Inductive Link Prediction via Interactive Learning Across Relations in Multiplex Networks

Mengzhou Gao, Pengfei Jiao, *Member, IEEE*, Ruili Lu, Huaming Wu, *Senior Member, IEEE*, Yinghui Wang, and Zhidong Zhao, *Member, IEEE*

***Abstract*— Network embedding is an important class of link prediction methods, which can use the distance between learned low-dimensional node representations to characterize the similarity between nodes. Traditional network embedding methods focus on single-layer networks, while in reality, a large part of complex networks are not isolated, but interdependent and interrelated, forming multiplex complex networks. Also, how to effectively exploit layer correlations in multiplex networks to learn more robust and valuable representations, to improve link prediction performance, has been a hot research topic in the field of complex network analysis. However, previous studies mainly focus on inferring intralinks in each layer of complex networks or anchor links among layers. Another issue that has not been discussed is how to predict potential links or reconstruct the network in unobserved relations based on existing multiplex networks. To this issue, we define a novel inductive link prediction problem in multiplex networks, in which most existing multichannel network embedding methods fail to solve. This is either because they only emphasize the specific structure information of an individual layer or only capture the common information for all layers. To effectively address this problem, we propose a novel embedding method termed interactive learning across relations (ILAR), to capture and fully exploit the multiple relations and complex layer correlations in multiplex networks. We leverage two convolutional modules and ILAR to capture the sufficient complementary and correlations in multiplex networks. Moreover, during interactive learning, a disparity constraint is introduced, which enforces the features encoded from two convolutional modules to be different and prevents information redundancy. Finally, the extensive experiments in several real-world datasets show that our model can significantly outperform the existing state-of-the-art network embedding methods on the novel link prediction problem in multiplex networks.**

***Index Terms*— Layer correlations, link prediction, multiplex network, network embedding.**

NOMENCLATURE

L	Numbers of layers in the known multiplex network.
G^l	lth layer network in multiplex network.
G^t	Unknown target network to be predicted.
V	Set of nodes in the multiplex network.
$\boldsymbol{A}^l$	Adjacency matrix of G^l.
$\boldsymbol{D}^l$	Degree matrix of G^l.
$\hat{\boldsymbol{A}}$	Reconstruction adjacency matrix.
$\boldsymbol{Z}_s^l$	Specific representation for nodes in the lth layer.
$\boldsymbol{Z}_c^l$	Complementary representation for nodes in the lth layer.
$\boldsymbol{Z}^r$	Nodes representation of the remaining networks for anchor network G^l.
$\boldsymbol{Z}^l$	General representation for nodes in the lth layer.
$\boldsymbol{Z}$	Final representation for nodes to predict the target network.
$\boldsymbol{W}_s^l$	Project matrix of specific convolutional module in the lth layer.
$\boldsymbol{W}_c^l$	Project matrix of complementary convolutional module in the lth layer.

I. INTRODUCTION

WITH the massive growth of social media usage [1], a large collection of multimodal data is generated, which poses great challenges for data analysis [2]. Complex network is a ubiquitous data structure that captures connections between individual entities, where nodes represent entities and edges encode the interactions between these nodes. It can be employed in a wide range of domains, e.g., social networks [3], citation networks [4], biological protein–protein networks [5], and vehicular ad hoc networks (VANETs) [6]. Link prediction is a fundamental problem in analyzing and mining complex networks, with the goal of predicting lost or potential links based on known network information [7]. It is critical for understanding the evolutionary process and exploration of complex networks and helps to complement partially observed networks. Link prediction is the basis for online marketing, e-commerce services, and various recommendation systems.

In recent years, link prediction in complex networks has attracted a lot of attention and a large number of methods have been proposed gradually. Network embedding [8] learns the low-dimensional vector representation of the nodes while maximizing the structural information of the original network and can simply characterize the similarity of the node pairs by the distance between the node representations. Therefore, network

Manuscript received January 5, 2022; revised April 10, 2022; accepted May 18, 2022. This work was supported in part by National Key Technology Research and Development Program of China under Grant 2019YFB2102100; in part by the National Natural Science Foundation of China under Grant 62003120, Grant 61902278, and Grant 62071327; in part by the Zhejiang Provincial Natural Science Foundation of China under Grant LQ20F030012; and in part by the Zhejiang Provincial Key Technology Research and Development Program under Grant 2019C03134. *(Corresponding author: Huaming Wu.)*

Mengzhou Gao, Pengfei Jiao, and Zhidong Zhao are with the School of Cyberspace, Hangzhou Dianzi University, Hangzhou 310018, China (e-mail: mzgao@hdu.edu.cn; pjiao@hdu.edu.cn; zhaozd@hdu.edu.cn).

Ruili Lu is with the Tianjin International Engineering Institute, Tianjin University, Tianjin 300072, China (e-mail: luruili@tju.edu.cn).

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

Yinghui Wang is with the College of Intelligence and Computing, Tianjin University, Tianjin 300350, China (e-mail: wangyinghui@tju.edu.cn).

Digital Object Identifier 10.1109/TCSS.2022.3176928

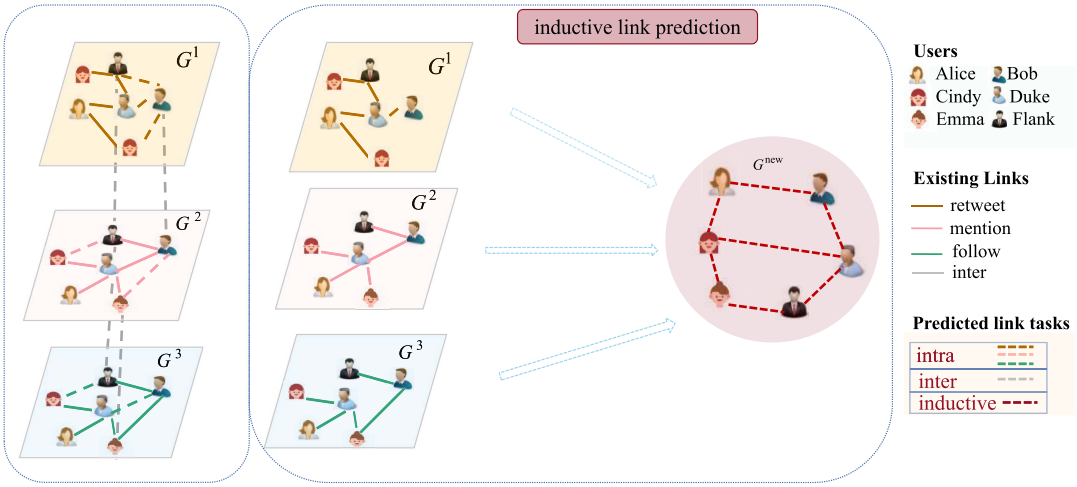

Fig. 1. Illustrative example of link prediction in multiplex networks. It is constructed from the richly structured data of Twitter. G^1, G^2, and G^3 represent different kinds of relation networks, namely, retweet, mention, and follow, respectively. Solid lines of the corresponding color are the existing connections between them, while dashed lines represent the links to be predicted.

embedding methods can easily and efficiently take link prediction and are widely used in link prediction tasks. Although existing network embedding methods have been extensively researched with good results on link prediction tasks, most of the previous research has focused on single-layer complex networks, i.e., complex networks consisting of only one type of interaction behavior between nodes. In reality, however, complex network systems are often consisting of multiple interacting relationships, e.g., the social network may represent a network of friend's relationships, a network of family relationships, or a network of cooperative relationships. Such complex networks formed by the interaction of multiple relationship types between the same entities can naturally be described as multiplex networks [9], [10].

Link prediction based on multiplex network embedding has become an important research direction. Unlike single-layer networks that only take link prediction in one network, there are two main types of link prediction tasks in multiplex networks. The first type [11]–[13] is inferring links for intralayer, that is, inferring the potential links in every single layer using multiple networks structure. Some work extends the link to interlayer [14]–[18], commonly referred to as network alignment. Network alignment establishes the correspondence between the nodes from two networks and predicts the anchor links. However, there is often the situation in real life, e.g., the data of a network are completely lost for technical reasons and need to be recovered or the behavior of a gene in an entirely new type of interaction is inferred based on the known behavior of the type of relationship. Predicting the potential links for such a network with no topology at all is very practical, and yet, it has not been discussed. In this article, we propose the novel link prediction problem and define it as inductive link prediction in multiplex networks, which exploits the existing multiple types of relational data to predict links in an unobserved relation. We give a specific example in Fig. 1, and there are several kinds of relations such as retweet, mention, and follow in the Twitter social network. The first type is forecasting links at each relation displayed as the dashed line as the color corresponding to the located network and the network alignment predicts interlayer links finding that the correspondence nodes between two networks are shown as a gray dashed line. Unlike these two tasks, the inductive link prediction is to predict an unobserved relation, friend in life, which is a brand new relationship without any known links.

