Hindawi
Complexity
Volume 2021, Article ID 6644111, 18 pages
https://doi.org/10.1155/2021/6644111

Research Article

Multiplex Network Embedding Model with High-Order Node Dependence

Nianwen Ning, Qiuyue Li, Kai Zhao, and Bin Wu

*Beijing Key Laboratory of Intelligent Telecommunications Software and Multimedia,
Beijing University of Posts and Telecommunications, No. 10 Xitucheng Road, Beijing 100876, China*

Correspondence should be addressed to Bin Wu; wubin@bupt.edu.cn

Received 9 October 2020; Revised 22 January 2021; Accepted 21 February 2021; Published 8 March 2021

Academic Editor: Shenghua Liu

Multiplex networks have been widely used in information diffusion, social networks, transport, and biology multiomics. They contain multiple types of relations between nodes, in which each type of the relation is intuitively modeled as one layer. In the real world, the formation of a type of relations may only depend on some attribute elements of nodes. Most existing multiplex network embedding methods only focus on intralayer and interlayer structural information while neglecting this dependence between node attributes and the topology of each layer. Attributes that are irrelevant to the network structure could affect the embedding quality of multiplex networks. To address this problem, we propose a novel multiplex network embedding model with high-order node dependence, called HMNE. HMNE simultaneously considers three properties: (1) intralayer high-order proximity of nodes, (2) interlayer dependence in respect of nodes, and (3) the dependence between node attributes and the topology of each layer. In the intralayer embedding phase, we present a symmetric graph convolution-deconvolution model to embed high-order proximity information as the intralayer embedding of nodes in an unsupervised manner. In the interlayer embedding phase, we estimate the local structural complementarity of nodes as an embedding constraint of interlayer dependence. Through these two phases, we can achieve the disentangled representation of node attributes, which can be treated as fined-grained semantic dependence on the topology of each layer. In the restructure phase of node attributes, we perform a linear fusion of attribute disentangled representations for each node as a reconstruction of original attributes. Extensive experiments have been conducted on six real-world networks. The experimental results demonstrate that the proposed model outperforms the state-of-the-art methods in cross-domain link prediction and shared community detection tasks.

1. Introduction

The abundant relations and views between entities can be collected from various sources or scenarios, allowing a slew of problems to be better solved in different application domains, e.g., information diffusion [1], social network analysis [2], intelligent transportation [3], biomedicine, and ecology [4, 5]. Taking together these data may be able to give a more accurate and nuanced picture of network structure than the individual network alone [6]. Taking social networks as an example, different online social networks show different views and behavior patterns of people. A user makes connections to their friends on Facebook or WeChat but uses Twitter or Weibo to follow people that interested him/her. Though different online social networks present distinct views and aspects of social behavior of one same user with the consistent feature, abundant user features and social information can facilitate the construction of a more accurate and nuanced user profile. Therefore, these multiple sources and views of network data are worth exploring because they often contain complementary information that improves the quality of analysis results [7].

Intuitively, modeling the information fusion problem of nodes as a feature fusion problem is a straightforward way. Based on the fused features, we can further mine the network data for node classification, link prediction, node clustering, and visualization. Multiple-relation or view network data are vividly modeled as a multiplex network (also known as multidimensional, multiview, or multilayer networks) [8–12] in which the same set of nodes are connected by different types of relations. Different from a single network, multiplex networks reflect more complex topological properties. Multiplex networks can not only present the intralayer dependence between nodes but also can well model the interlayer network dependence. The analysis of multiple networks not only needs to consider the interdependence or interaction between nodes at the intralayer and interlayer but also focus on the dependence of node attributes and the topological structure of nodes. In this paper, the high-order node dependence of multiplex networks is defined as intralayer dependence between nodes, interlayer dependence in respect of anchor nodes, and the dependence between node attributes and the topology of each layer. In a multiplex network, the information fusion of multiple layers of nodes is a significant fundamental issue for the joint analysis of networks. A multiplex network, as shown in the middle of Figure 1, is composed of three social networks, which are Douban (https://www.douban.com/), LinkedIn (https://www.linkedin.com/), and Weibo (https://weibo.com/). These three social networks are geared towards different social scenarios; Douban provides books and music services, LinkedIn serves for social occupation, and Weibo is geared towards entertainment services. Multiplex network representation learning (also known as multiplex network embedding) is an effective method to analyze and mine the network. It can project the node (or network) into a continuous low-dimensional space. In this paper, we are motivated to focus on multiplex network representation learning considering the high-order dependence.

Recently, existing methods have achieved excellent performance in the intralayer dependence between nodes. However, few studies have comprehensively focused on the properties unique to multiplex networks. The first challenge is preserving high-order proximity information of nodes. Some state-of-the-art models based on the graph neural network (GNN) [10, 13, 14] take into account both intralayer and interlayer dependencies of nodes. However, due to the oversmoothing problem of GNN models [15], such methods cannot effectively preserve high-order proximity information. The second challenge is preserving the interlayer dependence property of multiplex networks. The layers with strong interlayer dependence have similar local structure characteristics, while those with weak interlayer dependence show obvious differences in the local topology [16]. From Figure 1, we can see that the nodes in the Douban layer and the Weibo layer have similar local structures. It indicates the interlayer dependence property of nodes in these two layers. This dependency cannot be preserved by extended random walk-based methods [17–19] and GNN-based methods [20–22]. The extended random walk-based representation learning method realizes the generation of node sequences through cross-layer sampling. In the node sampling process, most of them use random strategy to cross-layer sampling, but this ignores the similarity between layers. For GNN-based methods, nodes are embedded independently in interlayer. The node embedding of each layer is concatenated in the later stage. Such embedding and fusion processes will introduce repetitive and redundant information. However, LinkedIn layer is dissimilar with the other two layers. In this situation, it makes the fusion embedding of nodes obtained by methods [23–26] based on the assumption of information sharing between layers is inaccurate. The third challenge is preserving the dependence between node attributes and the topology of each layer. Previous studies also ignore the interaction of node attributes with the topology of each layer. Figure 1 illustrates this important property that different social scenarios depend on different attribute information of the user. The formation of friendship in the Douban network mainly depends on the user's preference for music and books. The formation of the following relationships in LinkedIn mainly depends on attributes such as the user's job and education level. The formation of relationships in Weibo mainly depends on the user's multiple attributes (books, sports, and music) besides job and education. In support of the dependence between the node attributes and the network structure, the interaction between them has been shown in several cases [27–29]. Therefore, the embedding of multiplex networks contains not only dependence information between nodes in each layer (intralayer dependence) but also local structure similarity information (interlayer dependence) and dependence between node attributes and the topology of each layer (attribute dependence).

In light of this, we propose a novel and hierarchy representation learning model for multiplex networks with node attributes called HMNE. We propose a symmetric graph convolution-deconvolution (GCD) method with multiple convolution layers to embed the intralayer adjacency information of a node as a low-dimensional dense vector in an unsupervised manner. The graph convolution module (GCM) preserves high-order proximity information, and the graph deconvolution module (GDM) serves as an embedding restriction to alleviate the oversmoothing problem of GCM. To preserve interlayer local dependence information, inspired by Graph Infomax [30], we use the similarity between the representations obtained by the multilayer convolution and the entire layer embedding as the estimation of complementary information. We fit this estimation of complementary information to actually quantify the local structural complementarity of nodes. For the dependence between attributes and the topology of the layer where the node is located, we treat the output of the graph deconvolution module as the disentangled representation of node attributes. Each disentangled representation is the result of the interaction between node attributes and the topology of each layer.

The main contributions of this paper are summarized as follows:

(i) We propose a symmetrical graph convolution-deconvolution neural network model to achieve intralayer node embedding, which is an unsupervised and general representation learning method. This method can not only flexibly adjust the number of hidden layers to capture the high-order structural information but also avoid the oversmoothing problem.

(ii) We present a method to estimate interlayer complementary information. This method can measure

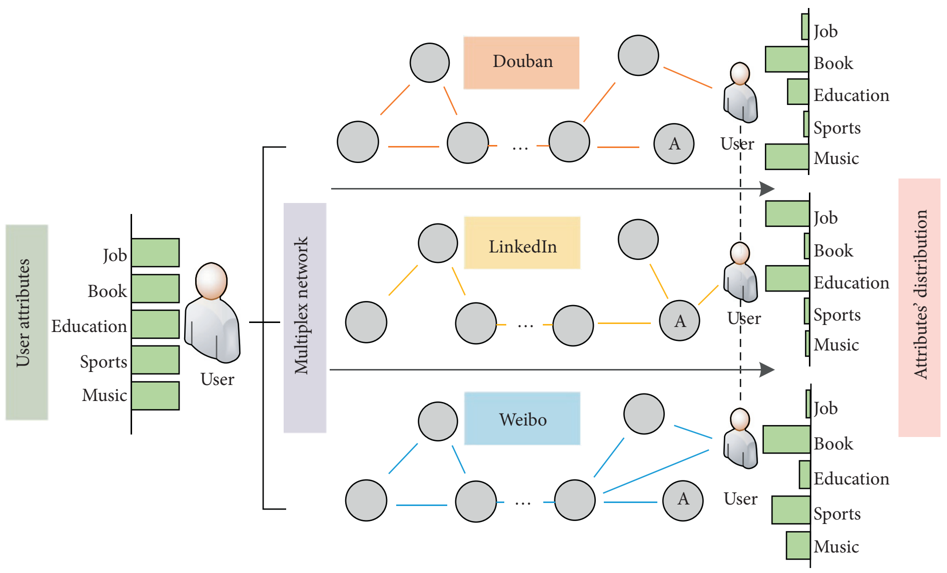


Figure 1: The illustration of our data structure and dependence between node attributes and each layer's topology for three-layer multiplex networks as an example. The first part on the left presents node attributes. The second part on the middle indicates the multiplex network data model. The latter part shows the different dependence between node attributes and each layer.

the interlayer dependence property [9, 10, 31] in respect of the topology of the layer where the node is located and constrain the intralayer embedding.

(iii) We design a disentangled representation learning architecture to solve the dependence between node attributes and its local topology. Graph deconvolution component is used to select attribute fragments associated with the semantics of each layer. We use a linear layer to restructure the original node attributes.

(iv) Extensive evaluations on real-world datasets have been conducted, and the experimental results demonstrate the superiority of the proposed HMNE model against the state-of-the-art models.

The rest of the paper is organized as follows. Section 2 describes some related works. Section 3 introduces related definitions of the data model we use, problem formulation, and preliminary knowledge. Section 4 presents HMNE's core modules. Section 5 shows the experiment results. Finally, the summary and outlook are described in Section 6.

2. Related Work

In this section, to distinguish from the single-layer network, we call the traditional representation learning method of one network as single-layer network embedding and the embedding of multiple networks as multiplex network embedding. Among them, we introduce the related work from joint embedding and cooperative embedding of multiplex network embedding. We first describe the ideas of network embedding for a single-layer network. Then, we, respectively, introduce related works about multiplex network (mainly involves multiview networks, multirelation networks, multidimensional networks, and multilayer networks) embedding methods. Finally, we also summarize the shortcomings of these related works and the similarities and dissimilarities with the proposed model.

