# Variational Autoencoder based Bipartite Network Embedding by Integrating Local and Global Structure

Pengfei Jiao[a], Minghu Tang[b], Hongtao Liu[b], Yaping Wang[b], Chunyu Lu[b], Huaming Wu[c,*]

[a] *Center of Biosafety Research and Strategy, Tianjin University, Tianjin, China*
[b] *College of Intelligence and Computing, Tianjin University, Tianjin, China*
[c] *Center of Applied Mathematics, Tianjin University, Tianjin, China*

**Abstract**

As a powerful tool for machine learning on the graph, network embedding, which projects nodes into low-dimensional spaces, has a variety of applications on complex networks. Most current methods and models are not suitable for bipartite networks, which have two different types of nodes and there are no links between nodes of the same type. Furthermore, the only existing methods for bipartite network embedding ignore the internal mechanism and highly nonlinear structures of links. Therefore, in this paper, we propose a new deep learning method to learn the node embedding for bipartite networks based on the widely used autoencoder framework. Moreover, we carefully devise a node-level triplet including two types of nodes to assign the embedding by integrating the local and global structures. Meanwhile, we apply the variational autoencoder (VAE), a deep generation model with natural advantages in data generation and reconstruction, to enhance the node embedding for the highly nonlinear relationships between nodes and complex features. Experiments on some widely used datasets show the effectiveness of the proposed model and corresponding algorithm compared with some baseline network (and bipartite) embedding techniques.

*Keywords:* Bipartite network embedding, Local and global structure, Variational autoencoder, Nonlinear structure

*Corresponding author
*Email address:* whming@tju.edu.cn (Huaming Wu )

## 1. Introduction

Network embedding (NE), which aims to learn the latent representation of nodes that can be used for a variety of tasks such as link prediction, community detection and node classification in complex networks [1], has recently become a popular research problem based on the neural network and deep learning [2, 3]. In particular, NE is typically denoted as a pairwise proximity function that represents each node as a feature vector based on the proximity, the node content or the label of the network. Based on the diversity of notations, motivations, and conceptual models, many different types of methods have been proposed for homogeneous NE [4], such as matrix factorization-based approaches [5, 6], random walk-based approaches [7, 8] and graph convolutional approaches [9, 10].

A large number of methods and models for homogeneous NE have been developed and have shown good performance and effectiveness in certain tasks on many networks. However, these methods cannot be well applied to the bipartite network, which usually consists of two types of nodes and the links exist only between different types, such as the user-item networks in recommender systems [11, 12] and author-venues networks in DBLP [13]. There are two primary reasons why these methods fail. Firstly, in the homogeneous NE, the node embedding is learned all based on a hypothesis that the representations of two nodes should be similar or closer if there is a direct link between them and vice versa, which is not valid in the bipartite network. Existing links in the bipartite network are usually called distinct relations, which depict the similarity between two types of nodes, and the proximity among nodes of the same type is called implicit relation, though there are no direct links in the network we have observed. Secondly, the internal characteristics and mechanisms of the bipartite network have not been preserved, such as the power-law degree distribution of nodes based on random walk-based methods, which has been elaborated in [14].

As far as we know, a class of NE methods can be used for bipartite networks, i.e., methods of embedding heterogeneous information networks (HINs), which

usually have more than one type of nodes and links, so the bipartite network can be as a special case. A typical type of method of NE for HINs is based on meta-path, such as Metapath2Vec++ [15], which formalizes meta-path-based random walks to construct the heterogeneous neighborhood of a node and then leverages a heterogeneous skip-gram model to learn the node embeddings of HIN. However, these methods have two limitations: (1) the difference between the distinct and implicit relationships with different link semantics in bipartite networks is not considered; (2) the imbalance problem, i.e., the numbers of two types of nodes in the bipartite network are usually unequal, or even quite different when applying the methods of NE for HINs, and the similarity between different types of nodes has a larger proportion, resulting in worse embeddings for applications.

In addition to traditional methods like singular value decomposition (SVD) and matrix decomposition, which usually have high computational complexity, BiNE [14] has been recently proposed for NE of the bipartite network. In BiNE, an optimization framework is proposed that learns the embedding of nodes based on a well-designed random walk and accounts for both the explicit and implicit relations. However, this method still faces two shortcomings: (1) the highly nonlinear relationships from the embeddings to links that are ubiquitous in complex networks are neglected; (2) there are many parameters that need to be adjusted, which weakens the representation result and limits its application.

Fortunately, a widely used framework, autoencoder (AE), designed to model the nonlinear relationships of data [16, 17], has been developed to the NE, such as SDNE [18], which is a semi-supervised deep learning model capable of capturing the highly nonlinear structure with multiple layers of nonlinear functions. Furthermore, the SDNE preserves the local network structure by exploiting the first-order and second-order proximity. However, compared with the AE, the variational autoencoder (VAE) [19] has been successfully applied to image modeling and recommender systems. It is also extended to NE in homogeneous networks, such as DVNE [20] and the deep generative model by exploiting the VAE [21], analogously, these methods have the same shortcomings as the meth-

ods previously used for the NE in homogeneous networks. Considering these NE methods based on the AE and VAE, it is necessary to design a new representation learning method for bipartite network which can model its intrinsic statistical properties.

To address these challenges, in this paper, we propose a new deep learning method called BiVAE, to learn the embedding for bipartite networks based on the widely used autoencoder framework. In particular, we develop VAE to enhance the node embedding for the highly nonlinear relationships between nodes and complex features. To fill in the gap of VAE and NE for bipartite networks, we carefully devise the triplet of node-level including the two types of nodes to assign the embeddings by integrating the local and global structure, for one node with the only type of itself in the triple. We use a generic VAE framework to represent and reconstruct the global structure of the node. For the other two nodes with the same type, we design a unified VAE framework for parameter sharing. We also propose a mechanism to preserve the local structure via the distinct relations of the bipartite networks. Compared with the previous NE methods, our proposed method is well designed based on the inherent properties of bipartite networks. It can effectively deal the links between different types of nodes, and model the intrinsic similarity between nodes of the same type although no links are observed. In summary, the contributions of this paper can be summarized as follows:

- We propose BiVAE, a deep learning framework based on the variational autoencoder for network embedding of bipartite networks, which could model the highly nonlinear relations effectively and integrate the global and local structural features of the network solidly.

- A triplet consisting of two nodes of the same type and one node of another type, is proposed to enhance the node embedding of bipartite networks based on inherent characteristics.

- Experiments on some widely used datasets show that the effectiveness of

the proposed model and corresponding algorithm compared to some state-of-the-art homogeneous and bipartite network embedding techniques.

The rest of the paper is organized as follows. In section 2, we discuss the existing works related to this paper. In section 3, we briefly introduce the bipartite network and its embedding. The proposed model BiVAE is presented in section 4. In section 5, we implement our proposed method in experiments and show the comparisons with the baseline approaches. Section 6 concludes the paper and highlights future work.

## 2. Related Work

Complex network analysis has a variety of applications in different fields [22]. Community detection, link prediction and identifying influential nodes in complex networks are important and difficult tasks [23][24]. Network embedding, projecting the topology structure into a lower dimensional continuous space, is an effective dimension reduction [25] framework on complex networks and has devoted to different network tasks. In this paper, we focus on the representation learning of bipartite networks based on deep learning. So here we introduce the popular network embedding methods as follows.