Although existing work demonstrates promising performance in the two conventional link prediction tasks, they face the following challenges when applied to the new issue we raised. First, the methods designed for the first type of link prediction emphasize the structure specific to the single layer, although they incorporate the correlations among multiple layers. The specific structure may be useless or even negatively impacting in predicting links in a new relation. Second, the alignment models focus on two-layer networks and predict links to the same node in both networks, rather than infer links between different nodes. Third, the multiplex embedding methods learn general representations of nodes in all layers and are mainly used for node classification and clustering but can also be used to predict links in a new relation. However, the type defining common vectors [11], [19] shared by all layers may lose the information essential for a part of layers, while the type aggregating the interactions of all layers [12], [20] encoding the specific information for individual layers but not useful for other layers and redundant information included in other layers in some cases. To overcome the limitations, we are seeking an approach that can facilitate the collaboration of different layers and capture sufficient complementary information and eliminate unique information for specific layers but the noise for the target network.

In this article, we propose a novel method termed interactive learning across relations (ILAR), to inductively predict links by exploiting sufficient complementary information and correlations in multiplex networks. In the proposed method, we design two convolutional modules, i.e., a specific convolutional module and a complementary convolutional module, to encode different features that capture the specific

information from a single layer and complementary information from other layers, respectively. Also, for the complementary convolutional module, we apply a parameter sharing strategy to build the correlations of multiple layers. Moreover, we design a disparity constraint to ensure that the features extracted are different.

The main contributions of this article can be summarized as follows.

1) We summarize two types of link prediction tasks of existing multiplex network methods and raise a new issue to be settled, that is how to inductive link prediction on an unobserved relation based on the existing multiple types of relational data.
2) We propose a novel ILAR method to solve the novel inductive link prediction problem by fully capturing the complementary and useful information in multiplex networks. Also, we design two convolutional modules, i.e., specific convolutional module and complementary convolutional module, and disparity constraints to encode the complementary information.
3) Extensive experiments on a series of real multiplex networks clearly demonstrate that our proposed model outperforms the state-of-the-art network embedding methods in the novel link prediction problem.

The rest of this article is structured as follows. Section II reviews related work on network embedding methods, including single-layer network embedding methods, multichannel network embedding methods, and heterogeneous network embedding methods. In Sections III and IV, we present the problem formulation and the proposed method in detail, respectively. Extensive experiments are conducted and analyzed in Section V. Finally, Section VI concludes this article.

II. Related Work

A. Single-Layer Network Embedding Methods

Single-layer network embedding methods can be roughly divided into three categories: methods based on matrix decomposition, methods based on random walk-based, and methods based on deep learning.

Most of the early methods learned node embedding by taking the form of matrix decomposition, where they used a matrix representation of the features of the network and then factorized the matrix to obtain the low-dimensional vector. Isomap [21] first used the K-nearest neighbor (KNN) algorithm to construct a neighborhood graph and then used this graph by computing the shortest path between node pairs to obtain the distance matrix of the graphs. It then applied the classical multidimensional scaling (MDS) algorithm to the distance matrix to obtain the coordinate vectors of the nodes. The local linear embedding (LLE) method [22] eliminated the need to estimate pairwise distances between separated nodes, which assumed that each node and its neighbors lie at or near a local linear patch of the principal line and then constructed a neighborhood-preserving mapping based on local linear reconstruction. LE [23] also constructed a graph from using ϵ neighbors or KNN and then used a heat kernel to select weights between pairs of nodes in the graph.

However, these methods based on matrix decomposition are difficult to use in large networks due to the high computational and time-consuming. With the great success of word2vec [24] in the field of natural language processing (NLP), scholars have proposed methods based on random walks to learn node representations. These models treat a node in a network as a word and use random walks to obtain a sequence of contexts and then use the skip-gram [24] algorithm to model the probability of nodes within each local window in the random walk sequence. Inspired by the fact that the distribution of nodes in short sequences generated by random walks is similar to that in natural language, DeepWalk [25] took a depth-first sampling strategy to perform truncated random walks to generate wandering sequences and then applied the skip-gram algorithm to learn node embedding. node2vec [26] designed a second-order random walk strategy to sample the neighborhood, defining two parameters p and q that adjust between depth-first sampling and breadth-first sampling during the random walk. LINE [27] considered the first-order similarity and the second-order similarity of nodes.

Although these shallow models perform well on many tasks, they have a limited ability to discover underlying relationships in complex data. In recent years, deep learning-based network embedding methods have emerged as the main key technology, primarily extending graph neural networks (GNNs) to complex networks to learn complex nonlinear features in network structures. SDNE [28] and DNGR [29] are based on a deep autoencoder framework to capture highly nonlinear network structures. However, both of them use the global neighborhood of each node as input, which is computationally expensive. Graph convolutional networks (GCNs) are solved by defining a convolution operator in the network [30]. GCNs iteratively aggregate the representations of a node's neighbors and use the obtained representation and its representation in the previous iteration to obtain a new node representation, with multiple iterations allowing the learned node representation to characterize the global neighborhood. GCNs can be broadly classified into two categories based on the spectral domain [30]–[32] and the spatial domain [33]–[35]. From the spectral domain perspective, GCN applied Fourier transform to the graph signal and filter and then applies the convolution theorem to transform the convolution operation in the time domain to a product operation in the spectral domain. The key difference between the different methods is the different choice of filters. From the spatial domain perspective, GCNs can be seen as an operation that aggregates feature information from a node's neighborhood.

B. Multiplex Network Embedding Methods

The above embedding methods, although proven to be effective, are mainly used to deal with single-layer networks and are not applicable to learn node embedding in multiplex networks. With the increasing prevalence of multiplex networks in real-world applications, multiplex network embedding methods have been proposed, which can be divided into methods that learn specific representations for each single layer and methods that learn general representations for all layers.

Some multiplex network embedding methods learn specific representation for every single layer and usually applied to link prediction for intralayer of multiplex networks. MNE [11] and CrossMNA [19] introduce a common vector for anchor nodes to capture the shared information across all the relations. MNE [11] represents the embedding vector in each relation through the combination of common vector and low-dimensional vectors for each relation, while CrossMNA [19] combines the common vector and network vector for each relation, which is introduced to extract the semantic meaning of the single network. MELL [36] enforces embedding vectors in each layer for the same node to be close to the average embedding matrix among layers in order to share the common information. The asp2vec [37] dynamically assigns relation for each node based on its local context and models the interactions among layers in terms of both relatedness and diversity. LISCNE [38] takes advantage of the common and local features in multiplex networks and exploited layer similarity at the same time. DMGE [39] first learned consistent node embeddings through shared GCN, and then, based on the consistent node embeddings, it defined specific graph convolutional layers for each complex network to learn the node embeddings of each layer. MANE [40] divided the node pairs in multiplex networks into intraview pairs, cross-view, intranode pairs, and cross-view, cross-node pairs, and jointly trained the three types of node pairs to learn the corresponding node representations for each layer of the network.

Some multiplex network embedding methods learn consistent and general representation and often applied to node classification and clustering. MNE [11] and CrossMNA [19] learn the general representations with the definition of the common vector and intervector, and CrossMNA is also a method applied to network alignment. MVE [41] provides a robust representation by promoting the collaboration of different views and different weights were assigned to views during voting. HMNE [42] extracts the cross-layer neighborhood of a node to learn the unified embedding by applying a heuristic 3-D interactive walk technique. mGCN [12] is a multidimensional convolutional neural network method for modeling intra- and cross-dimensional interactions in multiplex networks. DMGI [20] extends deep graph infomax (DGI) [35] for attributed multiplex networks and jointly integrates the relation-type specific node embeddings by minimizing the disagreements among them. SSDCM [43] leverages a novel cluster-aware, node-contextualized global graph summary generation strategy and then modeled the nodes and cluster structures by maximizing the mutual information between local nodewise patch representations and label correlated structure-aware global graph representations. HDGI [44] first designs a joint supervision signal containing both extrinsic and intrinsic mutual information through high-order mutual information and then introduces a high-order deep infomax (HDI) to optimize the proposed supervision signal.

C. Heterogeneous Network Embedding Methods

A heterogeneous network consists of multiple types of nodes and edges, while the multiplex network can be viewed as a special form with only one type of node. metapath2vec [45] is a well-known model, which extends the skip-gram model to model the heterogeneous neighborhood of a node based on the meta-paths. RHINE [46] divides structural characteristics of relations in HINs into affiliation relations (ARs) and one-centered-by-another structures and interaction relations (IRs) with peer-to-peer structures. R-GCN [47] introduces the GCN framework that can be to model multiple types of relations. HeGAN [48] employs adversarial learning for HIN embedding in order to model the rich semantics on HINs. GATNE [49] proposed a general framework of MNE [11] to attribute multiplex heterogeneous networks. Although many recent studies applied GNNs to heterogeneous networks [50], they are mostly semisupervised methods and not adapted to our tasks.

III. Problem Formulation

Our goal in this article is to take inductive link prediction based on the learned node embeddings from the existing multiplex network data. In this section, we will introduce some background knowledge and definitions of multiplex networks, followed by the problem formulation.

Definition 1 (Multiplex Network): A multiplex network $MG = (G^1, G^2, \ldots, G^L)$ consists of a set of N nodes $V = (v_1, v_2, \ldots, v_N)$ and L relations, and interactions in each relation form a set of edges $(\xi^1, \xi^2, \ldots, \xi^L)$, where $G^l = (V, \xi^l)$ denotes the lth layer and the topological structure of this layer can be represented by the adjacency matrix $\boldsymbol{A}^l$.

Due to the sparse and high-dimensional format of adjacency matrix, multiplex network embedding has recently attracted great attention to learning the low-dimensional representations for nodes.