2.1. Single-Layer Network Embedding

2.1.1. Random Walk-Based Methods. Embedding techniques based on random walk to obtain node representations have been proposed: DeepWalk [32] is the first algorithm based on random walk to learn node representation. Based on the breadth-first search and depth-first search, node2vec [33] was proposed to replace the node sampling strategy of the DeepWalk method. Both algorithms are traditional single-layer network embedding. Gu et al. [34] proposed an approach based on the open-flow network model to reveal the underlying flow structure and its hidden metric space of different random walk strategies on networks. It shows that the essence of network embedding by random walk is the latent metric defined on the open-flow network. In order to learn the representation of multirelation heterogeneous information networks, the following algorithm is proposed. Dong et al. [35] proposed a strategy for random walk sampling from heterogeneous networks, where the random walk is restricted to transition between particular types of nodes. This strategy allows many methods to be applied to heterogeneous graphs and complements the idea of taking type-specific encoders and decoders into account. Ribeiro et al. [36] presented struc2vec, a novel and flexible framework with the target to learn latent representations for the structural identity of nodes. The framework uses a hierarchy to measure node similarity at different scales and constructs a multilayer graph to encode structural similarities and generate a structural context of nodes.

2.1.2. Graph Neural Network-Based Methods. Kipf and Welling [37] introduced the variational graph autoencoder (VGAE), a framework for unsupervised learning on graph-structured data based on the variational autoencoder (VAE). This model makes use of latent variables and is capable of learning interpretable latent representations for undirected graphs. Hamilton et al. [38] proposed GraphSAGE, which uses a two-layer deep neural architecture. In each convolution layer, a node computes its representation as an aggregation of its neighbors' representations (from the previous layer). In addition, to achieve unsupervised embedding, the parameters of aggregation functions are learned using the loss function similar to DeepWalk. GraphSAGE is incapable of selective neighbor sampling and has a lack of memory of known nodes that have been trained. To address these problems, Luo and Zhuo [39] proposed an unsupervised method that samples neighborhood information attended by co-occurring structures and optimizes a trainable global bias as a representation expectation for each node in the given graph. Velickovic et al. [30] presented Deep Graph Infomax (DGI), a general approach for learning node representations within graph-structured data in an unsupervised manner. DGI relies on maximizing mutual information between patch representations and corresponding high-level summaries of graphs, both derived using established graph convolution network architectures. Li et al. [29] proposed a principled unsupervised feature selection framework ADAPT to find informative features that can be used to regenerate the observed links and further characterize the adaptive neighborhood structure of the network. Yu et al. [15] proposed KS2L, a novel graph Knowledge distillation regularized Self-Supervised Learning framework, with two complementary regularization modules, for intra- and cross-model graph knowledge distillation. Xiao et al. proposed three rumor propagation models based on evolutionary game and antirumor [40], data enhancement [41], and representation learning [42]. They proved that rumors are not only influenced by antirumor information but also affected by user behavior and psychological factors. And they studied the user's network structure and historical behavior characteristics in the rumor topic communication space in social networks and predicted the user behavior in the next time slice based on the current time slice data. At the same time, they introduced evolutionary game theory and considered the internal and external factors that affect user behavior within rumor propagation.

2.2. Multiplex Network Embedding. The goal of multiplex network embedding methods is to achieve the information fusion of multiple features of networks, in which these methods can be divided into joint representation learning and coordinated representation learning [43] (in Figure 2 of [44], an illustration of coordinated and joint representation learning is presented).

2.2.1. Joint Representation Learning. Zhang et al. [24] proposed a scalable multiplex network embedding (MNE) method, which assumes that the same nodes in multiple networks preserve certain common features and unique features of each layer. Thus, the common and unique embedding of nodes in each layer is learned by the DeepWalk algorithm separately. Ma et al. [25] implemented node embedding for multidimensional networks with hierarchical structure. They simply added up node embedding in multiple dimensions as the fusion feature of nodes in multiple networks. Matsuno and Murata [26] presented a multilayer network embedding method (MELL) that captures and characterizes each layer's connectivity. The method utilizes the overall structure to consider the similar or complementary structure of the layer. Finally, the fusion feature learning of nodes in multiplex networks is obtained by combining node embedding in each layer with layer vectors. Cen et al. [9] focused on embedding learning for attributed multiplex heterogeneous networks, where different types of nodes might be linked with multiple different types of edges, and each node is associated with a set of different attributes. GATNE splits the overall node embedding into three parts: base embedding, edge embedding, and attribute embedding. GATNE-T contains only the first two parts. Zhao et al. [45] proposed a novel and principled approach: a multiview adversarial completion model (MV-ACM). Each relation space is characterized in a single viewpoint, enabling us to use the topological structural information in each view. Yuan et al. [46] proposed a novel multiview network embedding model with node similarity ensembles. Node similarities are first selected to maximize the represented network information while minimizing the information redundancy. For each combination of the selected node similarities, a latent space is generated as a view of the network.

2.2.2. Coordinated Representation Learning. In some cases, graphs have multiple "layers" that contain copies of the same nodes. They can be beneficial to share information across layers so that a node's embedding in one layer can be informed by its embedding in other layers. Qu et al. [47] proposed an attention-based method (MVE) to learn the weights of views for different nodes with a few labeled data. MVE can obtain robust node representations across different views by vote strategy. Recently, Liu et al. [17] extended a standard graph mining into the area of the multilayer network. The proposed methods ("network aggregation," "results' aggregation," and "layer coanalysis") can project a multilayer network of a continuous vector space. Zitnik and Leskovec [18] proposed the OhmNet framework to learn the features of proteins in different tissues. They represented each tissue as a network, where nodes represent proteins. Individual tissue networks act as layers in a multilayer network, where they use a hierarchy to model dependencies between the layers (i.e., tissues). Schlichtkrull et al. [20] introduced relational graph convolution networks (R-GCNs) and applied them to two standard knowledge base completion tasks: link prediction (recovery of missing facts, i.e., subject-predicate-object triples) and entity classification (recovery of missing entity attributes). Zhiyuli et al. [48] proposed highly scalable node embedding for link prediction in large-scale networks. The method learns node pairs' co-occurrence features to embed a node into a vector by a damping-based random walk

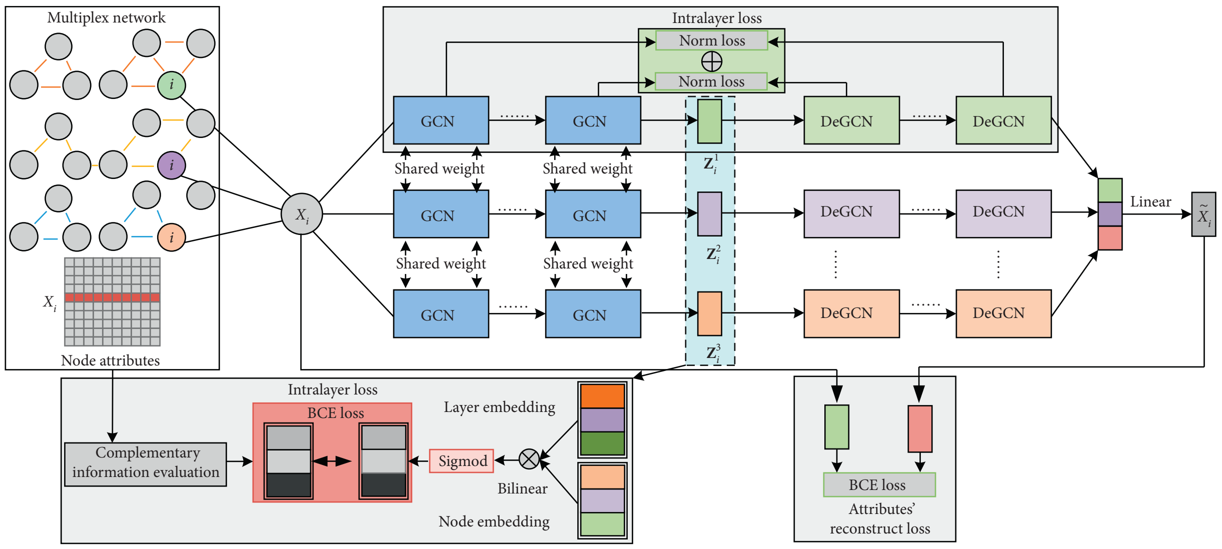


FIGURE 2: The architecture of HMNE. Our model includes the calculation of three types of loss functions, namely, intralayer embedding loss with shared weight, interlayer embedding loss, and global reconstruction loss. In this figure, GCN is a graph convolution neural layer, and DeGCN is a graph deconvolution neural layer.

algorithm. In the node sampling process, there is a bias problem with these existing methods that samples are trapped in a local structure. In addition, cross-layer sampling heavily depends on fixed parameters, which is in an inflexible manner. Sun et al. [49] presented a MNGAN framework for multiview network embedding by the generative adversarial network, aimed at preserving the information from the individual network views while accounting for connectivity across different views. Wei et al. [50] proposed an attributed node random walk framework, which can not only be able to incorporate both topology and attribute information flexibly but also easily deal with missing data and is applied to large networks. For the multiple-network alignment problem, Chu et al. [51] proposed a cross-network embedding method (CrossMNA). It defines two categories of embedding vectors for each node: intervector, and intravector. The idea of CrossMNA is the same as that of MNE. They thought intravector contains both the commonness among counterparts and the specific local connections in its selected network due to the semantics. Park et al. [10] presented a simple yet effective unsupervised network embedding method for the attributed multiplex network called DMGI, inspired by Deep Graph Infomax (DGI), which maximizes the mutual information between local patches of a graph and the global representation of the entire graph. Vashishth et al. [21] proposed a novel graph convolutional framework (COMPGCN) which jointly embeds both nodes and relations in a relational graph. COMPGCN leverages a variety of entity-relation composition operations from knowledge graph embedding techniques and scales with the number of relations. Yu et al. [22] proposed a novel GEneralized Multirelational Graph Convolutional Networks framework, which combines the power of GCNs in graph-based belief propagation and the strengths of advanced knowledge-based embedding methods, and goes beyond.

In summary, in response to the challenges presented in this paper, the single-layer network embedding methods cannot achieve the preservation of interlayer-dependent information. The joint representation learning methods of multiplex networks assume that nodes have shared embeddings in interlayer, and information sharing and transfer are realized through these embeddings. However, different levels of dependence between layers will cause this assumption to be invalid (please refer to Figure 3 of literature [44]). The existing coordinated representation learning methods neglect node attributes and their local topology. Aggregating these coarse-grained attributes in the graph neural network can include noise and affect the performance of the model. In order to fill this gap, we propose a hierarchical multiplex network embedding (HMNE) model with high-order node dependence. The specific implementation will be described in detail in Section 4.

3. Data and Problem Formulations

In this section, we describe related symbols, concepts, and definitions in detail. Our data model's basic concepts are introduced in Section 3.1. Then, we formalize a generalized node embedding problem of multiplex networks in Section 3.2. The important notations are summarized in Table 1.

3.1. Data Model. In terms of network data of multiple views and sources, it is more appropriate to represent such networks as multiplex networks. As shown in Figure 1, three layers of this multiplex network are derived from three modal data, such as social network, semantic relation network, and co-occurrence network. Multiplex networks can not only express the intralayer link but also can well model the dependencies and interactions between networks [44]. The detailed definitions of multiplex networks are as follows.