**Homogeneous network.** Most of the previous embedding methods and models were designed only for homogeneous networks with only one type of nodes and edges [26]. These methods usually have two goals, i.e., reconstruction and inference, so the network structure and its properties should be preserved [1]. As mentioned above, the commonly used models can be divided into matrix factorization-based, random walk-based, and graph neural network (GNN)-based methods [27]. Laplacian eigenmaps and non-negative matrix factorization are commonly used in network embedding based on the low-rank approximation [5], however, these methods usually have high computational complexity. The core idea of the random walk-based methods is similar to the classic Word2Vector, in which a word (one node in the network) embedding vector should be able to reconstruct the vectors of its neighborhoods defined

by a co-occurrence rate in the documents [28, 29]. DeepWalk [7] and Node2Vec [8] are representative methods that use different random walk strategies. As discussed in the introduction, the intrinsic challenge of network embedding is to find a mapping function from the original network to the embedding space, which is usually high nonlinear, so the methods based on GNNs show competitive and preferable performance. Some representative methods, such as SDNE [18], SDAE [30], and SiNE [31], have been proposed and applied to various tasks.

**Heterogeneous information network (HIN).** Due to the heterogeneity of nodes and edges in the HIN, the most important step is to get the proximity of the same and different types of nodes in HINs. Currently, heterogeneous network representation learning can be divided into three types: random walk-based methods, network decomposition methods and deep learning methods. Random walk-based methods usually use meta-paths [32], a composite relation connecting two objects, to capture the structure and rich semantic information. Metapath2vec [15] first uses meta-paths to guide random walks over a HIN, and then feeds the walk sequence to a SkipGram model [28] to obtain node embeddings. HIN2Vec [33] cannot only learn the representation of nodes but also the vector representation of meta-paths. In addition, JUST [34] uses random walks with JUmp and STay strategies that do not involve capturing node embeddings of any meta-path. The network decomposition methods usually decompose heterogeneous networks into several simple networks, perform representation learning on these networks, respectively, and finally integrate these representations. PTE [35] decomposes the heterogeneous network into three bipartite networks, and perform representation learning on the decomposed subnetworks based on the skip-gram model.

In recent years, some work has begun to use deep learning to model heterogeneous network data. SHINE [36] utilizes multiple deep autoencoders to extract highly nonlinear user representations from the sentiment network, social network and profile network, respectively. The three user representations are then fused together through a specific aggregation function to obtain the final node representation. As a novel and powerful graph representation learn-

ing method, GNN has shown excellent performance in network analysis, which has aroused wide research interest. HAN [37] leverages node-level attention and semantic-level attention to learn the importance of nodes and meta-paths in a hierarchical manner, respectively, which enables the learned node embeddings to better capture the complex structure and rich semantic information.

**Bipartite network.** As a typical type of heterogeneous information networks, the bipartite networks also have a variety of applications, such as for recommendation systems. To the best of our knowledge, BiNE [14] and BiNE-IEI [38] are the only embedding methods for the bipartite network at present. The former learns the node representations for bipartite networks by purposefully performing biased random walks, which can capture the implicit relations of the networks, and the later models the nonlinear generation mechanism of links via the autoencoder and decoder models.

## 3. Problem and Definitions

In this section, we introduce the bipartite network and its embedding, as a preliminary introduction to the proposed model later.

We reiterate that there are two types of nodes in the bipartite network, and the links only exist between different types of nodes. Therefore, we define a bipartite network as $G = (U, V, E)$, where $U = \{u_1, u_2, \cdots, u_n\}$ and $V = \{v_1, v_2, \cdots, v_m\}$ represent node sets of both two types, respectively, $E = \{e_1, e_2, \cdots, e_l\}$ denotes the edge set. Here $n = |U|$, $m = |V|$ and $l = |E|$ denote the number of nodes of different types and the links of $G$, respectively, $\mathbf{A} = \{\mathbf{a}_i\}_{i=1}^{n} \in R^{n \times m}$ is the adjacency matrix of $G$ and $\mathbf{a}_i = \{a_{ij}\}_{j=1}^{m}$, where $a_{ij} > 0$ indicates that there exists a link between nodes $u_i$ and $v_j$, and otherwise $a_{ij} = 0$. Bipartite network embedding aims to find a function that converts each node of both two types of the network into a low-dimensional representation as a vector.

Before introducing the framework of our model, we first describe some notations used in this paper as Table 1, and furthermore, we give formal definitions

Table 1: Terms and notations.

| Symbol | Definition |
|---|---|
| $G = (U, V, E)$ | A bipartite network |
| $U$, $V$ | The set of nodes of two types in $G$ |
| $E \subseteq U \times V$ | The set of links in $G$ |
| $n$, $m$, $l$ | The number of two type nodes and links |
| $\mathbf{x} = \{\mathbf{x}_i\}$ | The $i$-th column of $\mathbf{A}$ |
| $\mathbf{y} = \{\mathbf{y}_j\}$ | The $j$-th row of $\mathbf{A}$ |
| $K$ | The number of layers in BiVAE model |
| $\mathbf{W}_k$, $\hat{\mathbf{W}}_k$ | The $k$-th layer weight matrices |
| $\mathbf{b}_k$, $\hat{\mathbf{b}}_k$ | The $k$-th layer biases |
| $\lambda_1, \lambda_2$ | The balanced parameters |
| $\mathbf{z}_i$, $\mathbf{z}_j$ | The embedding vectors of nodes $u_i$ and $v_j$ |

related to our model as follows.

**Definition 1 (The Set of Valid Triplets).** *Let $M = D \cup S$ denote the set of all valid triplets, where $D = \{(i,j,k)|\forall v_i \in V, \forall u_j, u_k \in U, a_{ij} > 0, a_{ik} = 0\}$ and $S = \{(i,j,k)|\forall u_i \in U, \forall v_j, v_k \in V, a_{ij} > 0, a_{ik} = 0\}$. Each triplet of $M$ is composed of two types of nodes, in which node $i$ is the neighbor of node $j$ but not the neighbor of node $k$ .*

**Definition 2 (Local Network Structure).** *For any pair of nodes $(u_i, v_j)$, if they are linked by an observed edge, there exists local structure between them and can be defined as: $\{a_{ij}|a_{ij} > 0, \forall u_i \in U, \forall v_j \in V\}$, which are usually described by the similarity between directly connected nodes.*

**Definition 3 (Global Network Structure).** *The global network structure between nodes can be described by the similarity of their neighborhood structures. For any node $u_i \in U$, the set of its neighbors is formulated as $Nb_{u_i} = \{v_j|\{a_{ij} > 0\}_{j=1}^{m}\}$. Similarly, the neighbors of any node $v_j \in V$ is defined as: $Nb_{v_j} = \{u_i|\{a_{ij} > 0\}_{i=1}^{n}\}$. The global structure between any pair of nodes $(u_i, v_j)$ is determined by the similarity of $Nb_{u_i}$ and $Nb_{v_j}$. If none of the nodes linked with both $u_i$ and $v_j$, there is no global structure between them.*

**Definition 4 (First Order Structure).** *For each node $u_i \in U$ or $v_j \in V$, we denote its neighbors as the First Order Structure of the node.*

**Definition 5 (Second Order Structure).** *For the node $u_i$, its neighbors is denoted as the set $Nb_{u_i}$, for each node $v_j \in Nb_{u_i}$, we set a new set $U_i' = \cup Nb_{v_j}$ including the node $u_i$, then, for each $u_k \in U_i'$, we set $V_l' = \cup Nb_{u_k}$, so we could denote the Second Order Structure of node $u_i$ as $V_l'/Nb_{u_i}$. For each node in $V$, it has a similar definition.*