Definition 2 (Multiplex Network Embedding): Given a multiplex network with L relations, $MG = (G^1, G^2, \ldots, G^L)$, the type of learning specific representations for nodes in each relation multiplex network embedding aims to learn the robust node representations for each node $v_i \in V$ in each layer $Z_i \in R^{L\times d}$ with $d \ll N$. The type of learning general representations for each node $v_i \in V$ in all relations is to learn the d-dimensional vector $Z_i \in R^d$ with $d \ll N$.

There are not only intralayer interactions but also interlayer interactions in multiplex networks; correspondingly, there are two types of link prediction in multiplex networks, one for intralayer link prediction that is to predict missing or potential edges in each layer of network and one for interlayer link prediction that is commonly referred to as network alignment.

Definition 3 (Intralayer Link Prediction in Multiplex Network): Given a multiplex network $MG = (G^1, G^2, \ldots, G^L)$, intralayer link prediction in multiplex network aims at inferring the missing or potential edges in each relation of $G^l \in MG$, that is, to generate the edge probability function $P(v_i^l, v_j^l) \to [0, 1]$.

Definition 4 (Interlayer Link Prediction in Multiplex Network): It is often referred to as network alignment. Given the network $MG = (G^1, G^2, \ldots, G^L)$, a set of observed anchor nodes V_t, the goal of network alignment task is to discover the unknown or potential anchor nodes among layers of MG,

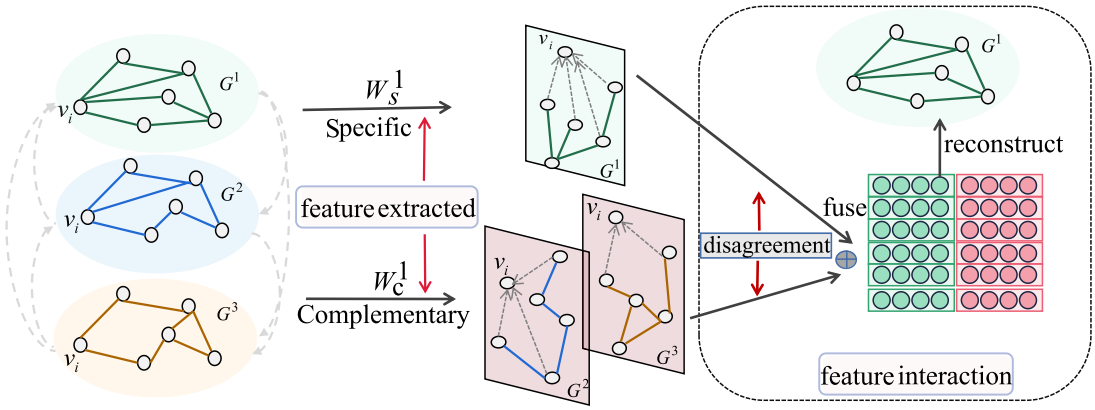

Fig. 2. Schematic of ILAR main architecture: 1) the input is a three-layer multiplex network $G = \{G^1, G^2, G^3\}$ and the multiple relations involved are interrelated; 2) features extracted: specific features for the anchor network and complementary features from other remaining networks are learned by two convolutional modules and parameter sharing strategy is used for the complementary convolutional module; and 3) ILAR: fuse the features encoded by two convolution models to reconstruct the anchor network and disparity constraint is used to enforce features encoding different information for anchor network.

i.e., for a pair of nodes $(v_i^{l_1}, v_j^{l_2})$, where v_i and v_j are in the set of node set V, and to predict whether there is an anchor link between them in networks G^{l_1} and G^{l_2}.

However, it is another highly practical link prediction problem that is to predict possible links for networks with no topology at all, for example, sometimes data for a network are completely lost for technical reasons or the behavior of genes in an entirely new type of interaction is inferred based on the behavior of genes interacting in some known types of relationship. For the novel link prediction problem, we refer to it as inductive link prediction in multiplex networks.

Definition 5 (Inductive Link Prediction in Multiplex Network): Based on the multiplex network $MG = (G^1, G^2, \ldots, G^L)$ with node set V, the inductive link prediction problem is to find a function $P(v_i^t, v_j^t) \rightarrow [0, 1]$ to infer the potential edges in G^t, where $v_i, v_j \in V$ and $G^t \in (G^{t_1}, G^{t_2}, \ldots, G^{t_k})$ are the networks without any topology need to infer the potential edges.

In this article, we aim at solving the problem that takes inductive link prediction in one unobserved network, that is, $k = 1$, and we use G^t to denote the target network to take inductive link prediction. Then, we will introduce some notations we will use in the remaining of this work. The Nomenclature provides a list of major notations in this article.

IV. Proposed Model

In this section, we introduce our proposed approach for inductive link prediction by integrating the network structure of multiple relations in multiplex networks.

A. Overall Structure

When applied to the problem, most existing approaches, e.g., the methods designed for the link prediction in the individual layers and learning general embedding for nodes in all layers, fail to achieve satisfactory results. This is because either some of them emphasize the unique structure information for an individual layer, which may be noise for the target network, or some only capture the common information for all layers, which is insufficient. Thus, we need an approach integrating multiple independent relations to encode sufficient complementary information and eliminate noise information.

The overall process for ILAR is shown in Fig. 2. The core idea is that the topological structures of the anchor network are not only correlated with itself but also with other networks formed by multiple relations. For the given multiplex network, our proposed model iteratively selects one layer as the anchor network, and for each anchor network, we design two different convolutional modules to separately encode specific information for anchor network and complementary information from other remaining networks. Through the interactive learning and fusion of features obtained from the two modules, the representation of the reconstructed anchor network not only fuses the different relational features but also preserves the properties and structures of the original network. The final embedding to inductively predict links is the average value of outputs of complementary convolutional module for each anchor network, rather than the specific module to eliminate the exclusive information.

B. Interactive Learning Across Relations

First, we assume that the anchor network selected is G^l. To capture the specific information for the anchor network as well as the complementary information and correlations from other remaining networks, we introduce two convolutional modules, i.e., the specific convolutional module and the complementary convolutional module. Then, we introduce these two types of modules in detail.

1) Specific Convolutional Module: In order to make full use of all layers in multiplex networks, we iteratively select one layer of the multiplex network as the anchor network. For the anchor network G^l with the adjacency matrix $\boldsymbol{A}^l$ and its degree matrix $\boldsymbol{D}^l$ as topological structures, we utilize the specific convolutional module to capture the unique information in this network. The output $\boldsymbol{Z}_s^l \in \boldsymbol{R}^{N \times d}$ denotes the specific representation.

In this article, we use two-layer GCN [51] as the convolutional module, and through the module, $\boldsymbol{Z}_s^l$ is learned by

$$\boldsymbol{Z}_s^l = \tilde{\boldsymbol{A}}^l \sigma\left(\tilde{\boldsymbol{A}}^l \boldsymbol{I} \boldsymbol{W}_{s(0)}^l\right) \boldsymbol{W}_{s(1)}^l \tag{1}$$

where $\tilde{\boldsymbol{A}}^l = \tilde{\boldsymbol{D}}^{l^{-(1/2)}}(\boldsymbol{A}^l + \boldsymbol{I})\tilde{\boldsymbol{D}}^{l^{-(1/2)}}$ is the symmetrically normalized adjacency matrix. $\boldsymbol{I}$ is the identity matrix and $\tilde{\boldsymbol{D}}^l$ is the renormalized diagonal degree matrix. In (1), $\boldsymbol{I}$ cannot be replaced by the feature matrix of the network, that is, our model is suitable for attribute-free networks, which we will explain after introducing other convolution modules. $\boldsymbol{W}^l_{s(0)}$ and $\boldsymbol{W}^l_{s(1)}$ are the weight matrices. σ is the activation function and we use $\mathrm{ReLU}(\cdot) = \max(0, \cdot)$ in this article.

2) Complementary Convolutional Module: As the structure of the same nodes in different layers is not independent in multiplex network, we introduce a complementary convolutional module. For the remaining networks except for anchor network in the multiplex network, we apply the complementary convolutional module to capture the complementary and useful information for the anchor network. It is also a two-layer GCN, while the weight matrices $\boldsymbol{W}^l_{c(0)}$ and $\boldsymbol{W}^l_{c(1)}$ are shared by all the other remaining networks. This parameter sharing strategy is supposed to learn the most correlated and common information of the remaining networks. We share the same weight matrix for each network G^r of the remaining layers as follows:

$$\boldsymbol{Z}^r = \tilde{\boldsymbol{A}}^r \sigma\left(\tilde{\boldsymbol{A}}^r \boldsymbol{I}\boldsymbol{W}^l_{c(0)}\right)\boldsymbol{W}^l_{c(1)}. \tag{2}$$

We train the link prediction in each layer based on the embedding

$$\hat{\boldsymbol{A}}^{\boldsymbol{r}} = \mathrm{sigmoid}\left(\boldsymbol{Z}^r \boldsymbol{Z}^{r\,T}\right). \tag{3}$$

Also, we train the model by minimizing the negative cross-entropy between the reconstructed adjacency matrix $\hat{\boldsymbol{A}}^{\boldsymbol{r}}$ and the original adjacency matrix $\boldsymbol{A}^r$

$$\mathrm{loss}_r = -\frac{1}{N}\sum_r \sum_{ij} \boldsymbol{A}^r_{ij} \log \hat{\boldsymbol{A}}^r_{ij} + \left(1 - \boldsymbol{A}^r_{ij}\right)\log\left(1 - \hat{\boldsymbol{A}}^r_{ij}\right). \tag{4}$$