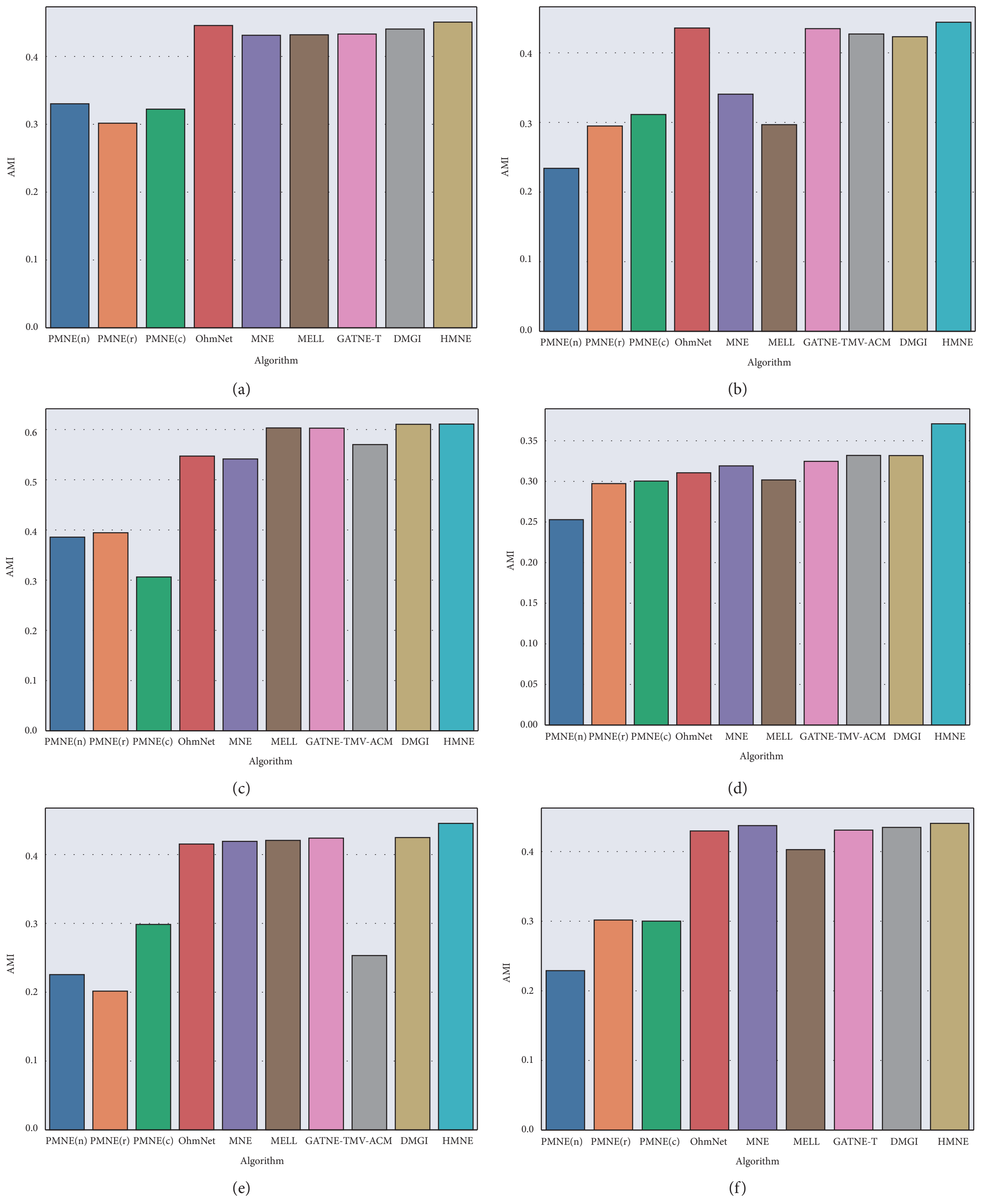


Figure 3: The AMI scores of HMNE in real-world multiplex networks. For the cross-domain link prediction task, AMI is regarded as an indicator to evaluate the embedding performance. (a) Celegans. (b) CKM. (c) CS-Aarhus. (d) London. (e) Vickers. (f) Ack-co-author.

Definition 1 (multiplex network architecture). Given a multiplex network of N nodes with the sets of layer L, in which each node can interact with the other ones through $|L|$ kinds of relations with $|L| \geq 2$, we denote an aligned multiplex network $G = \{G^l(\mathscr{V}, \mathscr{E}^l), l \in L\}$ which is made up of $|L|$ layers with $N = |\mathscr{V}|$ nodes and $E = |\sum_{l\in L} \mathscr{E}^l|$ edges.

Table 1: Notations.

Notation	Explanation
G	A multiplex network
$\mathcal{V}, \mathcal{E}^{\alpha}$	The sets of nodes and edges in layer α, respectively
G^l, A^l	A network/adjacency matrix of layer l, respectively
N, E^l	The node number/edge number of layer l, respectively
$\mathcal{X}$	$\{x_i \mid i \in \mathcal{V}\}$, the set of node attributes
$\mathcal{H}$	The node representations of multiplex networks
i, x_i	A node and its features, respectively
$\deg(i)$	The degree of node i
$e_{i,j}^l$	An edge between i and j in layer l
k	The number of neural network layers
$\mathcal{N}_i^l$	Neighbors of node i in layer l
$\mathbf{h}_i^l$	The learned representation of node i in layer l
d	The dimension of node representations
$\mathbf{Z}^l, \widetilde{\mathbf{Z}}^l$	The output of GCC/GDC in layer l of G
$\mathbf{Z}^{(k)}, \widetilde{\mathbf{Z}}^{(k)}$	The output of the k-th convolution/deconvolution layer
Θ, Θ_d	The convolution/deconvolution kernel

Each layer in multiplex networks has the same node set and different edge sets, as shown in the middle part of Figure 1. Let $i, j \in \mathcal{V}$ be two nodes. i^l denotes node i at layer l, and $e_{i,j}^l \in \mathcal{E}^l$ denotes the edge to link i^l and j^l in layer l. i^l and i^l are the duplicates of the same node i in different layers. We assume that nodes i^l and $j^{l'}$ can be implicitly linked by the duplicates of i in layer l' and $e_{i,j}^{l'}$ cross-layers l and l'. Figure 1 shows an illustrative example of a multiplex network with $|L| = 3$-layer network (i.e., $L = \{\text{Douban, LinkedIn, Weibo}\}$) and a target node User. The dotted line represents an anchor link. $e_{\text{User},A}^{\text{LinkedIn}}$ is an edge between node User and node A in layer LinkedIn. $e_{\text{User},A}^{\text{Douban,LinkedIn}}$ is a cross-layer link between node $\text{User}^{\text{Douban}}$ and node A^{LinkedIn} through an anchor link.

3.2. Problem Formulation

Definition 2 (multiplex network representation learning). Suppose the methods make use of a real-valued superadjacency matrix A, $A \in R^{(N\times|L|, N\times|L|)}$ (e.g., representing text or metadata associated with nodes). Node embedding aims at learning a map function $f\colon A \longrightarrow \mathcal{H}$.

f is a function, which maps $A_i = \{A_i^{l_1}, A_i^{l_2}, \ldots, A_i^{l_{|L|}}\} \in R^{(N,|L|)}$ to a d-dimensional representation of node i, and A_i is a group of vectors of node i in the superadjacency matrix of G, and it can also be understood that it is composed of adjacency matrices of multiple layers. $\mathcal{H}$ is a d-dimensional vector/tensor, and $d \ll N$. For coordinated representation learning, $\mathbf{h}_i$ is a vector for node i. For joint representation learning, $\mathbf{h}_i$ is a tensor for node i. Notice that all the aforementioned definitions can be easily extended to the case of weighted networks. We only focus on coordinated representation learning in this paper.

4. Proposed Model

In this section, we introduce the overall model of our HMNE by addressing the three major challenges mentioned in Section 1:

(1) Preserving high-order proximity information of nodes: as shown in Figure 2, a symmetric graph convolution-deconvolution network (SGCD) model is designed to solve the oversmoothing problem of the traditional GCN. GCD includes the graph convolution component (GCC) and graph deconvolution component (GDC). We formulate a restriction constraint for the GDC to restructure the original input feature of the GCC. The output feature of the GCC with K (graph) convolution layers $\mathbf{x}_i^k$ in respect of node i is inputted into the GDC for reconstructing original input feature x_i. Even if many graph convolution layers are added to the GCC, the oversmoothing problem can be avoided because of this reconstruction constraint. Therefore, we can conveniently preserve high-order proximity information of nodes by increasing the graph convolution layers.

(2) Preserving the interlayer dependence property of multiplex networks: as shown in Figure 2, there are two major components to capture the intralayer dependence property of multiplex networks. We first utilize a structural similarity metric method to measure the difference target layer l and the other layer l', respectively, in respect of node i. The result is served as a structural complementary information estimation P_{true}. Then, the similarity measure between the embedding $\mathbf{h}_i^l$ of node i in the target layer l and the global embedding $\mathbf{H}^{l'}$ in the other layer l' is served as the complementary information P_{pred} in respect of node i. Through the minimization of P_{pred} and P_{true}, the learned embedding of node i can preserve the dependency property between layers.

(3) Preserving the dependence of node attributes with the topology of each layer: as shown in Figure 2, the input feature is attributes x_i of node i. We utilize the idea of disentanglement learning to disentangle x_i as $|L|$ attribute subsets. These attribute subsets dependent on the topology of each layer have different semantic information. Three main processes are as follows: firstly, we use x_i of node i as the input of the GCC. Then, the embedding $\mathbf{h}_i^l$ of node i with attribute information and structure information is obtained by the GCC in layer l of multiplex networks. Finally, the disentangled representations of the node's attributes are the output of the GDC in each layer. In the GCC, the attributes associated with the topology of each layer are preserved. In the GDC, the structure information is disentangled from $\mathbf{h}_i^l$.

4.1. Preserving High-Order Proximity

4.1.1. Graph Convolution. In spectral-based graph convolution models, a mathematical representation of an undirected graph is the normalized graph Laplacian matrix defined as $\mathbf{L} = \mathbf{I}_n - \mathbf{D}^{-(1/2)}A\mathbf{D}^{-(1/2)}$, where $\mathbf{D}$ is a diagonal matrix of node degrees. The normalized Laplacian matrix can be factored as $\mathbf{L} = \mathbf{U}\Lambda\mathbf{U}^{\mathrm{T}}$, where Λ is the diagonal matrix of eigenvalues. The eigenvectors of the normalized Laplacian matrix form an orthonormal space; in mathematical words, $\mathbf{U}^T\mathbf{U} = \mathbf{I}$. In graph signal processing, a feature vector of node i of a graph is a graph signal $x_i \in \mathbf{R}^N$.