## 4. Proposed Model

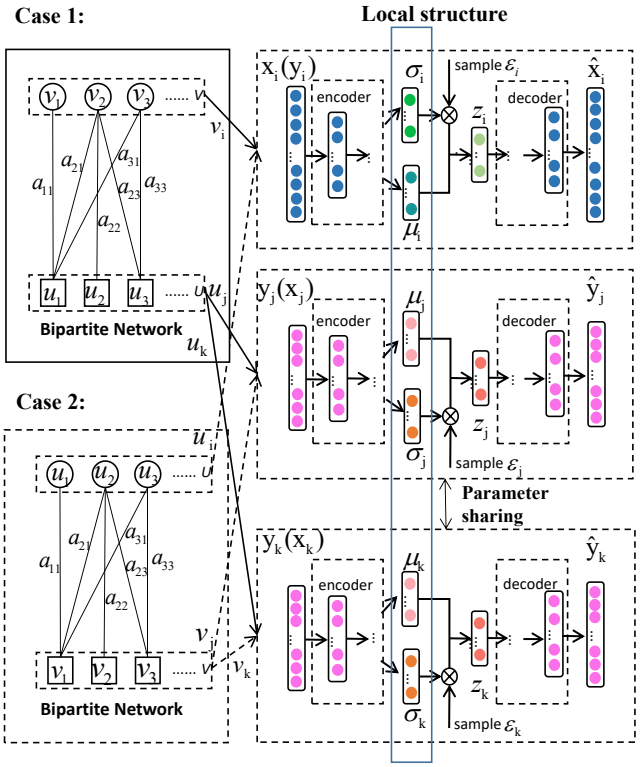



Figure 1: The framework of the proposed BiVAE. For a given bipartite network, $V$ and $U$ represent sets of the two types of nodes, case 1 and case 2 are the set of triplets as we have denoted. $\mathbf{x}$ and $\mathbf{y}$ represent vector representations of the two nodes of type $V$ and $U$, respectively. For each triplet $i, j, k$, we take the VAE to encode the corresponding nodes and design the parameter sharing for the decoding process on nodes with the same types.

In this section, we present our proposed model named BiVAE, as shown in Fig. 1. In detail, we propose a deep architecture consisting of three parallel deep VAEs that capture the highly nonlinear structure of the bipartite network and

embed nodes as Gaussian distributions to capture the uncertainty [39]. In order to preserve the global structure of the network, we reconstruct the neighborhood structure of each node by VAEs. Furthermore, we notice the similarity between neighbor nodes is larger than that between non-neighbor nodes. Therefore, we define a set of all valid triplets to preserve the local structure of the network while enhancing the effect of network embedding.

In the following subsections, we will describe the details of our model implementation and the loss functions. Because three parallel deep VAEs only differ in their inputs, we will focus on illustrating the detailed process when taking $\mathbf{x}$ as the input data. The same process is applied for the other two deep VAEs when taking $\mathbf{y}$ as the input data.

### *4.1. Encoder*

Given the adjacency matrix $\mathbf{A}$, the high-dimensional vector representations of any node $v_i$ of type $V$ and $u_j$ of type $U$ are formalized as $\mathbf{x}_i$ and $\mathbf{y}_j$ (the input data) separately, in which $\mathbf{x}_i$ represents the $i$-th column of $\mathbf{A}$ and $\mathbf{y}_j$ represents the $j$-th row of $\mathbf{A}$. For a deep VAE, given the input data $\mathbf{x}_i$, the output $\mathbf{h}_k^i$ of node $i$ for the $k$-th layer is shown as follows:

$$\mathbf{h}_1^i = f_1(\mathbf{W}_1\mathbf{x}_i + \mathbf{b}_1) \tag{1}$$

$$\mathbf{h}_k^i = f_1(\mathbf{W}_k\mathbf{h}_{k-1}^i + \mathbf{b}_k), k = 2, \cdots, K \tag{2}$$

where $f_1$ is the tanh function of each layer. In the last layer of the encoder, we obtain the mean vector $\boldsymbol{\mu}_i$ and the standard deviation vector $\boldsymbol{\sigma}_i$ of the input data distribution, which can be formulated as follows:

$$\boldsymbol{\mu}_i = \mathbf{W}_\mu\mathbf{h}_k^i + \mathbf{b}_\mu \tag{3}$$

$$\boldsymbol{\sigma}_i = f_2((\mathbf{W}_\sigma\mathbf{h}_k^i + \mathbf{b}_\sigma)/2) + 1 \tag{4}$$

where $\mathbf{W}_\mu$ and $\mathbf{W}_\sigma$ are the weight matrices, $\mathbf{b}_\mu$ and $\mathbf{b}_\sigma$ are the bias vectors, and $f_2$ is the exponential linear unit function [40]. We use $f_2((\mathbf{W}_\sigma\mathbf{h}_k^i + \mathbf{b}_\sigma)/2) + 1$

to ensure that $\boldsymbol{\sigma}_i$ is positive definite. Furthermore, we apply the reparameterization trick to reparameterize variable $\mathbf{z}_i$. The computing process can be presented as follows:

$$\mathbf{z}_i = \boldsymbol{\mu}_i + \boldsymbol{\sigma}_i * \epsilon_i, \;\; \epsilon_i \sim \mathcal{N}(0, \mathbf{I}) \tag{5}$$

where the value $\epsilon_i$ is sampled form $\mathcal{N}(0, \mathbf{I})$ and $\mathbf{z}_i \in R^d$.

### *4.2. Decoder*

Reversely, the calculation processes in each decoder layer are as follows:

$$\hat{\mathbf{h}}_K^i = f_1(\hat{\mathbf{W}}_{K+1}\mathbf{z}_i + \hat{\mathbf{b}}_{K+1}) \tag{6}$$

$$\hat{\mathbf{h}}_k^i = f_1(\hat{\mathbf{W}_{k+1}}\hat{\mathbf{h}}_{k+1}^i + \hat{\mathbf{b}}_{k+1}), k = K-1, \cdots, 2 \tag{7}$$

$$\hat{\mathbf{x}}_i = f_1(\hat{\mathbf{W}}_1\hat{\mathbf{h}}_1 + \hat{\mathbf{b}}_1) \tag{8}$$

where $\hat{\mathbf{x}_i}$ is the reconstructed data of $\mathbf{x}_i$.

During the encoding and decoding process, we reconstruct the neighborhood structure of node $i$ and preserve the global structure. In order to enhance the embedding effect, we introduce the constraint of triplets.

We need to emphasize that, although it seems that the technique used in our paper seems similar to that of DVNE [20], it is fundamentally different from DVNE. In summary, the differences lie in three aspects. First, we are dealing with a very different issue. DVNE mainly considers the classic complex network, and makes full use of the statistical characteristics of that. However, our work mainly considers the bipartite network. According to the BiNE [14], the bipartite network has some very different statistical characteristics, such as it does not obey the power-law of degree distribution. Second, although we all use the VAE framework, we have fully considered the characteristics (the bipartite network, which usually consists of two types of nodes and the links exist only between different types) of the binary network. Third, although we all use Local Structure and Global Structure in our technique, but they mean

different meanings, and we fully consider the characteristics of the bipartite network.