With the learned weight matrices, we encode the complementary information for the anchor network G^l

$$\boldsymbol{Z}^l_c = \tilde{\boldsymbol{A}}^l \sigma\left(\tilde{\boldsymbol{A}}^l \boldsymbol{I}\boldsymbol{W}^l_{c(0)}\right)\boldsymbol{W}^l_{c(1)}. \tag{5}$$

3) Interactive Learning Across Relations: After obtaining features based on network structures, embeddings for anchor network will undergo a fusion from dual feature spaces to promote interaction among relations. With the fusing of two features, we get the general embeddings for the anchor network to reconstruct an adjacency matrix

$$\boldsymbol{Z}^l = \boldsymbol{Z}^l_s + \alpha \cdot \boldsymbol{Z}^l_c. \tag{6}$$

Then, we use the embedding $\boldsymbol{Z}^l$ to reconstruct the matrix of anchor network

$$\hat{\boldsymbol{A}}^{\boldsymbol{l}} = \mathrm{sigmoid}\left(\boldsymbol{Z}^l \boldsymbol{Z}^{l\,T}\right). \tag{7}$$

We minimize the sum of reconstruction error of each network by

$$\mathrm{loss}_l = -\frac{1}{N}\sum_l \sum_{ij} \boldsymbol{A}^l_{ij} \log \hat{\boldsymbol{A}}^l_{ij} + \left(1 - \boldsymbol{A}^l_{ij}\right)\log\left(1 - \hat{\boldsymbol{A}}^l_{ij}\right). \tag{8}$$

a) Disparity constraint: The features learned by two different convolutional module, encoding specific features and complementary features for anchor network, help to reconstruct the network. To ensure that the two modules learn different information, we apply a score function $D(\boldsymbol{Z}^l_s, \boldsymbol{Z}^l_c)$ to keep them away from each other. We employ the Hilbert–Schmidt independence criterion (HSIC) [52] to enhance the disparity of their outputs. Formally, the HSIC constraint of $\boldsymbol{Z}^l_s$ and $\boldsymbol{Z}^l_c$ is defined as

$$\begin{aligned}\mathrm{loss}_d &= \sum_l \mathrm{HSIC}\left(\boldsymbol{Z}^l_s, \boldsymbol{Z}^l_c\right) \\ &= (n-1)^{-2} tr(\boldsymbol{R}\boldsymbol{K}_s\boldsymbol{R}\boldsymbol{K}_c)\end{aligned} \tag{9}$$

where $\boldsymbol{K}_s$ and $\boldsymbol{K}_c$ are the Gram matrices with $k_{s,ij} = \boldsymbol{K}_s(\boldsymbol{Z}^i_s \boldsymbol{Z}^j_s)$ and $\boldsymbol{k}_{c,ij} = \boldsymbol{K}_c(\boldsymbol{Z}^i_c \boldsymbol{Z}^j_c)$, $\boldsymbol{R} = \boldsymbol{I} - (1/n)ee^T$, $\boldsymbol{I}$ is an identity matrix, and e is an all-one column vector. In our implementation, we use the inner product kernel function for $\boldsymbol{K}_s$ and $\boldsymbol{K}_c$.

With the disparity constraint of HSIC for $\boldsymbol{Z}^l_s$ and $\boldsymbol{Z}^l_c$, it enforces $\boldsymbol{Z}^l_s$ to learn the unique structure existing in the anchor network and $\boldsymbol{Z}^l_c$ to learn the complementary structure shared by other networks that do not exist in the anchor network. This is also the reason that the identity matrix is input into the convolutional module instead of the network attributes. The two convolutional modules cannot effectively capture different features by propagating the same attributes if the attributes of the network are input.

b) Overall objective function: The total objective function is defined as follows:

$$\mathrm{loss} = \sum_l^L \left(\mathrm{loss}_l + \beta \cdot \sum_{r \neq l}^L \mathrm{loss}_r + \gamma \cdot \mathrm{loss}_d\right) \tag{10}$$

where β is the coefficient that controls the degree of distorting embedded space and γ is the parameter of disparity constraint term. Also, we let $\beta = \gamma = 1.0$ for all experiments. The corresponding pseudocode of the model is summarized in Algorithm 1.

c) Predict links in the target network: We sum the nodes representation of all anchor networks as the final embedding to predict the links in the target network. The final embedding $\boldsymbol{Z} \in R^{n \times d}$ is expressed as follows:

$$\boldsymbol{Z} = \sum_l^L \boldsymbol{Z}^l \tag{11}$$

where L is the number of layers of the multiplex network. We can perform inductive link prediction to predict unseen links in the new relation by utilizing the embedding $\boldsymbol{Z}$

$$p\left(v^t_i, v^t_j\right) = \mathrm{sigmoid}\left(z^T_i z_j\right) \tag{12}$$

where G^t is the target network without any topology requiring to predict potential links and z_i and z_j are the final embedding of nodes v_i and v_j, respectively.

4) Complexity Analysis: In the entire loop of selecting the anchor network to reconstruct the network and learn the complementary features by the convolutional module, the time complexity is $O(L^2N^2)$. To reduce the time complexity,

Algorithm 1 ILAR

Input: The multiplex network $G = \{G^1, G^2, \ldots, G^L\}$ with adjacency matrix, the parameters α, β, γ, d

Output: The adjacent matrix $\hat{\boldsymbol{A}}^{\boldsymbol{t}}$ of the target network G^t

repeat
  foreach *layer l in L* **do**
    $\boldsymbol{Z}_s^l = \tilde{\boldsymbol{A}}^{\boldsymbol{l}} \sigma(\tilde{\boldsymbol{A}}^l \boldsymbol{I} \boldsymbol{W}_{s(0)}^l) \boldsymbol{W}_{s(1)}^l$;
    foreach *layer r in L and* $r \neq l$ **do**
      $\boldsymbol{Z}^r = \tilde{\boldsymbol{A}}^r \sigma(\tilde{\boldsymbol{A}}^r \boldsymbol{I} \boldsymbol{W}_{c(0)}^l) \boldsymbol{W}_{c(1)}^l$;
      % reconstruct adjacent matrix :
      $\tilde{\boldsymbol{A}}^r \leftarrow sigmoid(\boldsymbol{Z}^r \boldsymbol{Z}^{rT})$;
      $loss_r = \sum_r loss(\boldsymbol{A}^r, \hat{\boldsymbol{A}}^r)$;
    end
    $\boldsymbol{Z}_c^l = \tilde{\boldsymbol{A}}^l \sigma(\tilde{\boldsymbol{A}}^l \boldsymbol{I} \boldsymbol{W}_{c(0)}^l) \boldsymbol{W}_{c(1)}^l$;
    $\boldsymbol{Z}^l = \boldsymbol{Z}_s^l + \alpha \cdot \boldsymbol{Z}_c^l$;
    $\hat{\boldsymbol{A}}^{\boldsymbol{l}} = sigmoid(\boldsymbol{Z}^l \boldsymbol{Z}^{lT})$;
    $loss_l = loss(\boldsymbol{A}^l, \hat{\boldsymbol{A}}^l)$;
    % Disparity constraint
    $loss_d = HSIC(\boldsymbol{Z}_s^l, \boldsymbol{Z}_c^l)$;
  end
until *converge*;
$\boldsymbol{Z} = \sum_l^L \boldsymbol{Z}^l$;
return $\hat{\boldsymbol{A}}^{\boldsymbol{t}} = sigmoid(\boldsymbol{Z}\boldsymbol{Z}^T)$;

we take two strategies: sampling negative edges (negative sampling approach [53]) and sampling complementary layers. More specifically, the sparsity of network in real life, the number of negative edges, is very large and can generally be linearly bounded by the number of nodes. It is computationally expensive to consider all the negative edge pairs. Therefore, we randomly sample K nodes that are not connected to node v_i for each $(v_i, v_j) \in \xi^l$ in every network G^l. These K samples are put into the set of negative samples. In this way, the size of negative samples is only k times as large as that of positive samples. Also, the time complexity for reconstructing each network is $O(M)$, and $M = |\xi|^l$. Then, the loss in (8) can be trained like this (see (4) is the same)

$$\text{loss}_l = -\left[\sum_{\left(v_i^l, v_j^l\right) \in \xi^l} \log p\left(v_i^l, v_j^l\right) + \sum_{\left(v_i^l, v_j^l\right) \in \xi_{\text{neg}}^l} \left(1 - \log\left(p\left(v_i^l, v_j^l\right)\right)\right)\right] \tag{13}$$

where ξ_{neg}^l is the sampled negative edge set.

On the other hand, for most multiplex networks, it is not necessary to sample layers because L is usually a relatively small number. For L larger to sample, we randomly select R layers to learn the complementary features for each anchor network instead of all the rest of the networks to reduce the learning time. After implementing the two sampling strategies, the overall time complexity of our model is $O(L(R+1)M)$.

V. Experiment

To verify the effectiveness of our method, we conduct inductive link prediction on four real multiplex networks and compare three types of network embedding methods, including single-layer network embedding methods, multiplex network embedding methods, and heterogeneous network embedding methods.