The graph Fourier transform to a signal x_i is defined as $\mathcal{F}(x_i) = \mathbf{U}^T x_i$, and the inverse graph Fourier transform is defined as $\mathcal{F}^{-1}(\hat{x}_i) = \mathbf{U}\hat{x}_i$, where $\hat{x}_i$ represents the resulting signal from the graph Fourier transform. The graph convolution of the input signal x_i with a convolution kernel (filter) $\mathbf{g}$ is defined as

$$x_i *_G \mathbf{g} = \mathcal{F}^{-1}(\mathcal{F}(x_i) \odot \mathcal{F}(\mathbf{g})) = \mathbf{U}\left(\mathbf{U}^T\mathbf{x} \odot \mathbf{U}^T\mathbf{g}\right), \quad (1)$$

where $\odot$ denotes the Hadamard product. If we denote a filter as $\mathbf{g}_\theta = \operatorname{diag}(\mathbf{U}^T\mathbf{g})$, then the graph convolution is simplified as

$$x_i *_G \mathbf{g} = \mathbf{U}\mathbf{g}_\theta\mathbf{U}^T x_i. \quad (2)$$

The graph convolution component from [52] limits the layerwise convolution operation to alleviate the problem of overfitting on local neighborhood structures for graphs with very wide node degree distributions. The equation simplifies to

$$x_i *_G \mathbf{g} \approx \theta_0' x_i + \theta_1'(\mathbf{L} - \mathbf{I}_N)x_i = \theta_0' x_i - \theta_1'\mathbf{D}^{-(1/2)}\mathbf{A}\mathbf{D}^{-(1/2)}x_i. \quad (3)$$

After constraining the number of parameters with $\theta = \theta_0' = -\theta_1'$, we can obtain the following expression:

$$x_i *_G \mathbf{g} \approx \theta\left(\mathbf{I}_N + \mathbf{D}^{-(1/2)}\mathbf{A}\mathbf{D}^{-(1/2)}\right)x_i. \quad (4)$$

Kipf and Welling [52] introduced the trick: $\mathbf{I}_N + \mathbf{D}^{-(1/2)}\mathbf{A}\mathbf{D}^{-(1/2)} \approx \widetilde{\mathbf{D}}^{-(1/2)}\widetilde{\mathbf{A}}\widetilde{\mathbf{D}}^{-(1/2)}$, where $\widetilde{\mathbf{A}} = \mathbf{A} + \mathbf{I}_N$ and $\widetilde{\mathbf{D}}_{ii} = \sum_j \widetilde{\mathbf{A}}_{ij}$. Finally, we treat Θ as a convolution kernel (a matrix of filter parameters), a general definition of graph convolution as follows:

$$\mathbf{Z} = \widetilde{\mathbf{D}}^{-(1/2)}\widetilde{\mathbf{A}}\widetilde{\mathbf{D}}^{-(1/2)}\mathbf{X}\mathbf{\Theta}. \quad (5)$$

In order to express the following sections more clearly, we denote $\mathbf{Z}^l$ as the node embedding of layer l of a multiplex network G and $\mathbf{Z}^{(k)}$ as a node embedding output of the k-th layer of the graph convolution neural network.

4.1.2. Graph Deconvolution. To capture the high-order proximity information of the nodes, we can simply stack multiple convolution layers as our HMNE's graph convolution component (GCC) based on equation (5). However, previous studies showed that graph convolution is a type of Laplacian smoothing. They proved that, after repeatedly applying Laplacian smoothing many times, the features of the nodes in the (connected) graph would converge to similar values. To avoid this problem and capture the high-order proximity of nodes, we design a graph deconvolution component (GDC). We first take the output $\mathbf{Z}^{(k)}$ of the k (multiple) stacked convolution neural layers as the input of the GDC. Then, analogous to the definition of deconvolution in the field of computer vision, according to equation (5), a graph deconvolution layer with a deconvolution kernel Θ_d is defined as

$$\widetilde{\mathbf{Z}} = \widetilde{\mathbf{D}}^{-(1/2)}\widetilde{\mathbf{A}}\widetilde{\mathbf{D}}^{(1/2)}\mathbf{Z}^{(k)}\Theta_d, \quad (6)$$

where A is an adjacency matrix, $A \in R^{N\times N}$, $\widetilde{A} = A + I_N\widetilde{D}$ is a degree matrix, and $\widetilde{D}_{ii} = \sum_j A_{ij}$. The embedding of nodes $\widetilde{\mathbf{Z}}^{(k)}$ is an output of the k-th layer of the graph deconvolution neural network.

4.1.3. Intralayer Embedding Loss. In this initialization of the GDC, the input matrix $\widetilde{\mathbf{Z}}^{(1)}$ is $\mathbf{Z}^{(k)}$, where k is the number of graph convolution layers, and $\mathbf{Z}^{(k)}$ is a final node embedding matrix according to equation (5). To separate the structure information from the input by the deconvolution kernel, we propose an intralayer embedding loss formula. We use a symmetric structure containing k convolution layers and k deconvolution layers as our graph convolution-deconvolution component (SGCD). The reconstruction loss formula of SGCD is

$$L_{\text{intra}} = \sum_{j=1}^{(K/2)} \left\|\widetilde{\mathbf{Z}}^{(j)} - \mathbf{Z}^{(K-j)}\right\|_2^2. \quad (7)$$

We assume the input $\mathbf{Z}^{(1)}$ of the GCC is the node attributes $\mathcal{X}$ so that the output $\widetilde{\mathbf{Z}}^{(k)}$ of the GDC is a reconstruction matrix in respect of $\mathcal{X}$. This reconstruction process is significant for Section 4.3.

4.1.4. Node Representation Learning. In order to preserve the attribute and structural information of the node in each layer, we need to aggregate the embedding $\mathbf{h}_i^l$ $(l \in L)$ of node i in each layer to obtain a more complementary global node embedding $\mathbf{h}_i$. We use a sum function to integrate the embeddings of node i in each layer:

$$\mathbf{h}_i = \sum_{l\in L} \mathbf{h}_i^l. \quad (8)$$

Then, the final embedding of nodes in multiplex networks is

$$\mathcal{H} = \sum_{l\in L} \mathbf{Z}^l. \quad (9)$$

The final embedding $\mathbf{h}_i^l \in \mathbf{Z}^l$ of node i in layer l obtained by the GCC, where $\mathbf{Z}^l \in \mathcal{R}^{N\times d}$ and $\mathbf{h}_i^l \in \mathcal{R}^{1\times d}$, is a row of $\mathbf{Z}^l$.

4.2. Preserving Interlayer Dependence

4.2.1. Interlayer Structure Complementary Information Estimation. To capture the interlayer dependence between layers, we introduce how to get the true sample (l', l, i), which indicates layer l' is complementary for i in l. This complementary information computing can effectively measure the interlayer dependence property. The basic idea is that the more dissimilar the local structures in two layers, the more reason to believe complementary information exists between these two layers. So, we utilize the structural similarity between two layers to produce true samples. Let $P_{\text{true}}(\cdot \mid i, l)$ denote the true underlying connecting distribution of node i in layer l, and we can estimate it as

$$p(j \mid i, l) = \frac{e^{l}_{i,j}}{\sum_{v\in\mathcal{V}} e^{l}_{i,v}}. \tag{10}$$

Then, the locally topological structural similarity of node i between layers l' and l can be calculated by Jensen–Shannon distance between $P_{l',i} = P(\cdot \mid i, l')$ and $P_{l,i} = P(\cdot \mid i, l)$ as

$$D_{\text{JS}}\left(P_{l',i} \| P_{l,i}\right) = \frac{1}{2}\left[D_{\text{KL}}\left(P_{l,i} \| M\right) + D_{\text{KL}}\left(P_{l',i} \| M\right)\right], \tag{11}$$

where $M = ((P_{l,i} + P_{l',i})/2)$ and D_{KL} is the Kullback–Leibler divergence:

$$D_{\text{KL}}(P \| Q) = \sum_{v} P(v) \log \frac{P(v)}{Q(v)}. \tag{12}$$

Note that when the locally topological structures of node i between layers l' and l are identical, $D_{\text{JS}}(P_{l',i} \| P_{l,i}) = 0$; otherwise, $D_{\text{JS}}(P_{l',i} \| P_{l,i}) = 1$. So, we get $S_{\text{struc}}(l', l \mid i) = 1 - D_{\text{JS}}(P_{l',i} \| P_{l,i})$ as the locally topological structural similarity between layers l' and l regarding node i. Finally, we can estimate $P_{\text{true}}(\cdot \mid l, i)$ and sample true layers according to the distribution:

$$P_{\text{true}}(\cdot \mid l, i) = \underset{l'\in L}{\Delta} \frac{S_{\text{struc}}(l', l \mid i)}{\sum_{r\in L} S_{\text{struc}}(r, l \mid i)}, \quad l' \in L, \tag{13}$$

where Δ denotes a function that can concatenate each element successively. Actually, this structure complementary information estimation can be served as the similarity of node i in layer l with respect to the topology of the layer where the node is located.

4.2.2. The Interlayer Dependence Estimation of Nodes.

In order to realize the interlayer dependence property, inspired by the idea of Deep Infomax in [53], we regard the membership of node i in layer l for layer l' as a measure of the interlayer local dependency of node i. Therefore, a layer-level embedding $\mathcal{H}^{l}$ of layer l in multiplex networks is computed by employing a readout function Readout: $R^{n\times d} \longrightarrow \mathcal{R}^{d}$.

$$\mathcal{H}^{l} = \text{Readout}\left(\mathbf{Z}^{l}\right) = \sigma\left(\frac{1}{N}\sum_{i=1}^{N} \mathbf{h}^{l}_{i}\right), \tag{14}$$

where $\mathbf{Z}^{l}$ is a final embedding matrix of layer l in the graph convolution component, $\mathbf{h}^{l}_{i}$ is an embedding of node i of the l layer, and σ is a logistic sigmoid nonlinearity function.

Based on the layer-level embedding and the embedding of each node in this layer, we calculate the measure of the interlayer dependence property of node i in layer l on layer l'. In this paper, we apply a simple bilinear scoring function as it empirically performs the best in our experiments:

$$\text{Score}(l' \mid l, i) = \text{Score}\left(\mathbf{h}^{l}_{i}, \mathcal{H}^{l'}\right) = \sigma\left(\mathbf{h}^{l}_{i}\mathbf{W}\mathcal{H}^{l'}\right), \tag{15}$$

where σ is the logistic sigmoid nonlinearity and $\mathbf{W} \in \mathcal{R}^{d\times d}$ is a trainable scoring matrix. We can estimate the interlayer local dependence measure of the nodes by calculating the scores of the nodes' embedding in each layer and the global embedding of each layer:

$$P_{\text{pred}}(\cdot \mid l, i) = \underset{l'\in L}{\Delta} \text{Score}(l' \mid l, i), \tag{16}$$

where $P_{\text{pred}}(\cdot \mid l, i)$ denotes a vector of interlayer dependence of node i in layer l in respect of the duplication of node i in each layer and Δ denotes a function that can concatenate each element successively.

4.2.3. Interlayer Dependence Loss.

Comparing equation (13) with (16), we have designed an objective function with BCELoss loss function for saving the node interlayer dependence property:

$$\begin{aligned} L_{\text{inter}} = \frac{1}{N}\sum_{i\in\mathcal{V}}\sum_{l'=1}^{L} &-\left[P_{\text{true}}(l' \mid l, i)\log\widetilde{P}_{\text{pred}}(l' \mid l, i)\right. \\ &\left.+ \left(1 - P_{\text{true}}(l' \mid l, i)\right)\log\left(1 - \widetilde{P}_{\text{pred}}(l' \mid l, i)\right)\right]. \end{aligned} \tag{17}$$

4.3. Preserving Dependence between Attributes and Topology.

In order to preserve the dependence between attributes and the topology of each layer, the original attributes of nodes are fed into the GCC. We perform GCC and GDC processes to disentangle the attributes of nodes as different semantic representations. We believe that the GCC can strengthen the attribute value related to the layer's semantic in the node attributes. GDC can disentangle the attributes of nodes with structure information of nodes. This is the main advantage of our GCD (intralayer embedding) method compared with the graph autoencoder and variational autoencoder. Then, in the final graph deconvolution network phase, each final output embedding of the GDC for each layer of multiplex networks is aggregated by a concatenate function. A linear layer is used to reconstruct the original attributes x_i, which makes the overall model framework designed as an autoencoder architecture. Based on the embedding of node i in layer l and the embedding of node i in other layers, we construct a simple nonlinear fusion method to obtain the reconstruction attributes $\widetilde{x}_i$ of node i:

$$\widetilde{x}_i = \sigma\left(W\Delta^{L}_{l'=1}\widetilde{\mathbf{Z}}^{l'}_{i}\right), \tag{18}$$

where σ is a sigmoid nonlinearity activation function, W is the trainable parameters, and $\widetilde{\mathbf{Z}}^{l'}_{i}$ is the output of the GDC of i node in the l' layer network. Then, we also utilize the

BCELoss function to calculate the loss between original attributes x_i and reconstruction attributes $\widetilde{x}_i$ of node i:

$$L_{\text{attr}} = \frac{1}{N} \sum_{i \in \mathcal{V}} \sum_{l \in L} -\left(\left(x_i \log \widetilde{x}_i^l + (1 - x_i) \log\left(1 - \widetilde{x}_i^l\right)\right)\right), \quad (19)$$

where L is the layer number of multiplex networks and x_i is the attributes of the i node.