### *4.3. Loss Functions*

We try to develop effective loss functions to learn deeply learned feature and improve the embedding effect.

#### *4.3.1. Triplet Constraint*

Directly connected nodes are closer than that whose links do not exist. We define a set of all triplets $M$ to restrain the relations between nodes. Meanwhile, we use the $2^{th}$ Wasserstein distance to measure the similarity between the distribution of two nodes, which can speed up the calculation process and preserve the transitivity of similarity [20], and the formula is shown as:

$$\begin{aligned} W_2^{'}(\mathbf{p}_i, \mathbf{p}_j)^2 =& W_2^{'}(\mathcal{N}(\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i), \mathcal{N}(\boldsymbol{\mu}_j, \boldsymbol{\Sigma}_j))^2 \\ =& \|(\boldsymbol{\mu}_i - \boldsymbol{\mu}_j)\|_2^2 + \|(\boldsymbol{\Sigma}_i^{1/2} - \boldsymbol{\Sigma}_j^{1/2})\|_F^2 \end{aligned} \tag{9}$$

where $W_2^{'}$ denotes the $2^{th}$ Wasserstein distance, $\mathbf{p}_i = \mathcal{N}(\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)$ is the low dimensional Gaussian distribution of node $i$. We focus on the diagonal covariance matrices, and thus $\boldsymbol{\Sigma}_i\boldsymbol{\Sigma}_j = \boldsymbol{\Sigma}_j\boldsymbol{\Sigma}_i$ and $\boldsymbol{\Sigma}_i = \boldsymbol{\sigma}_i^2\boldsymbol{I}$.

The relationships between two types of nodes under the constraint of triplets should satisfy the following inequality:

$$W_2^{'}(\mathbf{p}_i, \mathbf{p}_j) < W_2^{'}(\mathbf{p}_i, \mathbf{p}_k), \quad \forall (i, j, k) \in M \tag{10}$$

The effect of network embedding can be enhanced under the constraint of triplets.

*4.3.2. Loss Function of the Local Structure*

Given a triplet $(i, j, k) \in M$, the loss function of the local structure can be formulated as:

$$\mathcal{L}_{local} = \sum_{(i,j,k)\in M} \left[ {E_{ij}^{'}}^{2} + \exp(-E_{ik}^{'}) \right] \tag{11}$$

The method is based on the energy between two nodes [39], and we denote the energy between node $i$ and node $j$ as $E_{ij}^{'}$ and $E_{ij}^{'} = W_2^{'}(\mathbf{p}_i, \mathbf{p}_j)$.

*4.3.3. Loss Function of the Global Structure*

Given a triplet $(i, j, k) \in M$, the loss function of the global structure is represented as follows:

$$\begin{aligned} \mathcal{L}_{global} = \sum_{(i,j,k)\in D} & (\|(\mathbf{x}_i - \hat{\mathbf{x}}_i) \odot \mathbf{g}_i\|_2^2 + \|(\mathbf{y}_j - \hat{\mathbf{y}}_j) \odot \mathbf{z}_j\|_2^2 \\ & + \|(\mathbf{y}_k - \hat{\mathbf{y}}_k) \odot \mathbf{z}_k\|_2^2) + \sum_{(i,j,k)\in S} (\|(\mathbf{y}_i - \hat{\mathbf{y}}_i) \odot \mathbf{z}_i\|_2^2 \\ & + \|(\mathbf{x}_j - \hat{\mathbf{x}}_j) \odot \mathbf{g}_j\|_2^2 + \|(\mathbf{x}_k - \hat{\mathbf{x}}_k) \odot \mathbf{g}_k\|_2^2) \end{aligned} \tag{12}$$

where $\odot$ denotes the Hadamard product. Given a triplet $(i, j, k)$ of $D$, $\mathbf{g}_i = \{c_{ij}\}_{j=1}^{n}$, $\mathbf{z}_j = \{c_{ji}\}_{i=1}^{m}$ and $\mathbf{z}_k = \{c_{ki}\}_{i=1}^{m}$.

In order to alleviate the sparsity problem of networks, we introduce the hyper-parameter $\beta$ to impose more penalties on the reconstruction error of non-zero elements than that of zero elements. For any node $v_i$ of type $V$ and $u_j$ of type $U$, if $v_i$ and $u_j$ are not connected $c_{ij} = 1$, otherwise $c_{ij} = \beta > 1$.

*4.3.4. Joint Loss Function*

We incorporate the loss functions of the global and local structures of our proposed model into a joint objective function, which is defined as follows:

$$\mathcal{L} = \mathcal{L}_{local} + \lambda_1 \mathcal{L}_{global} + \lambda_2 \mathcal{L}_{reg} \tag{13}$$

where $\mathcal{L}_{reg}$ is the regularization term to prevent overfitting, which is shown as follows:

$$\begin{aligned}\mathcal{L}_{reg} = &\sum_{k=1}^{K}(\|\mathbf{W}_k\|_F^2 + \|\hat{\mathbf{W}}_k\|_F^2 + \|\mathbf{b}_k\|_2^2 + \|\hat{\mathbf{b}}_k\|_2^2) \\ &+\|\mathbf{W}_\mu\|_F^2 + \|\mathbf{W}_\sigma\|_F^2 + \|\hat{\mathbf{W}}_{K+1}\|_F^2 + \|\mathbf{b}_\mu\|_2^2 \\ &+\|\mathbf{b}_\sigma\|_2^2 + \|\hat{\mathbf{b}}_{K+1}\|_2^2 \end{aligned} \tag{14}$$

### *4.4. Training Algorithm of BiVAE*

The entire learning procedure of BiVAE is presented in Algorithm 1. To be specific, Line 1 initializes all weight matrices and bias vectors; Line 2 uses negative sampling to construct the set of triplets $M = D \cup S$; Line $3-9$ learn the embeddings and the gradient is calculated using back-propagation and Adadelta algorithm.

Here we give the procedure of constructing the set of triplets in detail. For a node $u_i$ of type $U$, we randomly sample two nodes $v_j$ and $v_k$ of type $V$, where $v_j \in Nb_{u_i}$ and $v_k \notin Nb_{u_i}$. Thus we can obtain the set of triplets $D$. Reversely, given a node $v_i$ of type $V$, we can construct the corresponding triplet using the same negative sampling method and obtain the set of triplets $S$.

It is worth noting that, compared with the VAE framework, our algorithm has the same computational complexity as that for optimizing the VAE on networks. Although we need to consider all the nodes with different types, the bipartite network is usually much more sparse for that there are no links between nodes with the same type. Considering the optimization mechanism of our model, it is based on the negative sampling and stochastic gradient descent, our algorithm does not increase the computational complexity. Furthermore, as we know, the BiNE [14] takes the random walk mechanism and usually has higher computational complexity than the AE and VAE framework.