A. Datasets

Mobile crowdsensing (e.g., a community with people using mobile devices) has emerged as a promising way to collect data [54]. Here, we apply the datasets obtained from CoMuNe lab's web site.[1] We summarize the dataset statistics as listed in Table I.

1) *CKM [55]:* A social multiplex network of physicians in four towns, where three layers represent whom they turn to who they need advice, who they discuss cases with, and who they most often socially. There are 246 nodes and 1551 links.
2) *SACCHPOMB [56], [57]:* A genetic multiplex network of different types of genetic interactions. It has five layers of interactions: direct interactions, physical associations, suppressive genetic interactions, synthetic genetic interactions, and additive genetic interactions. There are 4092 nodes and 63 033 links.
3) *DROSOPHILA [56], [57]:* A genetic multiplex network of different types of genetic interactions, e.g., direct interactions, suppressive genetic interactions, additive genetic interactions, and physical interactions. There are 8215 nodes and 43 210 links.
4) *HOMO [56], [57]:* A genetic multiplex network of different types of genetic interactions, including direct interactions, physical associations, and colocalization. There are 18 222 nodes and 166 904 links.

B. Baseline Methods

We compare with three types of methods, namely, single-layered embedding models, heterogeneous, and multiplex embedding models to test the performance of the proposed model for inductive link prediction.

1) *vgae [58]:* An embedding model for single-layered networks. It was an inference model parameterized by a two-layer GCN.
2) *node2vec [26]:* An embedding model for single-layered networks. node2vec designed a biased random walk and explores diverse neighborhoods to learn richer representations.
3) *MELL [36]:* A multiplex network embedding model learning specific representation for each relation. MELL

[1] https://comunelab.fbk.eu/data.php

TABLE I
STATISTICS OF DATASETS

Dataset	Nodes	Layers	Edges	Edges in each relation				
CKM	246	3	1,551	turn to (480)	discuss with (565)	see most often (506)		
SACCHPOMB	4,092	5	63,033	Direct (1,686)	Physical (7,502)	Suppressive (16,989)	Synthetic (2,664)	Additive (34,192)
DROSOPHILA	8,215	4	43,210	Direct (24,094)	Suppressive (3,461)	Additive (2,669)	Physical 12,986	
HOMO	18,222	3	166,904	Direct (54,678)	Physical (93,662)	Colocalization (18,564)		

simultaneously learned the embedding vector of each node and layer embedding of each layer.

4) *mGCN [12]:* A multiplex network embedding method learning general representation for nodes in all layers. mGCN proposed a multidimensional GCN, which captured the interactions within and across multiple dimensions.
5) *DMGI [20]:* An attributed multiplex network embedding method learning general representation for nodes in all layers. It captures the local patches of a graph and the global properties of the entire graph.
6) *CrossMNA [19]:* A multiplex network embedding model not only learns the specific representation for each relation but also learns the general representation for the whole multiplex network. The representation in each relation was generated by combining the layer vector for each layer and the general embedding for each node.
7) *R-GCN [47]:* An embedding model for heterogeneous networks. R-GCN develops neural networks into heterogeneous networks, which specializes in dealing with the highly multirelational data characteristic of realistic knowledge bases.
8) *GATNE [49]:* An embedding model for heterogeneous networks. GATNE proposes a unified framework to address the problem of embedding learning for the attributed multiplex heterogeneous network, which supports both transductive and inductive learning. It is noted that the multiplex network is seen as a special heterogeneous network with only one node type.

For single-layer network embedding methods, node2vec and vgae, training is first performed on all network layers to learn the node representation of each layer of network, and then, the node representations of all layers are summed up as the final node representation for link prediction in the target network. For multiplex network embedding methods that learn specific embedding for each layer, MELL, the intravector of CrossMNA and GATNE, we also sum up the embedding of all layers as the final representation to predict in the target network. For multiplex network embedding methods that learn general representation for all layers, CrossMNA, mGCN, and DMGI, we take the general representation for prediction. For the convenience of display, we use CMNA_s and CMNA_g to indicate the learned specific representation and general representation, respectively. For the R-GCN, we follow the reference using DistMult factorization [59] as the scoring function to infer links in each layer and average the scores in all layers as the link possibility in the target network.

C. Experimental Settings

For each dataset, we select one layer at a time to be the target network for prediction and the other layers as inputting multiplex network. It is reasonable to use the information-rich networks to predict the other layers, while using a sparse and information-less network to predict a dense network has a relatively high error rate and is meaningless. For the CKM dataset, we choose "turn to" and "discuss with" as target networks to predict in turn and others as known multiplex network from which complementary structural information is learned. For the SACCHPOM dataset, we choose the layer of "direct interaction" and "synthetic genetic interaction" as target networks in turn. For the DROSOPHILA dataset, we choose "suppressive genetic interaction" and "additive genetic interaction" as target networks. For the HOMO dataset, we set "direct interaction" as the target network for prediction.

For the selected target network, all nodes and links present in the network are used as the test set positive samples, and the same number of disconnected edges as the negative samples of the test set is randomly selected. The remaining networks in a multiplex network dataset are taken as the training set. Each method is trained and tested five times, and the average result of the tests is taken as the final result. All models set 128 as the dimension of the final embedding. For CrossMNA, both the dimensions of layer vector d_1 and intervector d_2 are set as 128. For mGCN and DMGI, we use the representations learned by node2vec on each dataset as attributes to input. In addition, for node2vec, the optimal hyperparameters are empirically set as $p = 2$ and $q = 1$. For all models taking negative samples, we set the sample number to 5.

D. Inductive Link Prediction

In practice, link prediction algorithms can be applied to predict unseen links in the network and recommend the top-k most potential links for nodes. Therefore, we evaluate the ability of our model in the inductive link prediction problem from two aspects. As mentioned in the experimental setting, we select one or two networks as the target network for testing on each dataset. We conduct five times experiments for each network and take the average as the final result. For each pair of nodes, we follow the formula defined in (12) to measure the linking possibility.

1) *Predict Unseen Links:* In this case, we randomly sample an equal number of nonedges with positive edges as negative samples in each target network. The evaluation metrics used to evaluate the performance are area under the curve (AUC) and average precision (AP), where AP reflects the accuracy of predicting positive samples. Higher values of AUC and AP indicate better link prediction performance. The AUC score is defined as

$$\mathrm{AUC} = \frac{\sum_{i=1}^{N_p}\sum_{j=1}^{N_n} I\left(\mathrm{pred}(x_i), \mathrm{pred}(y_j)\right)}{N_p * N_n} \tag{14}$$

$$I(a,b) = \begin{cases} 1, & a > b \\ 0.5, & a = b \\ 0, & a < b \end{cases} \tag{15}$$

where pred$(\cdot)$ is the classification result of the model for positive sample x_i and negative sample y_j and N_p and N_n are the number of positive and negative samples, respectively. AP scores are defined as follows:

$$AP = \frac{\sum_k \mathrm{Precision}(k)}{N} \tag{16}$$

where Precision = TPTP+FP, TP denotes true positive, and FP denotes false positive.

The results for CKM, SACCHPOMB, and DROSOPHILA in this task are shown in Fig. 3 and the results of HOMO are shown in Table II.

2) *Top-k:* In this case, we recommend $top@k$ nodes for each node that are most likely to existing links with it in the target network. We compare the results by the score of the recommended node pairs hitting the real existing links and the evaluation metric of $top@k$ can be defined

$$top@k = \frac{|\xi(top@k) \cap \xi^t|}{|\xi^t|} \tag{17}$$

where ξ^t is the positive edge set of the target network and $|\xi^t|$ is the number of positive links. $\xi(top@k)$ is the most likely k edges set recommended for each node. Also, $|\xi(top@k) \cap \xi^t|$ indicates the number of correct predictions among the k edges recommended for the node. Here, we set different k as 5%, 10%, and 15% of nodes to evaluate the effectiveness of recommending potential edges for nodes. Also, the results of CKM, SACCHPOMB, and DROSOPHILA are presented in Fig. 4 and the results of HOMO are shown in Table II.

From the results of the two tasks, we can observe that the following conditions hold.