Finally, the global loss function of HMNE also considers the loss of different components. Therefore, we simply sum all the loss functions as the loss of the entire model and use Adam optimizer for backpropagation and parameter learning. The loss function of HMNE is

$$L = L_{\text{inter}} + L_{\text{intra}} + L_{\text{attr}}. \quad (20)$$

4.4. The Optimization and Time Complexity. We present the node representation learning process (HMNE) for multiplex networks in Algorithm 1. The total time complexity of HMNE is $O(\text{TNE}|L|^2)$ where T is the number of iterations, N is the number of nodes in each layer, E is the number of edges of the multiplex network, and $|L|$ is the number of layers.

5. Experiment Analysis

In this section, we study the performance of HMNE in different real-world datasets. We use cross-domain link prediction and shared community detection tasks to verify the performance of HMNE.

5.1. Datasets. For our experiments, we conduct HMNE and compare baseline methods on each of the following multiplex networks. These datasets contain two categories: public datasets and private dataset. Public datasets are composed of five multiplex network benchmark datasets involving social, biological, and transportation. Private dataset is an interesting semantic network dataset that we construct. This dataset is a network of acknowledgment relationships extracted from the acknowledgment part of dissertation data and the coauthor network of corresponding entities from AMiner (https://www.aminer.cn/). The specific information about public and private datasets is shown in Table 2.

5.1.1. Public Datasets. These multinetwork datasets were collected on M. De Domenico's homepage (https://comunelab.fbk.eu/manlio/index.php), and the processed datasets are available (https://github.com/Brian-ning/HMNE/).

Vickers classroom social multiplex network: this dataset was collected by Vickers from 29 seventh-grade students in a school in Victoria, Australia. Students were asked to nominate their classmates on a number of relations (class, best friend, and work).

CS-Aarhus social multiplex network: this dataset consists of five kinds of online and offline relationships (Facebook, leisure, work, coauthorship, and lunch) between the employees of the computer science department at Aarhus. These variables cover different types of relations between the actors based on their interactions.

London multiplex transport network: this dataset was collected in 2013 from the official website of Transport for London and manually cross-checked. Nodes are train stations in London, and edges encode existing routes between stations. Tube, overground, and DLR stations are considered.

CKM physicians' innovation multiplex network: this dataset was collected by Coleman, Katz, and Menzel on medical innovation, considering physicians in four towns in Illinois: Peoria, Bloomington, Quincy, and Galesburg. They were concerned with the impact of network ties on the physicians' adoption of a new drug, tetracycline. These views are advice, discussion, and friend.

Celegans multiplex connectome network: this dataset considered different types of genetic interactions for organisms in the Biological General Repository for Interaction Datasets (BioGRID, thebiogrid.org), a public database that archives and disseminates genetic and protein interaction (ElectrJ, MonoSyn, and PolySyn) data from humans and model organisms.

These networks have been used as benchmark datasets for evaluating multiplex network analysis methods. In addition, the CKM dataset has ground-truth information about the community label of nodes. Therefore, HMNE performs performance testing of the cross-domain link prediction task on all datasets and performs performance testing of the shared community detection task on the CKM dataset.

5.1.2. Private Dataset. This dataset is a two-layer network constructed from two views, one of which is a coauthor network constructed in the form of author co-occurrence from common paper data (from AMiner). Another view is to take the author of the dissertation as the central node from each acknowledgment chapter of the dissertation data, the named entity (including tutor, teacher, classmate, or family member) identified in the acknowledgment text as the neighbor node, and the co-occurrence of the entity as the edge constructed from the center network (ego network). Based on the acknowledgment text of the dissertation and paper data, the acknowledgment layer network and coauthor layer network of the Ack-co-author dataset are constructed, respectively.

5.2. Baseline Methods. In these experiments, we test 14 other comparison algorithms: 11 baseline methods with the same parameters and dimensions and 3 traditional methods. The explanations of these baseline methods are as follows. Some of these methods can be used to test two tasks simultaneously. Other methods can only be suited for one of two tasks. The details of baseline methods are as follows:

(i) CN (common neighbor): it captures the notion that two nodes that have a common neighbor may be introduced by that neighbor. It has the effect of

```
Input: graph $G = \langle \mathscr{V}, \mathscr{E}, L, \mathscr{X} \rangle$; neural layer number $K \geq 3$ for GCC and GDC, graph convolution/deconvolution kernel $\Theta$, $\Theta_d$,
iteration times $T$.
Output: $\mathscr{H}$: the node embeddings of multiplex network $G$
(1)  begin
(2)    Initialize all parameters for GCC and GDC with $K$ neural layers, respectively.
(3)    $t = 1$
(4)    while $t \geq T$ or not converge do
(5)      for $l$ in $L$ do
(6)        Sample nodes and calculate $P_{(l,\cdot)}$ in layer $l$ based on equation (15).
(7)        Generate convolution embedding $\mathbf{Z}^l$ using $\chi$ and $G^l$ by equation (5)
(8)        Readout the embedding $\mathscr{H}^l$ of layer $l$ by equation (14)
(9)        Generate disentangled embedding $\widetilde{\mathbf{Z}}^l$ using $\mathbf{Z}^l$ and $G^l$ by equation (6).
(10)     end
(11)     Calculate $P_{\text{true}}$ by equations (11) and (13).
(12)     Calculate $P_{\text{pred}}$ by equation (16).
(13)     Update $\Theta$ and $\Theta_d$ by minimizing equations (7) and (17).
(14)     Generate the reconstruction attributes $\widetilde{\chi}$ by $\widetilde{\mathbf{Z}}^l$ and equation (18).
(15)     Update $\Theta$, $\Theta_d$, and $W$ by minimizing equation (19).
(16)     $t += 1$
(17)   end
(18)   Incorporate node embedding $\mathscr{H}$ by equation (9).
(19) end
(20) return the node representation $\mathscr{H}$.
```

Algorithm 1: HMNE model.

Table 2: Basic statistics about different multiplex networks used in this study.

Name	Nodes	Edges	Layers	Description
Vickers	29	740	3	Class: 316; best friend: 226; work: 198
CS-Aarhus	61	620	5	Facebook: 193; leisure: 124; work: 21; coauthor: 87; lunch: 195
London	369	441	3	Tube: 312; overground: 82; DLR: 46
CKM	246	1551	3	Advice: 480; discussion: 565; friend: 506
Celegans	279	5863	3	ElectrJ: 1031; MonoSyn: 1639; PolySyn: 3193
Ack-co-author	3383	29128	2	Acknowledgment: 1733; coauthor: 1285

"closing a triangle" in the graph and likes a common mechanism in real life.

(ii) JC (Jaccard coefficient): it is a measure used for gauging the similarity and diversity of sample sets and is defined as the size of the intersection divided by the size of the union of the sample sets.

(iii) AA (Adamic/Adar): it is a measure to predict links, according to the number of shared links between two nodes. It is defined as the sum of the inverse logarithmic degree centrality of the neighbors shared by the two nodes.

(iv) AAMT [54]: it is a link prediction method for multiplex networks based on the Adamic/Adar coefficient neighbor similarity, which considers the intensity and structural overlap of multiplex links simultaneously.

(v) Node2vec [33]: it adds a pair of parameters to achieve BFS and DFS sampling process on the single-layer network. It makes it better for capturing the role of nodes, such as hubs or tail users.

(vi) OhmNet [18]: it is a node embedding method for multiplex networks, where hierarchy information is used to model dependencies between the layers.

(vii) PMNE [17]: it has three methods of node embedding, each of which generates a common embedding of each node by merging multiple networks. We compare these three models with other baseline methods. We denote "network aggregation," "results' aggregation," and "coanalysis model" as PMNE(n), PMNE(r), and PMNE(c), respectively.

(viii) MNE [24]: it is a scalable multiplex network embedding. It contains one high-dimensional common embedding and a lower-dimensional additional embedding for each type of relations. Then, multiple relations can be learned jointly based on a unified network embedding model.

(ix) MELL [26]: it is a novel embedding method for multiplex networks, which incorporates an idea of layer vector that captures and characterizes each layer's connectivity. This method exploits the overall structure effectively and embeds both directed and undirected multiplex networks, whether their layer structures are similar or complementary.

(x) GraphSAGE [38]: it is a graph neural network framework for inductive representation learning on graphs. GraphSAGE is used to generate low-dimensional vector representations for nodes and is especially useful for graphs that have rich node attribute information. We use an unsupervised learning version of GraphSAGE to serve as a baseline method of the link prediction task.

(xi) GATNE-T [9]: it considers the network structure and uses base embeddings and edge embeddings to capture the influential factors between different edge types. The attention mechanism is used to capture the influential factors between different edge types.

(xii) DMGI [10]: it is a simple yet effective unsupervised network embedding method for the attributed multiplex network, inspired by Deep Graph Infomax (DGI), which maximizes the mutual information between local patches of a graph and the global representation of the entire graph.

(xiii) MV-ACM [45]: it is a novel multiview adversarial completion model (MV-ACM). Each relation space is characterized in a single viewpoint, enabling them to use the topological structural information in each view.

(xiv) GenLouvain [55]: it is a modularity-based multiplex network community detection algorithm. The algorithm not only considers the modularity within the layer but also considers the modularity between layers. By maximizing the modularity metrics, the algorithm completes the community detection task. We only use this algorithm as a baseline method for the node clustering task.

In this paper, we only apply CN, JC, AA, node2Vec, and GraphSAGE to link prediction tasks at the single layer where test edges are located at. For OhmNet, we construct a hierarchy describing relationships between different layers randomly. We regard the common embedding in the MNE algorithm as the global embedding of nodes. For MELL, we add layer-level embedding as the global-level embedding and then add it to the node-level embedding of the test node. AAMT uses the multiplexity property of nodes (interlayer information) and similarity between nodes (intralayer information) to predict the probability of link. For GATNE-T and MV-ACM, we only use the homogeneous skip-gram model for node representation learning. The categorical multislice network model is selected for GenLouvain. Besides the same walk length, walk times and embedded dimensions are set as the same parameters of HMNE, and we also set other experimental baseline methods using the default parameters, such as PMNE, MELL, and DMGI.

5.3. Experimental Setup. For implementing the network feature extraction module, we use representation learning of nodes to extract the feature of each layer. In these datasets we use, if nodes in these datasets have no attributes, we use the adjacency matrix of merged multiplex networks as the attribute information of nodes in compared experiments. The definition of the matrix is the adjacency matrix of the multilayer network after the multilayer network is aggregated or flattened (that is, the union of edges for each layer). The matrix can reflect that the topology of nodes in different networks depends on the network topology. In other words, neighbor nodes (denote node attributes) are dependent on the formation of the node topology under different semantics. We set $p = 2$ and $q = 1$ as default parameters in the biased sample process of the node2vec method. We set the number of walks to 20 and walk length to 30 for OhmNet, node2vec, PMNE (n, r, c), MNE, MELL, GraphSAGE (unsupervised vision), GATNE-T, and MV-ACM. The dimension of embedding is set to 128 for all methods. For GATNE-T, DMGI, MV-ACM, and our HMNE, the optimizer of the model is Adam, the learning rate is selected from {0.0001, 0.002}, and the batch size is 50 (except for the Vickers dataset). For three heterogeneous embedding network methods, an edge is usually input into the model as a meta-path for training. All the experiments are conducted on a Linux server with sixteen logical CPUs on Intel Xeon E5 CPU and four GTX 1080Ti GPUs. Notice that, in the community detection task, we uniformly remove the community label in the node attributes for representation learning. Although our model can alleviate the oversmoothing problem of the current graph neural network algorithm, to verify this feature of our model, we show the effect of different layers of the neural network on the model performance. According to the experiment results, it is a tradeoff between the performance and complexity of the model to use a 2-layer graph neural network in both compared experiments.