**Algorithm 1** Training algorithm of BiVAE

**Input**: Adjacency matrix of the bipartite network $\mathbf{A}$, the set of all valid triplets $M$, the parameters $\beta$, $\lambda_1$ and $\lambda_2$

**Output**: Network embeddings $\mathbf{Z} = \{\mathbf{z}_i\}_{i=1}^{n+m}$ and parameters $\mathbf{W} = \{\mathbf{W}_1, \cdots, \mathbf{W}_K, \mathbf{W}_\mu, \mathbf{W}_\sigma\}$, $\mathbf{b} = \{\mathbf{b}_1, \cdots, \mathbf{b}_K, \mathbf{b}_\mu, \mathbf{b}_\sigma\}$, $\hat{\mathbf{W}} = \{\hat{\mathbf{W}}_1, \cdots, \hat{\mathbf{W}}_{K+1}\}$, and $\hat{\mathbf{b}} = \{\hat{\mathbf{b}}_1, \cdots, \hat{\mathbf{b}}_{K+1}\}$

Initialize the parameters: $\theta = \{\mathbf{W}, \hat{\mathbf{W}}, \mathbf{b}, \hat{\mathbf{b}}\}$
Construct the set of triplets $M$ via negative sampling
**for** each triplet $(i, j, k) \in M$ **do**
    Feed the vector representations of each node in the triplet into encoders, obtain $\boldsymbol{\sigma} = \{\boldsymbol{\sigma}_i, \boldsymbol{\sigma}_j, \boldsymbol{\sigma}_k\}$ and $\boldsymbol{\mu} = \{\boldsymbol{\mu}_i, \boldsymbol{\mu}_j, \boldsymbol{\mu}_k\}$
    Sample $\epsilon_i$, $\epsilon_j$, and $\epsilon_k$ from three Gaussian distributions, respectively
    Calculate $\mathbf{z}_i$, $\mathbf{z}_j$ and $\mathbf{z}_k$ w.r.t. Eq. 5
    Decode the latent representations $\mathbf{z}_i$, $\mathbf{z}_j$ and $\mathbf{z}_k$ to obtain the reconstruction data
    Based on Eq. 13 update $\theta$
    Until convergence
**end for**
Obtain the network embeddings $\mathbf{Z}$

## 5. Experiments and Analysis

In this section, we introduce some datasets and baselines that are used in our paper, and then we give experimental results on different network tasks.

### *5.1. Datasets*

In order to evaluate the effectiveness of our proposed framework, we use three benchmark datasets: DBLP[1], VisualizeUs[2], Wikipedia[3], Movielens[4] and Wikibooks[5]. The bipartite network constructed from DBLP dataset is the publish bipartite network, where the links mean the publishing relations between the authors and the venues. The VisualizeUs contains the bipartite picture tagging network, where the links denote the tagging relations between pictures and tags. The network of Wikipedia contains the edit relationships between authors and

[1] http://dblp.uni-trier.de/xml
[2] http://konect.uni-koblenz.de/networks/pics_ti
[3] http://konect.uni-koblenz.de/networks/wikipedia_link_en
[4] http://konect.uni-koblenz.de/networks/movielens-100k_rating
[5] http://konect.uni-koblenz.de/networks/edit-frwikibooks

pages. This Movielens network consists of 100,000 user–movie ratings, an edge between a user and a movie represents a rating of the movie by the user. The Wikibooks is a bipartite edit network of the French Wikipedia, it contains users and pages from the French Wikipedia, connected by edit events and each edge represents an edit. We summarize the statistics of these datasets in Table 2.

Table 2: Statistics of the datasets. The $| U |$, $| V |$, $| E |$ and Density are denoted as the number of left nodes, right nodes, edges, and its density of each bipartite network

| **Metric** | **DBLP** | **VisualizeUs** | **Wikipedia** | **Movielens** | **Wikibooks** |
|---|---|---|---|---|---|
| $\| U \|$ | 6001 | 6009 | 15000 | 943 | 2884 |
| $\| V \|$ | 1308 | 3355 | 3214 | 1682 | 27732 |
| $\| E \|$ | 29256 | 38780 | 172426 | 100000 | 67613 |
| Density | 0.4% | 0.2% | 0.4% | 6.30% | 0.08% |

### *5.2. Baselines*

In terms of link prediction tasks and recommendation tasks, we compare BiVAE with the following three categories of methods for evaluations:

(1) For the evaluations of link prediction tasks, we use our previous proposed methods [41], including Common Neighbors (CN), Jaccard's Index (JC), Adamic Adar (AA), and Preferential Attachment (PA), which are based on the topological structure in bipartite networks.

(2) We compare our algorithm for the Top-$N$ item recommendation tasks with the following methods:

- **BPR** [42]: Bayesian Personalized Ranking (BPR) is a pairwise ranking approach which is widely used for item recommendation tasks.
- **RankALS** [43]: It is a method for personalized ranking by minimizing a ranking objective function rather than the conventional prediction mean squared error.
- **FISM** [44]: It is an item-based method for top-$N$ recommendation tasks and learns the item-item similarity matrices, which can alleviate the sparsity of the datasets.

(3) Compared to BiVAE, the following methods are based on network embedding and can be applied to the link prediction and recommendation tasks.

- **DeepWalk** [7]: It can learn the representations of nodes by performing the truncated random walks.
- **LINE** [45]: This method is designed to preserve the first-order and second-order similarities when the nodes of networks are embedded into the low-dimensional space.
- **Node2vec** [8]: This method designs a biased random walk strategy to generate the corpus of node sequences.
- **Metapath2vec++** [15]: This is a heterogeneous network embedding method by performing the meta-path-guided random walk.
- **BiNE** [14]: This is a bipartite network embedding method that can model the explicit and implicit relations.

### *5.3. Parameter Settings*

We use an encoder and a decoder with a single hidden layer. The dimension of each layer is $512 - 128 - 512$. Besides, the parameter $\beta$ of reconstruction error for non-zero elements is set to 5. The balanced hyper-parameters of the overall loss function is set to $\lambda_1 = 20$ and $\lambda_2 = 0.02$, respectively. The gradient is calculated using back-propagation and the parameters are optimized using Adadelta. In addition, the parameters for all the baselines are tuned to be optimal for the model mentioned in their own papers.

Furthermore, to analyze the influence of sampling mechanism in the algorithm of our model on different tasks[46, 47], we design a new sample method instead of the random sampling on the non-links of the network when training the model. In detail, it samples the non-links from the Second Order structure of each node based on our Definitions 4 and 5. We set this as another version of our model **BiVAE**, called **BiVAE+local_ns**. We also give its results on different tasks.

Table 3: Link prediction on DBLP and Wikipedia.

| Performance(%) | DBLP | | Wiki-pedia | |
|---|---|---|---|---|
| | AUC-ROC | AUC-PR | AUC-ROC | AUC-PR |
| **CN** | 82.85 | N/A | 86.85 | 90.68 |
| **JC** | 81.05 | N/A | 63.90 | 73.04 |
| **AA** | 82.70 | N/A | 87.37 | 91.12 |
| **PA** | 81.05 | N/A | 90.71 | 93.37 |
| **DeepWalk** | 66.94 | 71.51 | 89.71 | 91.20 |
| **LINE** | 69.36 | 73.64 | 91.62 | 93.28 |
| **Node2vec** | 63.24 | 67.69 | 89.93 | 91.23 |
| **Metapath** | 71.61 | 66.78 | 89.56 | 91.72 |
| **BiNE** | 84.48 | 86.21 | 92.91 | 94.45 |
| **BiVAE** | **85.70** | **86.23** | **95.22** | **95.83** |
| **BiVAE+local_ns** | **85.72** | **86.39** | **95.08** | **95.36** |

### *5.4. Link Prediction*

In this task, we take each node pair without edges as a negative instance, while the link is treated as a positive instance. For DBLP, Wikipedia, Movielens and Wikibooks datasets, we randomly sample 60% of the instances as the training samples and use the left instances as the test samples. After the training, we can obtain the vector representations of the nodes, and concatenate the representations of two nodes in networks to obtain the vector representation of an edge. Then, the representation of edges are treated as features and we take whether a node pair has an edge as the ground truth. We use the area under the ROC curve (AUC-ROC) and the Precision-Recall curve (AUC-PR) as the evaluation metrics [14]. The results of the link prediction tasks are shown in Table 3. Note that the symbol "N/A" in Table 3 indicates that the result of the link prediction task cannot be computed, which is due to some baselines cannot be applied to large-scale networks. From the results, the main observations and analysis are as follows:

- Tables 3 and 4 show that the BiVAE and BiVAE+local_ns outperform all baselines based on the network topological structure in all the datasets. This is because these baselines only consider the local or global network structure.