1) Our proposed model ILAR almost outperforms the other compared methods on all datasets in both predicting unseen links and recommending top k potential links, which validates the superiority of our model in predicting links in new relations by fully utilizing the complementary information and correlated relationship in the multiplex network. In particular, the AUC score of predicting unseen links in the SACCHPOM dataset obtain remarkable results and boost the performance by around 25% when compared to the best comparable method DMGI.
2) In the CKM dataset, the structure is relatively similar between layers, MELL and DMGI enforcing that the same node cross relation close also gets good results comparable to our model in predicting unseen links, while in the recommending top-k case, there is still a gap between their results and our model. In other datasets, MELL gets poor results, and this is because even for the same node, their structures vary greatly between layers, and it is unreasonable to force the embeddings to be closer.
3) DMGI has relatively good results in all datasets, which may be due to the additional information of inputting the

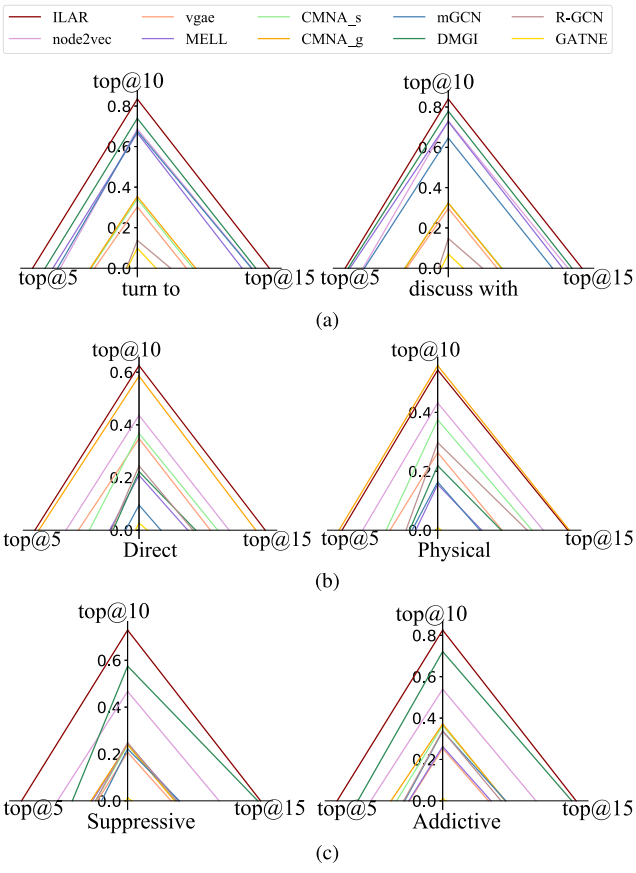

Fig. 3. Performance comparison of top-k in terms of top@5, top@10, and top@15 on (a) CKM, (b) SACCHPOM, and (c) DROSOPHILA, respectively. Note that MGCN and DMGI are attributed network embedding methods.

TABLE II

LINK PREDICTION RESULTS OF PREDICTING UNSEEN LINKS AND TOP-k ON THE HOMO DATASET. (BOLD IS THE BEST RESULT)

Methods	AUC	AP	top@5	top@10	top@15
node2vec	0.584	0.614	**0.248**	0.304	0.351
vgae	0.511	0.520	0.161	0.200	0.240
MELL	0.507	0.567	0.190	0.235	0.275
CMNA_s	0.628	0.561	0.123	0.146	0.161
CMNA_g	0.613	0.544	0.149	0.167	0.179
mGCN	0.587	0.539	0.127	0.205	0.279
DMGI	0.537	0.531	0.116	0.173	0.225
R-GCN	0.500	0.500	0.033	0.060	0.080
GATNE	0.582	0.553	0.009	0.014	0.020
ILAR	**0.761**	**0.767**	0.192	**0.341**	**0.451**

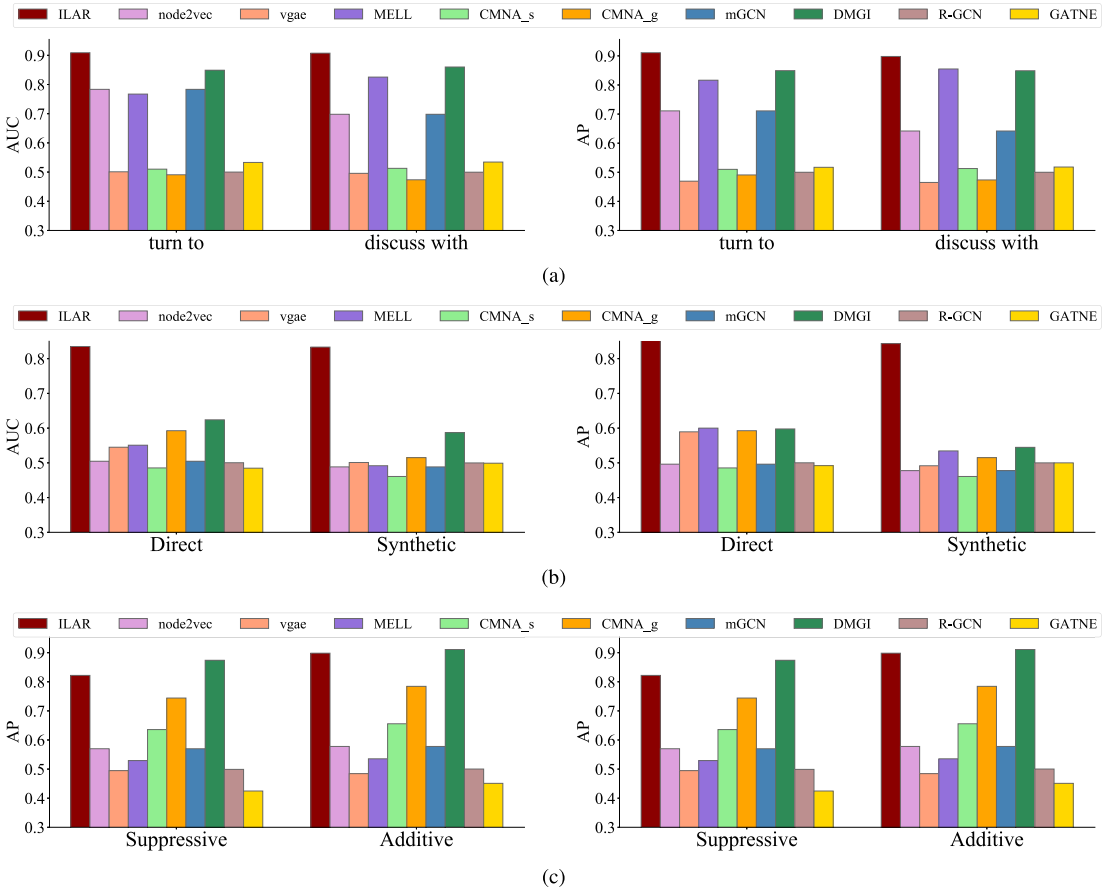

Fig. 4. Performance comparison of predicting unseen links in terms of AUC and AP on (a) CKM, (b) SACCHPOM, and (c) DROSOPHILA. Note that MGCN and DMGI are attributed network embedding methods.

node2vec embedding results as the attributes. We take the test using the identity matrix instead of the node2vec embedding results as attributes, the AUC value on suppressive network only reaches 0.652. Also, DMGI has a strong advantage in distinguishing between positive and negative edges and even outperforms ILAR on the DROSOPHILA dataset, but it does not perform well in recommending links.

4) The general representation of CrossMNA (shown in CMNA_g) works generally more effectively than the methods of learning specific representation in each layer (shown in CMNA_s). Learning specific representation emphasizes the structure-specific information for an individual layer, which may be noise for the target network.
5) The heterogeneous embedding methods have poor performance on almost all datasets. Maybe, it is because heterogeneity is designed for networks that contain multiple types of nodes and edges. When they are applied to multiplex networks with only one type of nodes, they are fundamentally single-view models and do not work well empirically on multiview networks.

E. Ablation Study

We conduct ablation studies on the CKM and Drosophila datasets to demonstrate the boost in predictive performance when considering the cross relationship of interactive learning. The methods for comparison are listed as follows.

1) *ILAR_1:* The proposed model without disparity constraints enforces the two designed convolutional modules to encode different features.
2) *ILAR_2:* The proposed model without the complementary convolutional module learns the complementary information from other networks except for the anchor network. This method is also commonly known as GAE [58].
3) *ILAR_3:* The proposed model defines a specific vector for each layer and a common vector for all layers. The common vector is shared by all layers and trained by a two-layer GCN following our model.

Table III summarizes the results of ablation studies. From the results, we can observe that our model effectively learns sufficient and useful information in multiplex networks. Removing the disparity constraint (ILAR_1) degrades the performance considerably, which shows that through the disparity constraint, the two convolutional modules learn the unique structure that exists in the anchor network and the complementary structure that does not exist in the anchor network. Without the disparity constraint (ILAR_1), the results are even worse than without the complementary convolutional module (ILAR_2), which may be because the redundant information is learned by two convolutional modules. Compared with method 3 and our model ILAR, the results of method 2

TABLE III
ABLATION EXPERIMENT RESULTS

Methods	CKM				DROSOPHILA			
	turn to		discuss with		Suppressive		Additive	
	AUC	top@15	AUC	top@15	AUC	top@15	AUC	top@15
ILAR_1	0.770(0.013)	0.720(0.008)	0.789(0.013)	0.737(0.016)	0.767(0.002)	0.666(0.012)	0.815(0.006)	0.739(0.001)
ILAR_2	0.848(0.003)	0.809(0.006)	0.878(0.006)	0.850(0.001)	0.802(0.006)	0.706(0.001)	0.864(0.001)	0.776(0.002)
ILAR_3	0.873(0.006)	0.852(0.009)	0.897(0.001)	0.882(0.001)	0.801(0.001)	0.705(0.002)	0.856(0.004)	0.792(0.007)
ILAR	**0.909**(0.010)	**0.888**(0.006)	**0.907**(0.005)	**0.887**(0.010)	**0.822**(0.012)	**0.751**(0.009)	**0.899**(0.005)	**0.850**(0.006)

