850(.015)	0.9014(.001)
	FFME	0.7733(.006)	0.8499(.005)	0.8289(.014)	0.7054(.006)	0.8979(.005)
	RMNE	**0.8017(.002)**	**0.8957(.005)**	**0.9109(.004)**	**0.9274(.004)**	0.9151(.002)