Table 4: Link prediction on Movielens and Wikibooks.

| Performance(%) | Movielens | | Wikibooks | |
|---|---|---|---|---|
| | **AUC-ROC** | **AUC-PR** | **AUC-ROC** | **AUC-PR** |
| **CN** | 50.00 | 56.00 | 50.30 | 54.80 |
| **JC** | 50.20 | 56.30 | 50.20 | 54.80 |
| **AA** | 50.10 | 56.10 | 50.10 | 54.90 |
| **PA** | 50.30 | 56.30 | 50.10 | 54.90 |
| **DeepWalk** | 76.31 | 77.34 | 58.18 | 61.37 |
| **LINE** | 76.14 | 77.33 | 58.27 | 61.66 |
| **Node2vec** | 76.3 | 77.45 | 57.87 | 61.19 |
| **Metapath** | 81.67 | 81.65 | 68.53 | 71.68 |
| **BiNE** | 83.17 | 84.39 | 61.05 | 63.49 |
| **BiVAE** | **85.47** | **86.01** | **69.56** | **73.13** |
| **BiVAE+local_ns** | **85.34** | **86.03** | **69.66** | **73.85** |

- All the methods based on the Network Embedding usually have better performance based on the AUC-ROC and AUC-PR on all the datasets.

- The performance of DeepWalk, LINE, and Node2vec methods designed for homogeneous networks are generally worse than the BiNe, BiVAE and BiVAE+local_ns. This is because our method and the BiNE can model the special properties of bipartite networks.

- The performance of Metapath2vec++ is worse than BiNE and BiVAE on both datasets in both matrices. This implies that the methods tailored for bipartite networks perform much better. In addition, compared to the BiNE, although the improvement is a little tiny, the BiVAE outperforms better, which may due to BiVAE can model the relations of nodes by the set of all valid triplets and capture the uncertainty of nodes in bipartite networks. Additionally, the computational complexity of BiVAE is lower than that of BiNE.

- Compared with the BiVAE and BiVAE+local_ns, the former has better performance on the Wiki-pedia and Movielens, and the BiVAE+local_ns has achieved excellent results on the DBLP and Wikibooks. So we have reason to believe that the different sampling mechanisms do influence the

Table 5: Top-10 recommendation on VisualizeUs, DBLP, and Wikipedia.

| Performance(%) | Visua-lizeUs | | DBLP | | Wiki-pedia | |
|---|---|---|---|---|---|---|
| | F1@10 | MAP@10 | F1@10 | MAP@10 | F1@10 | MAP@10 |
| **BPR** | 6.22 | 5.51 | 8.95 | 13.55 | 14.12 | 17.20 |
| **RankALS** | 2.72 | 1.50 | 7.62 | 7.52 | 9.70 | 14.05 |
| **FISM** | 10.25 | 8.86 | 9.81 | 7.38 | 16.03 | 16.74 |
| **DeepWalk** | 5.82 | 4.28 | 8.50 | 19.71 | 2.28 | 1.20 |
| **LINE** | 9.62 | 7.81 | 8.99 | 9.62 | 5.52 | 14.93 |
| **Node2vec** | 6.73 | 6.25 | 8.54 | 19.44 | 3.83 | 2.59 |
| **Metapath** | 5.92 | 5.35 | 8.65 | 19.06 | 2.05 | 1.26 |
| **BiNE** | 13.63 | 16.46 | 11.79 | 20.62 | 13.67 | 19.66 |
| **BiVAE** | **21.94** | **43.75** | **25.76** | **42.62** | **25.75** | **52.84** |
| **BiVAE+local_ns** | **17.48** | **31.79** | **20.32** | **32.26** | **25.27** | **48.95** |

Table 6: Top-10 recommendation on Movielens and Wikibooks.

| Performance(%) | Movielens | | Wikibooks | |
|---|---|---|---|---|
| | F1@10 | MAP@10 | F1@10 | MAP@10 |
| **BPR** | 4.26 | 3.90 | 0.95 | 0.64 |
| **RankALS** | 1.10 | 2.40 | 1.22 | 3.36 |
| **FISM** | 2.15 | 3.36 | 1.02 | 0.76 |
| **DeepWalk** | 1.89 | 0.39 | 0.06 | 0.01 |
| **LINE** | 2.41 | 0.54 | 0.10 | 0.01 |
| **Node2vec** | 2.05 | 0.44 | 0.07 | 0.01 |
| **Metapath** | 2.68 | 0.63 | 1.39 | 3.77 |
| **BiNE** | 11.90 | 3.88 | 1.81 | 1.32 |
| **BiVAE** | **12.75** | **4.05** | **5.91** | **9.24** |
| **BiVAE+local_ns** | **12.29** | **3.95** | **4.94** | **9.06** |

results of the model, and how to design more effective and efficient sampling strategies is a more challenging issue.

### *5.5. Recommendation*

The recommendation is an important application of network embedding, which can reflect the ability of node representations. For each dataset, 60% of the links are randomly sampled for training, and the rest links are used as test samples. We use the inner product of two different types of nodes to evaluate the preference between them and select Top-10 items with larger scores for the recommendation. Table 5 shows the results of recommendations on three datasets by using two metrics, i.e., F1@10 and MAP@10 [14], the 6 gives the corresponding results on datasets of the Movielens and Wikibooks. The main observations from the results are summarized as follows.

- As shown in Tables 5 and 6, it is obvious that BiVAE and BiVAE+local_ns outperform all baselines on the five datasets in both matrices. This demonstrates the effectiveness of our proposed method on item recommendation tasks, and the improvement in this result is very significant.

- Among all the comparison methods, the BiNE achieve the best performance on these networks, other methods have yielded better results on only one or more datasets.

- Our method is superior to DeepWalk, LINE and Node2vec, which are the three state-of-the-art homogeneous network embedding methods. This indicates that although these homogeneous network embedding methods can be applied to bipartite networks, they consider neither the importance of the types of nodes nor the relations between the two types of nodes.

- The methods based on heterogeneous networks perform worse than our method significantly. This is because: (1) Metapath2vec++ is not optimal for bipartite networks composed of two types of nodes; (2) Although BiNE is tailored for bipartite networks, it does not take into account using the triplets to constrain the relations between nodes and cannot model the uncertainty of nodes in bipartite networks.

- Compared with the BiVAE, the BiVAE+local_ns has competitive results on all the datasets and metrics, this means that the random sampling mechanism is more conducive to the proposed method.