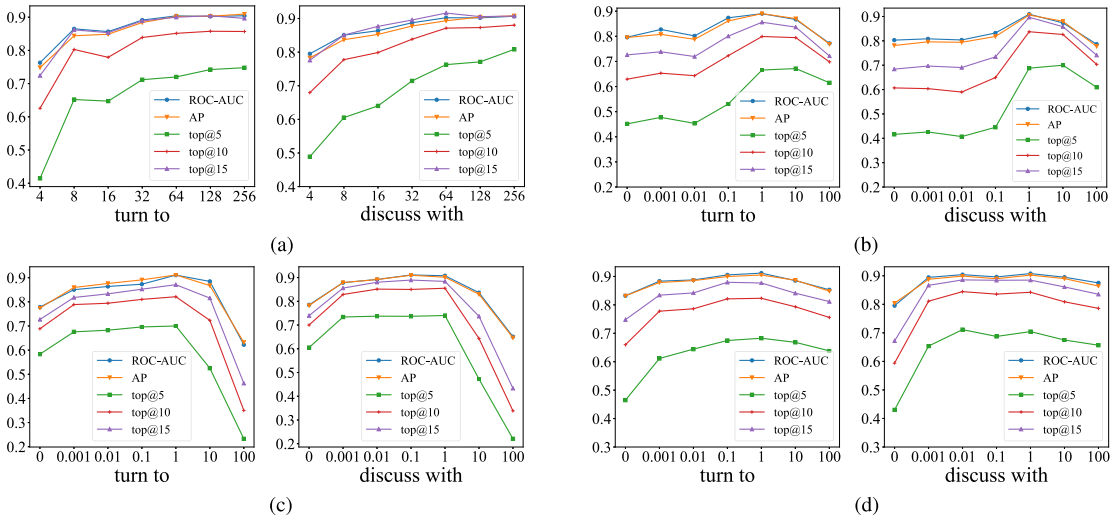

Fig. 5. Parameter sensitivity analysis on the CKM dataset. (a) Analysis of parameter d. (b) Analysis of parameter α. (c) Analysis of parameter β. (d) Analysis of parameter γ.

are weaker, indicating that integrating the correlations in multiplex networks plays a crucial role. Moreover, learning a common vector for all layers (ILAR_3) rather than learning complementary information for each anchor network may only capture the information shared by all layers and lose some essential information for some layers.

F. Parameter Sensitivity

In this section, we study the sensitivity of parameters on the CKM dataset. Specifically, we evaluate how different numbers of the embedding dimensions and different values of hyperparameter α, β, and γ can affect the model. Also, the results are shown in Fig. 5.

1) Parameter of Embedding Dimension d: First, we show how the dimension of the embedding vectors affects the performance in Fig. 5(a). The dimension values are set with various numbers of d ranging from 4 to 256. Also, we can see that the values of all metrics in both target networks of CKM dataset improve with the increase of the dimension until they tend to be smooth. This is intuitive as larger dimension can encode more useful information, while a too large number of dimensions may introduce noises. The ascending trend of our model slows down once the embedding dimension reaches around 32 and the best performance is around 128.

2) Parameter of α: Then, we show how the value of balance coefficient between specific features and complementary features affects the performance in Fig. 5(b). When $\alpha = 0$, the performance is totally determined by the specific features modeled from the anchor network and we can see that it gets the worst value. It demonstrated that both specific features and complementary features are essential for the model. With the increase of α, the performances raise first, but the performance will drop if α is larger than 1.0.

3) Parameter of β: We will check the impact of parameter β. β controls the weight of the complementary convolutional module in the training process. We vary it from 0 to 100 and the results are shown in Fig. 5(c). Similarly, with the increase of β, the performances also raise first, but the performance will drop quickly if β is larger than 1.0.

4) Parameter of γ: Finally, we show how the disparity constraint coefficient γ affects the performance. γ controls the disparity degree of specific features and complementary features from two different modules. We vary it from 0 to 100 and the results are shown in Fig. 5(d). When $\gamma = 0$ and no constraints force them to disparate, the specific and complementary features will model the common and redundant information and then get the worst results. However, the performance also degrades when γ is too large. The reason is that, in this case, the specific features and complementary features are too far apart, and the complementary features cannot apply useful information to the anchor network reconstruction.

VI. CONCLUSION

This article investigates existing network embedding methods of predicting links in multiplex networks and finds that previous studies dealing with the link prediction problem in

multiplex networks mainly focus on inferring intralinks in each relation or anchor links between layers based on the observed multiplex structure. In real life, however, it is often the situation predicting potential links for networks with no topology at all, which has not yet been discussed. This is a novel link prediction issue and we named it inductive link prediction in multiplex networks, which exploits the existing multiple types of relational data to predict completely unknown networks.

When applying to the inductive link prediction, the current multiplex network embedding methods of learning specific node representations for each layer may learn noise information for the target network by emphasizing the specific structure information, and the methods of learning general node representation for all layers may be insufficient to capture the common information among the layers. Therefore, to cope with the novel inductive link prediction problem in multiplex networks, we propose an ILAR network embedding method, which learns the sufficient complementary features through iteratively interacting learning features extracted by two designated convolutional modules. We test our method for the novel link prediction tasks using four datasets compared with several state-of-the-art baseline models. The experimental results demonstrate the superiority of our model in solving this problem.

In this article, we aim at addressing the challenge of link prediction in one unobserved network, while the link prediction of multiple relations will be extended in future work by introducing some external knowledge. We expect that the novel inductive link prediction problem in multiplex networks proposed in this work will be further investigated.

REFERENCES

[1] M. Poongodi, T. N. Nguyen, M. Hamdi, and K. Cengiz, "Global cryptocurrency trend prediction using social media," *Inf. Process. Manage.*, vol. 58, no. 6, Nov. 2021, Art. no. 102708.

[2] G. Chandrasekaran, T. N. Nguyen, and D. J. Hemanth, "Multimodal sentimental analysis for social media applications: A comprehensive review," *WIREs Data Mining Knowl. Discovery*, vol. 11, no. 5, p. e1415, May 2021.

[3] X. Wang, D. Jin, K. Musial, and J. Dang, "Topic enhanced sentiment spreading model in social networks considering user interest," in *Proc. AAAI Conf. Artif. Intell.*, Apr. 2020, vol. 34, no. 1, pp. 989–996.

[4] X. Chen, H.-J. Zhao, S. Zhao, J. Chen, and Y.-P. Zhang, "Citation recommendation based on citation tendency," *Scientometrics*, vol. 121, no. 2, pp. 937–956, Nov. 2019.

[5] D. A. Abduljabbar, S. Hashim, and R. Sallehuddin, "An enhanced evolutionary algorithm for detecting complexes in protein interaction networks with heuristic biological operator," in *Proc. Int. Conf. Soft Comput. Data Mining*, 2020, pp. 334–345.

[6] J. Gao, G. Manogaran, T. N. Nguyen, S. Kadry, C.-H. Hsu, and P. M. Kumar, "A vehicle-consensus information exchange scheme for traffic management in vehicular ad-hoc networks," *IEEE Trans. Intell. Transp. Syst.*, early access, Feb. 8, 2022, doi: 10.1109/TITS.2021.3130087.

[7] D. Liben-Nowell and J. Kleinberg, "The link-prediction problem for social networks," *J. Amer. Soc. Inf. Sci. Technol.*, vol. 58, no. 7, pp. 1019–1031, 2007.

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

[9] M. De Domenico *et al.*, "Mathematical formulation of multilayer networks," *Phys. Rev. X*, vol. 3, no. 4, 2013, Art. no. 041022.

[10] K.-M. Lee, B. Min, and K.-I. Goh, "Towards real-world complexity: An introduction to multiplex networks," *Eur. Phys. J. B*, vol. 88, no. 2, pp. 1–20, Feb. 2015.

[11] H. Zhang, L. Qiu, L. Yi, and Y. Song, "Scalable multiplex network embedding," in *Proc. 27th Int. Joint Conf. Artif. Intell. (IJCAI)*, vol. 18, 2018, pp. 3082–3088.

[12] Y. Ma, S. Wang, C. C. Aggarwal, D. Yin, and J. Tang, "Multi-dimensional graph convolutional networks," in *Proc. Int. Conf. Data Mining (SDM)*, Calgary, AB, Canada, May 2019, pp. 657–665.

[13] A. P. Singh and G. J. Gordon, "Relational learning via collective matrix factorization," in *Proc. 14th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2008, pp. 650–658.

[14] L. Liu, W. K. Cheung, X. Li, and L. Liao, "Aligning users across social networks using network embedding," in *Proc. IJCAI*, 2016, pp. 1774–1780.

[15] S. Zhang and H. Tong, "FINAL: Fast attributed network alignment," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2016, pp. 1345–1354.

[16] X. Kong, J. Zhang, and P. S. Yu, "Inferring anchor links across multiple heterogeneous social networks," in *Proc. 22nd ACM Int. Conf. Inf. Knowl. Manage.*, 2013, pp. 179–188.

[17] T. Man, H. Shen, S. Liu, X. Jin, and X. Cheng, "Predict anchor links across social networks via an embedding approach," in *Proc. IJCAI*, vol. 16, 2016, pp. 1823–1829.

[18] M. Heimann, H. Shen, T. Safavi, and D. Koutra, "REGAL: Representation learning-based graph alignment," in *Proc. 27th ACM Int. Conf. Inf. Knowl. Manage.*, Oct. 2018, pp. 117–126.

[19] X. Chu, X. Fan, D. Yao, Z. Zhu, J. Huang, and J. Bi, "Cross-network embedding for multi-network alignment," in *Proc. World Wide Web Conf., (WWW)*, San Francisco, CA, USA, May 2019, pp. 273–284.