5.4. Cross-Domain Link Prediction. In this section, we perform the cross-domain link prediction task on these multiplex networks. We refer to the experimental settings of the multiplex networks of literature [45]. For the cross-domain link prediction task, we remove 20% of edges of each layer in the original network and use the area under the curve (AUC) score and adjusted mutual information (AMI) score to evaluate the performance of these algorithms for predicting missing edges in each layer. We use the residual (80%) edges of each layer for training and the 20% of edges randomly selected from each layer for testing. These node pairs in edge sets of the test set are regarded as positive examples. Then, we randomly sample an equal number of node pairs from the test set, in which no edge connecting node pairs are served as negative examples. AUC is the area under the receiver operating characteristic (ROC) curve, which is equal to the probability that a classifier ranks a randomly chosen positive example higher than a randomly chosen negative one. With Pos positive examples and Neg negative examples, AUC can be calculated by

$$\mathrm{AUC} = \frac{\sum_{i\in +}\mathrm{rank}_i - (\mathrm{Pos}\,(1+\mathrm{Pos})/2)}{\mathrm{Pos}\times\mathrm{Neg}}. \tag{21}$$

Mutual information (MI) is also used to measure the degree of agreement between the two data distributions.

Assuming that U and Y are the distribution of N sample labels, the entropy of the two distributions is

$$\begin{aligned} P(i) &= \frac{|U_i|}{N}, \\ \widetilde{P}(j) &= \frac{|Y_j|}{N}, \\ H(U) &= \sum_{i=1}^{|U|} P(i)\log(P(i)), \\ \widetilde{H}(Y) &= \sum_{j=1}^{|Y|} \widetilde{P}(j)\log(\widetilde{P}(j)), \\ \mathrm{MI}(U,Y) &= \sum_{i=1}^{|U|}\sum_{j=1}^{|Y|} P(i,j)\log\left(\frac{\widetilde{P}(i,j)}{\widetilde{P}(i)\widetilde{P}(j)}\right), \\ \mathrm{AMI} &= \frac{\mathrm{MI} - E[\mathrm{MI}]}{\max(H(U),H(V)) - E[\mathrm{MI}]}, \end{aligned} \tag{22}$$

where $E[\mathrm{MI}]$ is an expected value of mutual information. The range of AMI values is $[-1, 1]$, and its value is larger, which means that the result is more consistent with the real situation.

We calculate the similarity between nodes by CN, JC, and AA metrics in the layer where the test node pair is located. For other single-layer network embedding methods, we train a separate embedding for each relation type of the network to predict links on the corresponding edges. It means that they do not have information from other layers of multiplex networks. We aim to verify the interlayer dependence can provide complementary structure information from other layers. In terms of node embedding methods, we use the cosine function of vectors as a similarity metric. The larger the similarity scores are, the more likely there exists a link between them.

From Table 3 and Figure 3, we can know that HMNE is significantly better than other comparison algorithms. Our model shows better performance on multiplex network datasets than single-layer methods such as CN, JC, AA, node2vec, and GraphSAGE, which directly proves that fusing different structural information by preserving the interlayer dependence property can improve the accuracy of the cross-domain link prediction task. This property of the multiplex network can provide critical complementary information from other layers. We regard OhmNet, PMNE, MNE, MELL, GATNE-T, DMGI, and HMNE as comparative experimental groups. These compared algorithms are the latest multiplex network representation learning methods to learn multiplex network representation. OhmNet and PMNE are extensions of the traditional single-layer network embedding method, but there is no direct consideration of the interlayer dependence property in the final embeddings. It leads to an inevitable loss of information in the embedding process, so the complementary information of the interlayer cannot be well preserving. For MNE and MELL methods, the common (or layer) embedding is considered based on the assumption that nodes have similar local structures in different layers. In fact, this assumption is rare, and it also affects the generalization ability of the algorithm. This process of interlayer node embedding based on common embedding can lead to distortion and inaccuracy of information. GATNE-T, DMGI, and MV-ACM are specially designed to handle such a scenario that the nodes have different types and attributes in each layer, so they cannot show excellent performance in the problem we are trying to solve. Moreover, these three methods ignore the dependence property between node attributes and the topology of the layer where the node is located. For our model, HMNE simultaneously considers intralayer, interlayer, and attribute dependence properties of nodes in the node embedding process.

5.5. Shared Community Detection. Shared community detection task aims to group similar nodes so that nodes in the same group are more similar to each other than those in different groups. In other words, each node in a multiplex network has different relations/views and only belongs to a unique community. In the CKM dataset, nodes have the global community label. For this dataset, this task is usually called a shared community detection task, which is a significant mining task in multiplex network analysis. Therefore, we treat the CKM dataset as the benchmark dataset of the shared community detection task. For these methods based on node representation learning, we use K-means++ algorithm to calculate the cluster of the final embedding of nodes. In order to evaluate fairness, we set the number of communities (clusters) to 2.

5.5.1. Evaluation Metrics. Given the ground-truth community in the real-world datasets, we use normalized mutual information (NMI) to evaluate the performance of the methods:

$$\mathrm{NMI}(X\,|\,Y) = 1 - \frac{H(X\,|\,Y) + H(Y\,|\,X)}{2}, \tag{23}$$

where X and Y denote two partitions of the network and $H(X\,|\,Y)$ denotes the normalized conditional entropy of partition X with respect to Y shown in the following equation:

$$H(X\,|\,Y) = \frac{1}{|C|}\sum_k \frac{H(X_k\,|\,Y)}{H(X_k)}, \tag{24}$$

where $|C|$ denotes the number of communities. The larger the NMI is, the better the result is. The value of NMI takes from 0 to 1. It is equal to 1 meaning two partitions match perfectly and is equal to 0 on the contrary.

In the domain of node clustering, the chance-corrected version of this measure is adjusted Rand index (ARI). It is known to be less sensitive to the number of parts. It is possible to say that two elements of $\mathbf{Y}$, i.e., (x, x'), are paired in $\mathbf{P}$ if they belong to the same cluster. Let $\mathbf{Q}$ and $\mathbf{U}$ be two partitions of the object set $\mathbf{Y}$. A formally formulation of the adjusted Rand index is

Table 3: Cross-domain link prediction task. All the results are the averaged AUC scores.

Node type	Network type	Algorithm	Datasets					
			Celegans	CKM	CS-Aarhus	London	Vickers	Ack-co-author
Homogeneous network	Single layer	CN	0.7467	0.6517	0.8855	0.5054	0.7932	0.5104
		JC	0.7330	0.6526	0.8883	0.5054	0.7864	0.5102
		AA	0.7524	0.6523	0.8962	0.5054	0.8145	0.6968
		Node2vec	0.7847	0.8021	0.8997	0.6816	0.6667	0.5097
		GraphSAGE	0.7629	0.8521	0.7023	0.5160	0.7571	0.3991
	Multiple layers	AAMT	0.8604	0.8239	0.9232	0.5266	0.7389	0.6968
		OhmNet	0.8427	0.8576	0.8826	0.3580	0.7841	0.8060
		PMNE(n)	0.5012	0.4773	0.5154	0.4993	0.5013	0.4981
		PMNE(r)	0.4945	0.5043	0.5205	0.4782	0.5002	0.5076
		PMNE(c)	0.5003	0.4757	0.5047	0.5043	0.4955	0.4983
		MNE	0.6313	0.7902	0.8842	0.4526	0.7048	0.7093
		MELL	0.8085	0.7599	0.9014	0.4991	0.7923	0.7227
Heterogeneous network		MV-ACM	—	0.8538	0.7966	0.7630	0.7810	—
		GATNE-T	0.8142	0.8605	0.8897	0.6631	0.8165	0.8152
Homogeneous and heterogeneous network		DMGI	0.8557	0.8535	**0.9275**	0.7501	0.8028	0.8203
Homogeneous network		HMNE	**0.8730**	**0.8669**	0.9252	**0.7785**	**0.8178**	**0.8223**

$$\mathrm{ARI} = \frac{2(ad - bc)}{b^2 + c^2 + 2ad + (a+d)(c+b)}, \tag{25}$$

where a is the number of pairs $(y, y') \in \mathbf{Y}$ that are paired in $\mathbf{Q}$ and in $\mathbf{U}$; b is the number of pairs $(y, y') \in \mathbf{Y}$ that are paired in $\mathbf{Q}$ but not paired in $\mathbf{U}$; c is the number of pairs $(y, y') \in \mathbf{Y}$ that are not paired in $\mathbf{Q}$ but paired in $\mathbf{U}$; and d is the number of pairs $(y, y') \in \mathbf{Y}$ that are neither paired in $\mathbf{Q}$ nor in $\mathbf{U}$. This index has an upper bound of 1 and takes value 0 when the Rand index is equal to its expected value.

5.5.2. Result Analysis. As shown in Table 4, HMNE shows excellent performance in the shared community detection task. Among them, HMNE has obtained the largest NMI and ARI scores. In terms of other methods, MNE and MELL learn a representation of a node separately in each layer. We sum the representations in different layers of nodes as the global embedding of nodes and compare them with our model. Therefore, the performance of MNE and MELL in this task shows that this kind of join representation learning algorithm cannot well preserve the shared community information of nodes. Compared with MV-ACM, GATNE-T, and DMGI that can handle heterogeneous networks, our model can show more excellent performance in the shared community detection task. The comparison with methods GATNE-T, MV-ACM, and DMGI that can handle heterogeneous networks shows that our model also has good performance. Unlike them, HMNE takes into account the high-order proximity property of nodes. The property encourages node embeddings for an identical community is similar. It should be noted that due to the use of the iterative strategy of maximizing modularity, GenLouvain shows competitive performance. However, GenLouvain only considers the topology of the multiplex network. HMNE can capture fine-grained semantic information by preserving the dependence property between node attributes and the topology of each layer. Compared with other algorithms, it is verified in the shared community detection task that our model can preserve the global mesoscale information of the multiplex network more effectively. We further validate that our model can more fully consider multiple properties of networks. The execution time of MV-ACM is more than 24 hours, so it does not show the final results on Celegans and Ack-co-author datasets. In general, the results of cross-domain link prediction and shared community detection tasks prove the effectiveness of our model. For the cross-domain link prediction task, the graph convolution-deconvolution component of HMNE guarantees that our model can save high-order proximity information. When there is a lack of available information within the layer, the interlayer dependence component of HMNE can provide more abundant information. For the shared community detection task, the component preserving dependence between node attributes and the topology of the layer where the node is located can obtain more fine-grained semantic information related to the layer's topology by disentangling the original attribute information.

5.6. Performance Analysis. In this section, we analyze the results of parameter analysis experiments on the CKM dataset that affect the performance of the model, mainly (1) the impact of the number of convolution (deconvolution) neural network layers on the performance of our model and (2) the impact of the embedding dimension on the performance of HMNE.