### *5.6. Parameters Sensitivity*

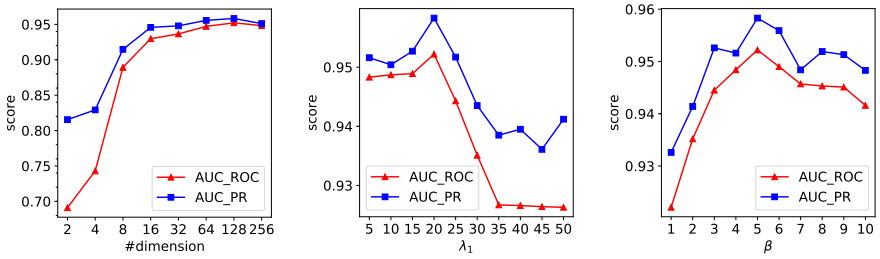
In this subsection, we investigate the influence of the embedding dimension $d$ and two hyper-parameters $\lambda_1$ and $\beta$. We report AUC-ROC and AUC-PR on the Wikipedia dataset to analyze the link prediction tasks as shown in Fig. 2,

Figure 2(a) shows how the embedding dimension affects the performance. The experimental performance first increases and then remains stable when $d$ reaches a certain value. This implies that when the embedding dimension is

too large, our framework cannot embed more useful information due to the introduction of some noises.

The hyper-parameter $\lambda_1$ is a balanced parameter, which denotes the weight of the global structure. From Fig. 2(b), we observe that the performance first increases and then decreases when $\lambda_1 = 20$. This demonstrates that preserving the global structure is essential to our framework. Especially, when $\lambda_1$ approaches 0, our model degenerates to consider only the local structures of the network, when $\lambda_1$ is big enough, the global term plays a larger role. From this figure, it is easy to know that our model performs better when we consider the both terms with a good tradeoff.

Figure 2(c) shows how the hyper-parameter $\beta$ affects the performance, where $\beta$ is the weight that imposes on the non-zero elements. The best performance is obtained when $\beta = 5$, while the performance decreases when $\beta$ is too large.




(a) $d$ (b) $\lambda_1$ (c) $\beta$

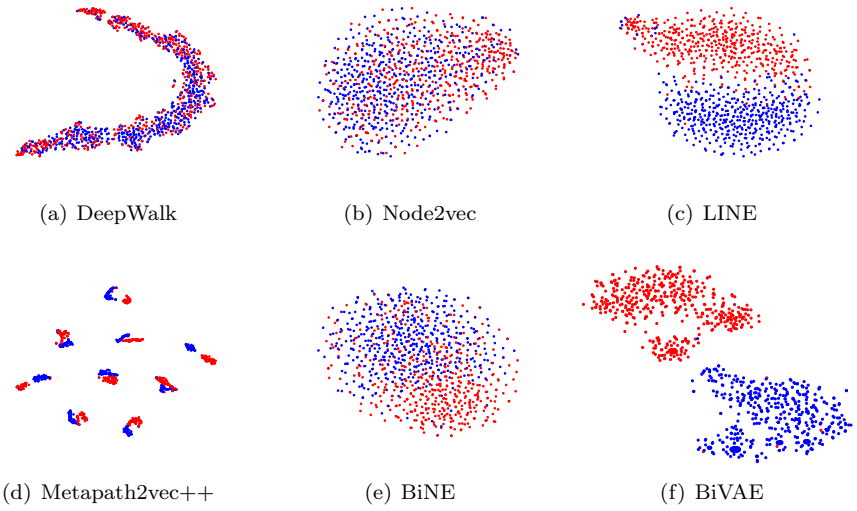
Figure 2: Parameter w.r.t. dimension $d$, hyper-parameters $\lambda_1$ and $\beta$

### *5.7. Visualization*

We show the visualization in this subsection, which is another important application for network embedding. Because of the lack of ground truth in our datasets mentioned earlier, we visualize the node representations of a subset of Aminer dataset[6] by using the visualization tool t-SNE [48]. We construct a heterogeneous collaboration network from a subset of Aminer dataset, which

[6]https://www.aminer.cn/data

consists of 981 authors and 28 venues. What's more, these nodes are from the research fields of theoretical computer science and computer science databases & information systems. A link represents the relationship between an author and a venue. Besides, we take the research field in which the author publishes the most papers as the ground truth of the author.



(a) DeepWalk (b) Node2vec (c) LINE

(d) Metapath2vec++ (e) BiNE (f) BiVAE

Figure 3: Visualization of authors in the subset of Aminer.

The visualization results are shown in Fig. 3. We use different colors to represent the author's research fields, where blue dots represent the research field of theoretical computer science and red dots represent the research field of computer science databases & information systems. From Fig. 3, we observe that our method can clearly separate two types of nodes when compared with DeepWalk, Node2vec, metapath2vec++ and BiNE, whose two types of nodes are mixed together. Although LINE can better separate the two types of nodes, it's obviously not good enough compared with BiVAE. The observations demonstrate the superiority of our method in this task.

## 6. Discussions and Conclusions

In this paper, we propose a deep learning framework BiVAE, which uses the set of triplets to restrain the relations between two different types of nodes to enhance the effect of node embedding in bipartite networks. The variational autoencoder framework can capture the uncertainty of nodes. Besides, BiVAE can preserve the local and global structures while establishing highly nonlinear relations. Further more, we define the Second Order Structure of the bipartite network and propose another new sample mechanism for the training algorithm.

In comparison with several state-of-the-art baselines, experiments on different tasks validate the superiority of our proposed framework. On the link prediction task, the BiVAE and BiVAE+local_ns not only have obvious improvement on the classic link prediction models and homogeneous network embedding methods, but also have some performance advantages with lower computational complexity. On the recommendation task, our method has been a significant performance improvement than all the baselines including the BiNE.

There are still some problems to be studied in the future. How to set the balance parameters automatically in the objective function? How to construct a more effective and efficient negative sampling mechanism to improve the performance of the model? Besides, it is possible to integrate external information for the bipartite network embedding, which will be our next work.


## Acknowledgment

This work was supported by the National Natural Science Foundation of China (61902278, 61801325, 91746205, 91746107) and the Key R&D Program of Tianjin (17ZLZDZF00430, 18YFZCSF01370).


## References


[1] P. Cui, X. Wang, J. Pei, W. Zhu, A Survey on Network Embedding, IEEE Transactions on Knowledge and Data Engineering 31 (5) (2019) 833–852.

[2] Y. LeCun, Y. Bengio, G. Hinton, Deep learning, nature 521 (7553) (2015) 436–444.

[3] R. Zhang, P. Xie, C. Wang, G. Liu, S. Wan, Classifying transportation mode and speed from trajectory data via deep multi-scale learning, Computer Networks 162 (2019) 106861.

[4] S. Pan, R. Hu, G. Long, J. Jiang, L. Yao, C. Zhang, Adversarially Regularized Graph Autoencoder for Graph Embedding., in: IJCAI, International Joint Conferences on Artificial Intelligence Organization, California, 2018, pp. 2609–2615.

[5] M. Ou, P. Cui, J. Pei, Z. Zhang, W. Zhu, Asymmetric Transitivity Preserving Graph Embedding, in: the 22nd ACM SIGKDD International Conference, ACM Press, New York, New York, USA, 2016, pp. 1105–1114.

[6] X. Wang, P. Cui, J. Wang, J. Pei, W. Zhu, S. Yang, Community preserving network embedding, in: Thirty-First AAAI Conference on Artificial Intelligence, 2017.

[7] B. Perozzi, R. Al-Rfou, S. Skiena, Deepwalk: Online learning of social representations, in: Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2014, pp. 701–710.