[20] C. Park, D. Kim, J. Han, and H. Yu, "Unsupervised attributed multiplex network embedding," in *Proc. AAAI Conf. Artif. Intell.*, vol. 34, no. 4, pp. 5371–5378, Apr. 2020.

[21] J. B. Tenenbaum, V. de Silva, and J. C. Langford, "A global geometric framework for nonlinear dimensionality reduction," *Science*, vol. 290, no. 5500, pp. 2319–2323, Dec. 2000.

[22] S. T. Roweis and L. K. Saul, "Nonlinear dimensionality reduction by locally linear embedding," *Science*, vol. 290, no. 5500, pp. 2323–2326, Dec. 2000.

[23] M. Belkin and P. Niyogi, "Laplacian eigenmaps for dimensionality reduction and data representation," *Neural Comput.*, vol. 15, no. 6, pp. 1373–1396, 2003.

[24] J. Bhatta, D. Shrestha, S. Nepal, S. Pandey, and S. Koirala, "Efficient estimation of nepali word representations in vector space," *J. Innov. Eng. Educ.*, vol. 3, no. 1, pp. 71–77, Mar. 2013.

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

[26] A. Grover and J. Leskovec, "Node2vec: Scalable feature learning for networks," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2016, pp. 855–864.

[27] J. Tang, M. Qu, M. Wang, M. Zhang, J. Yan, and Q. Mei, "Line: Large-scale information network embedding," in *Proc. 24th Int. Conf. World Wide Web*, 2015, pp. 1067–1077.

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

[29] S. Cao, W. Lu, and Q. Xu, "Deep neural networks for learning graph representations," in *Proc. AAAI Conf. Artif. Intell.*, vol. 30, no. 1, 2016, pp. 1145–1152.

[30] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *Proc. Int. Conf. Learn. Represent.*, 2017, pp. 1–14.

[31] M. Defferrard, X. Bresson, and P. Vandergheynst, "Convolutional neural networks on graphs with fast localized spectral filtering," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 29, 2016, pp. 3844–3852.

[32] J. Bruna, W. Zaremba, A. Szlam, and Y. LeCun, "Spectral networks and locally connected networks on graphs," in *Proc. Int. Conf. Learn. Represent.*, 2014, pp. 1–14.

[33] W. L. Hamilton, R. Ying, and J. Leskovec, "Inductive representation learning on large graphs," in *Proc. 31st Adv. Neural Inf. Process. Syst.*, vol. 30, 2017, pp. 1025–1035.

[34] A. Vaswani *et al.*, "Attention is all you need," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 30, 2017, pp. 6000–6010.

[35] P. Velickovic, W. Fedus, W. L. Hamilton, P. Lio, Y. Bengio, and R. D. Hjelm, "Deep graph infomax," in *Proc. Int. Conf. Learn. Represent.*, 2019, pp. 1–46.

[36] R. Matsuno and T. Murata, "MELL: Effective embedding method for multiplex networks," in *Proc. Companion Web Conf.*, 2018, pp. 1261–1268.

[37] C. Park, C. Yang, Q. Zhu, D. Kim, H. Yu, and J. Han, "Unsupervised differentiable multi-aspect network embedding," in *Proc. 26th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2020, pp. 1435–1445.

[38] R. Lu, P. Jiao, Y. Wang, H. Wu, and X. Chen, "Layer information similarity concerned network embedding," *Complexity*, vol. 2021, pp. 1–10, Aug. 2021.

[39] Y. Ouyang, B. Guo, X. Tang, X. He, J. Xiong, and Z. Yu, "Learning cross-domain representation with multi-graph neural network," 2019, *arXiv:1905.10095*.

[40] S. K. Ata, Y. Fang, M. Wu, J. Shi, C. K. Kwoh, and X. Li, "Multi-view collaborative network embedding," *ACM Trans. Knowl. Discovery Data*, vol. 15, no. 3, pp. 1–18, Apr. 2021.

[41] Q. Meng, T. Jian, J. Shang, R. Xiang, and J. Han, "An attention-based collaboration framework for multi-view network representation learning," in *Proc. ACM Conf. Inf. Knowl. Manage.*, 2017, pp. 1767–1776.

[42] M. Gong, W. Liu, Y. Xie, Z. Tang, and M. Xu, "Heuristic 3D interactive walk for multilayer network embedding," *IEEE Trans. Knowl. Data Eng.*, early access, Sep. 3, 2020, doi: 10.1109/TKDE.2020.3021393.

[43] A. Mitra, P. Vijayan, R. Sanasam, D. Goswami, S. Parthasarathy, and B. Ravindran, "Semi-supervised deep learning for multiplex networks," in *Proc. 27th ACM SIGKDD Conf. Knowl. Discovery Data Mining*, Aug. 2021, pp. 1234–1244.

[44] B. Jing, C. Park, and H. Tong, "HDMI: High-order deep multiplex infomax," in *Proc. Web Conf.*, Apr. 2021, pp. 2414–2424.

[45] Y. Dong, N. V. Chawla, and A. Swami, "Metapath2vec: Scalable representation learning for heterogeneous networks," in *Proc. 23rd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2017, pp. 135–144.

[46] C. Shi, Y. Lu, L. Hu, Z. Liu, and H. Ma, "RHINE: Relation structure-aware heterogeneous information network embedding," *IEEE Trans. Knowl. Data Eng.*, vol. 34, no. 1, pp. 433–447, Jan. 2022.

[47] M. Schlichtkrull, T. N. Kipf, P. Bloem, R. V. Berg, I. Titov, and M. Welling, "Modeling Relational Data With Graph Convolutional Networks," in *Proc. ESWC*. Springer, 2018, pp. 593–607.

[48] B. Hu, Y. Fang, and C. Shi, "Adversarial learning on heterogeneous information networks," in *Proc. 25th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Jul. 2019, pp. 120–129.

[49] Y. Cen, X. Zou, J. Zhang, H. Yang, J. Zhou, and J. Tang, "Representation learning for attributed multiplex heterogeneous network," in *Proc. 25th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Jul. 2019, pp. 1358–1368.

[50] X. Wang *et al.*, "Heterogeneous graph attention network," in *Proc. World Wide Web Conf.*, May 2019, pp. 2022–2032.

[51] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *Proc. 5th Int. Conf. Learn. Represent.*, 2017, pp. 1–14.

[52] L. Song, A. Smola, A. Gretton, K. M. Borgwardt, and J. Bedo, "Supervised feature selection via dependence estimation," in *Proc. 24th Int. Conf. Mach. Learn. (ICML)*, 2007, pp. 823–830.

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

[54] T. N. Nguyen and S. Zeadally, "Mobile crowd-sensing applications: Data redundancies, challenges, and solutions," *ACM Trans. Internet Technol.*, vol. 22, no. 2, pp. 1–15, May 2022.

[55] J. Coleman, E. Katz, and H. Menzel, "The diffusion of an innovation among physicians," *Sociometry*, vol. 20, no. 4, pp. 253–270, 1957.

[56] C. Stark, "BioGRID: A general repository for interaction datasets," *Nucleic Acids Res.*, vol. 34, no. 1, pp. D535–D539, Jan. 2006.

[57] M. D. Domenico, V. Nicosia, A. Arenas, and V. Latora, "Structural reducibility of multilayer networks," *Nature Commun.*, vol. 6, p. 6864, Apr. 2015.

[58] T. N. Kipf and M. Welling, "Variational graph auto-encoders," *Stat*, vol. 1050, p. 21, Nov. 2016.

[59] B. Yang, S. W.-T. Yih, X. He, J. Gao, and L. Deng, "Embedding entities and relations for learning and inference in knowledge bases," in *Proc. Int. Conf. Learn. Represent. (ICLR)*, 2015, pp. 1–12.

Mengzhou Gao received the B.E. degree in mechanical design, manufacturing, and automation from the Harbin Institute of Technology, Harbin, China, in 2012, and the Ph.D. degree in control science and engineering from Zhejiang University, Hangzhou, China, in 2017.

Since 2017, she has been an Assistant Professor with the School of Cyberspace, Hangzhou Dianzi University, Hangzhou. Her current research interests include cyber-physical systems security, secure control, and complex networks.

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

From 2018 to 2021, he was a Lecturer with the Center of Biosafety Research and Strategy, Tianjin University. He is currently a Professor with the School of Cyberspace, Hangzhou Dianzi University, Hangzhou, China. His current research interests include complex network analysis and its applications.

Ruili Lu received the bachelor's degree from Shenzhen University, Shenzhen, China, in 2016. She is currently pursuing the master's degree with the Tianjin International Engineering Institute, Tianjin University, Tianjin, China.

Her current research interests mainly focus on complex networks and multiplex network embedding.

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

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

Yinghui Wang received the master's degree from Tianjin University, Tianjin, China, in 2018, where she is currently pursuing the D.Eng. degree with the School of Computer Science and Technology.

Her current research interests include complex network analysis and computational social science.

Zhidong Zhao (Member, IEEE) received the B.S. and M.S. degrees in mechanical engineering from the Nanjing University of Science and Technology, Nanjing, China, in 1998 and 2001, respectively, and the Ph.D. degree in biomedical engineering from Zhejiang University, Hangzhou, China, in 2004.

He is currently a Full Professor with Hangzhou Dianzi University, Hangzhou. His research interests include biomedical signal processing, wireless sensor networks, biometrics, and machine learning.