5.6.1. Effect of the Neural Layers' Number. It can be seen from the illustration in Figures 4(a) and 4(b) that HMNE can avoid the smooth transition problem caused by the increase of the number of convolution layers. For AUC and AMI scores, it clearly reveals that the performance of HMNE first increases with the increase of the number of network layers

Table 4: The ARI and NMI scores' performance of our HMNE and baseline methods on the CKM multiplex network dataset.

Algorithm	ARI	NMI
OhmNet	0.7920	0.7885
PMNE(n)	0.1733	0.1574
PMNE(r)	0.0376	0.0228
PMNE(c)	0.1582	0.1679
MNE	0.1504	0.1550
MELL	0.1728	0.1805
MV-ACM	0.8942	0.7903
GATNE-T	0.8221	0.8196
DMGI	0.8507	0.8519
GenLouvain	0.9750	0.9742
HMNE	**0.9790**	**0.9771**

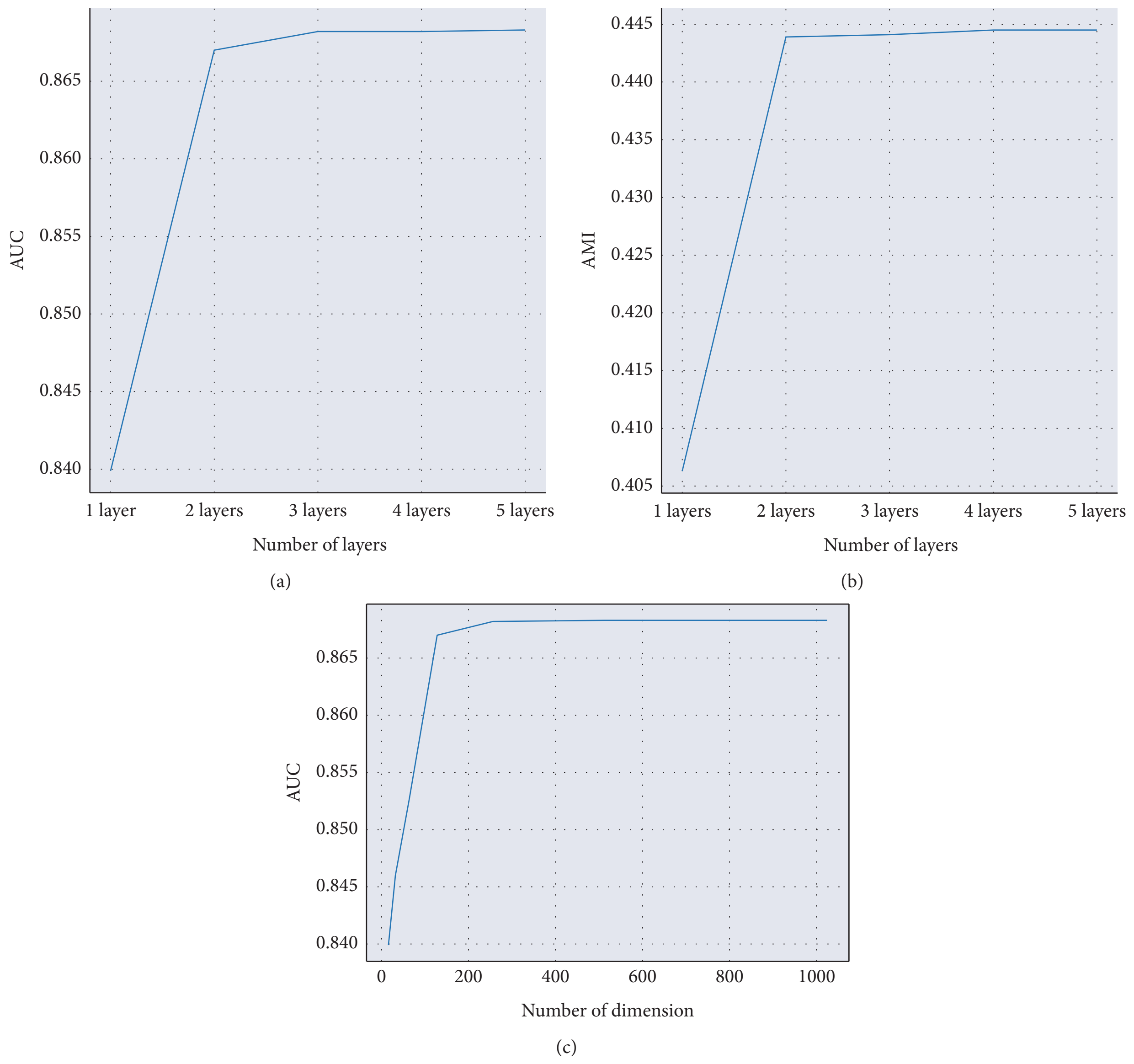
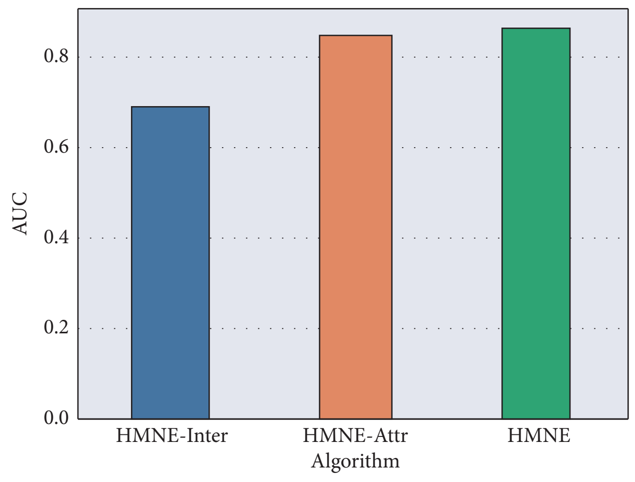
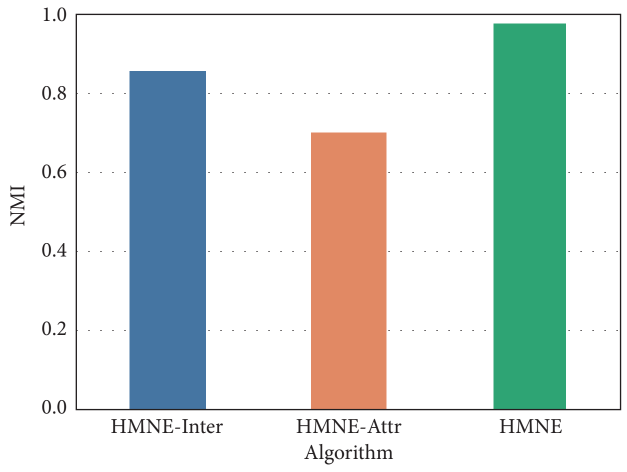


Figure 4: Impact of different experimental settings on our model's performance in the link prediction task: (a) the effect of layer's number on the AUC value, (b) the effect of layer's number on the AMI value, and (c) the effect of dimension on the AUC value.



(a)



(b)

FIGURE 5: The effectiveness of different dependence properties in cross-domain link prediction and shared community detection tasks: (a) the effectiveness comparison in cross-domain link prediction; (b) the effectiveness comparison in the shared community detection task.

and then tends to stabilize. In other words, HMNE does not appear to be oversmoothing as the number of layers increases like other methods [56] based on graph neural networks. Therefore, our proposed HMNE can not only preserve the high-level proximity information of nodes but also avoid oversmoothing problems caused by stacking multiple neural layers.

5.6.2. Effect of the Embedding Dimension. Figure 4(c) illustrates that AUC scores of HMNE also first increase with the increase of the number of embedding dimensions and then tend to stabilize. When the embedding dimension reaches a certain level, HMNE can capture enough key information. In a certain embedding range, node embedding already contains most of the important information that is needed by some tasks. If the embedding dimension continues to increase, it will learn higher-order or more abstract information. Therefore, its performance can show a certain stable state in an interval. In this state, owing to that HMNE has similar self-supervised and autocoder structure, we believe that, with the further increase of dimensions, the objective function designed by our model will purify the original information, filter some meaningless and redundant information, and preserve fine-grained features. Therefore, as the dimension increases, the performance of the model will not show an increasing trend again in a certain dimension range.

5.7. Ablation Experiment. In this section, we will verify the effectiveness of the two properties separately by ablating the constraints of the corresponding loss function from HMNE. (1) HMNE-Inter: to verify the effect of the interlayer dependence property on HMNE, we only ablate loss function equation (17). (2) HMNE-Attr: to verify the dependence between node attributes and the topology of each layer on HMNE, we only ablate loss function equation (19). The experimental results are shown in Figure 5.

5.7.1. The Effectiveness of the Interlayer Dependence Property. As can be seen from Figure 5(a), the interlayer dependence property is critical for link prediction tasks. After removing loss function equation (17) (called HMNE-Inter), the performance of HMNE in the cross-domain link prediction task decreases more significantly than the decrease in the community detection task. The reason is that the structure information of other layers provides effective complementary information for the node pair prediction of the target layer.

5.7.2. The Effectiveness of Dependence between Node Attributes and the Topology of Each Layer. After removing loss function equation (19) (called HMNE-Attr), Figure 5(b) illustrates that HMNE-Attr decreases significantly in the community detection task. In the shared community detection task, we believe the performance of HMNE is more dependent on the attribute information of the node. However, in the link prediction task, the information provided by the dependence between node attributes and the topology of each layer is limited.

6. Conclusion

In this paper, we propose an unsupervised node embedding model for multiplex networks, called HMNE. HMNE first addresses the problem of preserving of high-order proximity information of nodes through the symmetric graph convolution-deconvolution component (SGCD). SGCD utilizes the designed graph deconvolution component (GDC) to reconstruct the input of the graph convolution component (GCC) with multiple graph convolution neural layers. It can effectively avoid the oversmoothing problem. Secondly, HMNE preserves the interlayer dependence property with interlayer complementary information of multiplex networks by our designed interlayer dependence component. When there is a lack of available information within the layer, the interlayer dependence component of HMNE can provide more abundant information from other layers (e.g., cross-domain link prediction scenario). Finally, HMNE preserves the dependence between the node attributes and the topology of each layer through disentangled representation of attributes of nodes. It enables HMNE to have more fine-grained attributes with different semantic information of nodes associated with each layer structure. The final representation of nodes with fine-grained attribute information can perform better in downstream tasks (e.g., shared community detection scenario). Systematical experiments on six real-world networks show the excellent performance of HMNE on two downstream tasks compared with the state-of-the-art baselines. Experiments on large-scale network data based on HMNE will be our future research focus.

Data Availability

These compared methods and our code required for replicating reported results are available at https://github.com/Brian-ning/HMNE/. The public datasets can also be downloaded from https://comunelab.fbk.eu/data.php.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported by the National Key Research and Development Program of China (2018YFC0831500), the National Natural Science Foundation of China (Grant no. 61972047), and the NSFC-General Technology Basic Research Joint Funds (Grant no. U1936220).

References

[1] X. Zhu, J. Ma, X. Su et al., “Information spreading on weighted multiplex social network,” *Complexity*, vol. 2019, Article ID 5920187, 15 pages, 2019.

[2] L. Gao, H. Yang, J. Wu, C. Zhou, W. Lu, and Y. Hu, “Recommendation with multi-source heterogeneous information,” in *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence IJCAI-18*, pp. 3378–3384, Stockholm, Sweden, July 2018.

[3] A. Ghavasieh and M. De Domenico, “Enhancing transport properties in interconnected systems without altering their structure,” *Physical Review Research*, vol. 2, no. 1, 2020.

[4] S. Choobdar, M. E. Ahsen, M. E. Ahsen et al., “Assessment of network module identification across complex diseases,” *Nature Methods*, vol. 16, no. 9, pp. 843–852, 2019.