[8] A. Grover, J. Leskovec, Node2vec: Scalable feature learning for networks, in: Proceedings of the 22Nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2016, pp. 855–864.

[9] H. Wang, D.-Y. Yeung, Towards Bayesian Deep Learning: A Framework and Some Existing Methods, IEEE Transactions on Knowledge and Data Engineering 28 (12) (2016) 3395–3408.

[10] P. Velickovic, G. Cucurull, A. Casanova, A. Romero, P. Lió, Y. Bengio, Graph Attention Networks., in: International Conference on Learning Representations, 2018.

[11] T. Zhou, Z. Kuscsik, J. G. Liu, M. Medo, J. R. Wakeling, Y. C. Zhang, Solving the apparent diversity-accuracy dilemma of recommender systems, Proceedings of the National Academy of Sciences 107 (10) (2010) 4511–4515.

[12] S. Chen, Y. Peng, Matrix factorization for recommendation with explicit and implicit feedback, Knowledge-Based Systems 158 (2018) 109–117.

[13] J. Z. Jie Tang and, L. Yao, L. Z. Li, Juanzi and, Z. Su, ArnetMiner - extraction and mining of academic social networks., in: KDD, ACM Press, New York, New York, USA, 2008, p. 990.

[14] M. Gao, L. Chen, X. He, A. Zhou, Bine: Bipartite network embedding, in: The 41st International ACM SIGIR Conference on Research; Development in Information Retrieval, 2018, pp. 715–724.

[15] Y. Dong, N. V. Chawla, A. Swami, Metapath2vec: Scalable representation learning for heterogeneous networks, in: Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2017, pp. 135–144.

[16] Z. Gao, H.-Z. Xuan, H. Zhang, S. Wan, K.-K. R. Choo, Adaptive fusion and category-level dictionary learning model for multi-view human action recognition, IEEE Internet of Things Journal (2019).

[17] S. Ding, S. Qu, Y. Xi, S. Wan, Stimulus-driven and concept-driven analysis for image caption generation, Neurocomputing (2019).

[18] D. Wang, P. Cui, W. Zhu, Structural Deep Network Embedding, in: the 22nd ACM SIGKDD International Conference, ACM Press, New York, New York, USA, 2016, pp. 1225–1234.

[19] T. N. Kipf, M. Welling, Variational Graph Auto-Encoders., CoRR (2016).

[20] D. Zhu, P. Cui, D. Wang, W. Zhu, Deep variational network embedding in wasserstein space, in: Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery, ACM, 2018, pp. 2827–2836.

[21] H. Li, H. Wang, Z. Yang, M. Odagaki, Variation Autoencoder Based Network Representation Learning for Classification, in: Proceedings of ACL 2017, Student Research Workshop, Association for Computational Linguistics, 2017, pp. 56–61.

[22] S. Mao, F. Xiao, Time series forecasting based on complex network analysis, IEEE Access 7 (2019) 40220–40229.

[23] T. Bian, Y. Deng, Identifying influential nodes in complex networks: A node information dimension approach, Chaos: An Interdisciplinary Journal of Nonlinear Science 28 (4) (2018) 043109.

[24] B. Wei, Y. Deng, A cluster-growing dimension of complex networks: From the view of node closeness centrality, Physica A: Statistical Mechanics and its Applications 522 (2019) 80 – 87.

[25] G. Chao, Y. Luo, W. Ding, Recent advances in supervised dimension reduction: A survey, Machine Learning and Knowledge Extraction 1 (1) (2019) 341–358.

[26] M. Khosla, A. Anand, V. Setty, A comprehensive comparison of unsupervised network representation learning methods, arXiv:1903.07902, 2019 (2019).

[27] H. Chen, B. Perozzi, R. Al-Rfou, S. Skiena, A tutorial on network embeddings, arXiv:1808.02590, 2018 (2018).

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

[29] Q. Le, T. Mikolov, Distributed representations of sentences and documents, in: International conference on machine learning, 2014, pp. 1188–1196.

[30] S. Cao, W. Lu, Q. Xu, Deep neural networks for learning graph representations, in: Thirtieth AAAI Conference on Artificial Intelligence, 2016.

[31] D. Zhang, J. Yin, X. Zhu, C. Zhang, SINE: Scalable Incomplete Network Embedding, 2018 IEEE International Conference on Data Mining (ICDM) (2018) 737–746.

[32] Y. Sun, J. Han, X. Yan, P. S. Yu, T. Wu, Pathsim: Meta path-based top-k similarity search in heterogeneous information networks, Proceedings of the VLDB Endowment 4 (11) (2011) 992–1003.

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

[34] R. Hussein, D. Yang, P. Cudré-Mauroux, Are meta-paths necessary?: Revisiting heterogeneous graph embeddings, in: Proceedings of the 27th ACM International Conference on Information and Knowledge Management, ACM, 2018, pp. 437–446.

[35] J. Tang, M. Qu, Q. Mei, Pte: Predictive text embedding through large-scale heterogeneous text networks, in: Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, ACM, 2015, pp. 1165–1174.

[36] H. Wang, F. Zhang, M. Hou, X. Xie, M. Guo, Q. Liu, Shine: Signed heterogeneous information network embedding for sentiment link prediction, in: Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining, ACM, 2018, pp. 592–600.

[37] X. Wang, H. Ji, C. Shi, B. Wang, Y. Ye, P. Cui, P. S. Yu, Heterogeneous graph attention network, in: The World Wide Web Conference, ACM, 2019, pp. 2022–2032.

[38] Y. Wang, P. Jiao, W. Wang, C. Lu, H. Liu, B. Wang, Bipartite network embedding via effective integration of explicit and implicit relations, in: Database Systems for Advanced Applications, 2019, pp. 435–451.

[39] A. Bojchevski, S. Günnemann, Deep gaussian embedding of graphs: Unsupervised inductive learning via ranking, in: International Conference on Learning Representations, 2018.

[40] D.-A. Clevert, T. Unterthiner, Hochreiter, Fast and accurate deep network learning by exponential linear units (elus), in: International Conference on Learning Representations, 2016.

[41] W. Wang, X. Chen, P. Jiao, D. Jin, Similarity-based regularized latent feature model for link prediction in bipartite networks, Scientific reports 7 (1) (2017) 16996.

[42] S. Rendle, C. Freudenthaler, Z. Gantner, L. Schmidt-Thieme, Bpr: Bayesian personalized ranking from implicit feedback, in: Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence, 2009, pp. 452–461.

[43] G. Takács, D. Tikk, Alternating least squares for personalized ranking, in: Proceedings of the Sixth ACM Conference on Recommender Systems, 2012, pp. 83–90.

[44] S. Kabbur, X. Ning, G. Karypis, Fism: Factored item similarity models for top-n recommender systems, in: Proceedings of the 19th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2013, pp. 659–667.

[45] J. Tang, M. Qu, M. Wang, M. Zhang, J. Yan, Q. Mei, Line: Large-scale information network embedding, in: Proceedings of the 24th International Conference on World Wide Web, 2015, pp. 1067–1077.

[46] M. Armandpour, P. Ding, J. Huang, X. Hu, Robust negative sampling for network embedding, in: Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 33, 2019, pp. 3191–3198.

[47] H. Gao, H. Huang, Self-paced network embedding, in: Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, ACM, 2018, pp. 1406–1415.

[48] L. v. d. Maaten, G. Hinton, Visualizing data using t-sne, Journal of machine learning research 9 (11) (2008) 2579–2605.