[5] J. Zhang and S. Y. Philip, *Broad Learning through Fusions: Broad Learning Introduction*, Springer, Berlin, Germany, 2019.

[6] M. E. J. Newman, “Network structure from rich but noisy data,” *Nature Physics*, vol. 14, no. 6, pp. 542–545, 2018.

[7] Y. Zhang, J. Wu, C. Zhou, Z. Cai, J. Yang, and P. S. Yu, “Multi-view fusion with extreme learning machine for clustering,” *ACM Transactions on Intelligent Systems and Technology*, vol. 10, no. 5, pp. 1–23, 2019.

[8] M. Kivelä, A. Arenas, M. Barthelemy, J. P. Gleeson, Y. Moreno, and M. A. Porter, “Multilayer networks,” *Journal of Complex Networks*, vol. 2, no. 3, pp. 203–271, 2014.

[9] Y. Cen, X. Zou, J. Zhang, H. Yang, J. Zhou, and J. Tang, “Representation learning for attributed multiplex heterogeneous network,” in *Proceedings of the 25th ACM SIGKDD International Conference*, pp. 1358–1368, ACM, Anchorage, AK, USA, August 2019.

[10] C. Park, D. Kim, J. Han, and H. Yu, “Unsupervised attributed multiplex network embedding,” in *Proceedings of the Thirty-Fourth AAAI Conference on Artificial intelligence (AAAI)*, pp. 1–8, New York, NY, USA, February 2020.

[11] V. Gligorijevic, Y. Panagakis, and S. Zafeiriou, “Non-negative matrix factorizations for multiplex network analysis,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 41, no. 4, pp. 928–940, 2019.

[12] Y. Shi, F. Han, X. He, C. Yang, J. Luo, and J. Han, “mvn2vec: preservation and collaboration in multi-view network embedding,” *Knowledge-Based Systems*, 2020.

[13] C. Park, J. Han, and H. Yu, “Deep multiplex graph infomax: attentive multiplex network embedding using global information,” *Knowledge-Based Systems*, vol. 197, Article ID 105861, 2020.

[14] Y. Ouyang, B. Guo, X. Tang, X. He, J. Xiong, and Z. Yu, “Learning cross-domain representation with multi-graph neural network,” *Machine Learning*, 2019.

[15] L. Yu, S. Pei, C. Zhang et al., “Self-supervised smoothing graph neural networks,” 2020, http://arxiv.org/abs/2009.00934.

[16] F. Battiston, V. Nicosia, and V. Latora, “The new challenges of multiplex networks: measures and models,” *The European Physical Journal Special Topics*, vol. 226, no. 3, pp. 401–416, 2017.

[17] W. Liu, P.-Y. Chen, S. Yeung, T. Suzumura, and L. Chen, “Principled multilayer network embedding,” in *Proceedings of the 2017 IEEE International Conference on Data Mining Workshops (ICDMW)*, pp. 134–141, IEEE, New Orleans, LA, USA, November 2017.

[18] M. Zitnik and J. Leskovec, “Predicting multicellular function through multi-layer tissue networks,” *Bioinformatics*, vol. 33, no. 14, pp. i190–i198, 2017.

[19] P. Zhou, L. Du, X. Li, Y.-D. Shen, and Y. Qian, “Unsupervised feature selection with adaptive multiple graph learning,” *Pattern Recognition*, vol. 105, pp. 107375–107389, 2020.

[20] M. S. Schlichtkrull, T. N. Kipf, P. Bloem, R. Van Den Berg, I. Titov, and M. Welling, “Modeling relational data with graph convolutional networks,” in *Proceedings of the Semantic Web—15th International Conference (ESWC) 2018*, vol. 10843, pp. 593–607, Springer, Heraklion, Greece, June 2018.

[21] S. Vashishth, S. Sanyal, V. Nitin, and P. Talukdar, “Composition-based multi-relational graph convolutional networks,” in *Proceedings of the 8th International Conference on Learning Representations (ICLR)*, Addis Ababa, Ethiopia, April 2020.

[22] D. Yu, Y. Yang, R. Zhang, and Y. Wu, “Generalized multi-relational graph convolution network,” 2020, http://arxiv.org/abs/2006.07331.

[23] J. Li, C. Chen, H. Tong, and H. Liu, “Multi-layered network embedding,” in *Proceedings of the 2018 SIAM International Conference on Data Mining*, pp. 684–692, SIAM, San Diego, CA, USA, May 2018.

[24] H. Zhang, L. Qiu, L. Yi, and Y. Song, “Scalable multiplex network embedding,” in *Proceedings of the International Joint Conferences on Artificial Intelligence (IJCAI)*, vol. 18, pp. 3082–3088, Stockholm, Sweden, July 2018.

[25] Y. Ma, Z. Ren, Z. Jiang, J. Tang, and D. Yin, “Multi-dimensional network embedding with hierarchical structure,” in *Proceedings of the Eleventh ACM International Conference*

on Web Search and Data Mining, pp. 387–395, ACM, Los Angeles, CA, USA, February 2018.

[26] R. Matsuno and T. Murata, "Mell: effective embedding method for multiplex networks," in *Proceedings of the Companion the Web Conference (WWW)*, pp. 1261–1268, International World Wide Web Conferences Steering Committee, Lyon, France, April 2018.

[27] A. Mahmoudi, M. R. Yaakub, and A. A. Bakar, "The relationship between online social network ties and user attributes," *ACM Transactions on Knowledge Discovery from Data*, vol. 13, no. 3, pp. 15–26, 2021.

[28] S. Ali, M. Shakeel, I. Khan, S. Faizullah, and M. Khan, "Predicting attributes of nodes using network structure," *ACM Transactions on Intelligent Systems and Technology*, vol. 12, no. 2, pp. 1–23, 2020.

[29] J. Li, R. Guo, C. Liu, and H. Liu, "Adaptive unsupervised feature selection on attributed networks," in *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 92–100, Anchorage, AK, USA, August 2019.

[30] P. Velickovic, W. Fedus, W. L. Hamilton, P. Lio, Y. Bengio, and R. D. Hjelm, "Deep graph infomax," in *Proceedings of the International Conference on Learning Representations*, New Orleans, LA, USA, May 2019.

[31] C. Shi, Y. Li, J. Zhang, Y. Sun, and S. Y. Philip, "A survey of heterogeneous information network analysis," *IEEE Transactions on Knowledge and Data Engineering*, vol. 29, no. 1, pp. 17–37, 2016.

[32] P. Bryan, R. Al-Rfou, and S. Skiena, "Deepwalk: online learning of social representations," in *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 701–710, ACM, New York, NY, USA, August 2014.

[33] A. Grover and J. Leskovec, "node2vec: scalable feature learning for networks," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 855–864, ACM, San Francisco, CA, USA, August 2016.

[34] W. Gu, L. Gong, X. Lou, and J. Zhang, "The hidden flow structure and metric space of network embedding algorithms based on random walks," *Scientific Reports*, vol. 7, no. 1, pp. 13114–13125, 2017.

[35] Y. Dong, N. V. Chawla, A. Swami, Y. Dong, N. V. Chawla, and A. Swami, "metapath2vec: scalable representation learning for heterogeneous networks," in *Proceedings of the ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 135–144, Halifax, Canada, August 2017.

[36] L. F. R. Ribeiro, P. H. P. Saverese, and D. R. Figueiredo, "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, ACM, Halifax, Canada, August 2017.

[37] T. N. Kipf and M. Welling, "Variational graph auto-encoders," in *Proceedings of the NeurIPS Workshop on Bayesian Deep Learning (NeurIPS-16 BDL)*, Barcelona, Spain, December 2016.

[38] W. Hamilton, Z. Ying, and J. Leskovec, "Inductive representation learning on large graphs," in *Advances in Neural Information Processing Systems*, pp. 1024–1034, IEEE, New York, NY, USA, 2017.

[39] X. Luo and H. H. Zhuo, "Bigsage: unsupervised inductive representation learning of graph via bi-attended sampling and global-biased aggregating," 2019.

[40] Y. Xiao, D. Chen, S. Wei, Q. Li, H. Wang, and M. Xu, "Rumor propagation dynamic model based on evolutionary game and anti-rumor," *Nonlinear Dynamics*, vol. 95, no. 1, pp. 523–539, 2019.

[41] Y. Xiao, W. Li, S. Qiang, Q. Li, H. Xiao, and Y. Liu, "A rumor & anti-rumor propagation model based on data enhancement and evolutionary game," *IEEE Transactions on Emerging Topics in Computing*, p. 1, 2020.

[42] Y. Xiao, Q. Yang, C. Sang, and Y. Liu, "Rumor diffusion model based on representation learning and anti-rumor," *IEEE Transactions on Network and Service Management*, vol. 17, no. 3, pp. 1910–1923, 2020.

[43] T. Baltrušaitis, C. Ahuja, and L.-P. Morency, "Multimodal machine learning: a survey and taxonomy," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 41, no. 2, pp. 423–443, 2018.

[44] N. Ning, F. Long, C. Wang, Y. Zhang, Y. Yang, and B. Wu, "Nonlinear structural fusion for multiplex network," *Complexity*, vol. 2020, Article ID 7041564, 17 pages, 2020.

[45] K. Zhao, T. Bai, B. Wu et al., "Deep adversarial completion for sparse heterogeneous information network embedding," in *Proceedings of the Web Conference 2020*, pp. 508–518, Taipei, Taiwan, April 2020.

[46] W. Yuan, K. He, C. Shi et al., "Multi-view network embedding with node similarity ensemble," *World Wide Web*, vol. 23, pp. 2699–2714, 2020.

[47] M. Qu, J. Tang, J. Shang, X. Ren, M. Zhang, and J. Han, "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, Singapore, November 2017.

[48] A. Zhiyuli, X. Liang, and Y. Chen, "Hsem: highly scalable node embedding for link prediction in very large-scale social networks," *World Wide Web*, vol. 22, pp. 2799–2824, 2019.

[49] Y. Sun, S. Wang, T.-Yu Hsieh, X. Tang, and V. Honavar, "Megan: a generative adversarial network for multi-view network embedding," 2019, http://arxiv.org/abs/1909.01084.

[50] H. Wei, Z. Pan, G. Hu et al., "Attributed network representation learning via deepwalk," *Intelligent Data Analysis*, vol. 23, no. 4, pp. 877–893, 2019.

[51] X. Chu, X. Fan, D. Yao, Z. Zhu, J. Huang, and J. Bi, "Cross-network embedding for multi-network alignment," in *Proceedings of the World Wide Web Conference*, pp. 273–284, San Francisco, CA, USA, May 2019.

[52] T. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," 2017, http://arxiv.org/abs/1609.02907.

[53] D. Hjelm, A. Fedorov, S. Lavoie-Marchildon et al., "Learning deep representations by mutual information estimation and maximization," in *Proceedings of the International Conference for Learning Representations 2019*, New Orleans, LA, USA, May 2019.

[54] D. Hristova, A. Noulas, C. Brown, M. Musolesi, and C. Mascolo, "A multilayer approach to multiplexity and link prediction in online geo-social networks," *EPJ Data Science*, vol. 5, no. 24, pp. 1–17, 2016.

[55] L. G. S. Jeub, M. Bazzi, I. S. Jutla, and P. J. Mucha, "A generalized Louvain method for community detection implemented," 2011–2019, https://github.com/GenLouvain/GenLouvain.

[56] C. Cai and Y. Wang, "A note on over-smoothing for graph neural networks," 2020, http://arxiv.org/abs/2006.13318.