Received March 17, 2021, accepted April 11, 2021, date of publication April 14, 2021, date of current version April 21, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3073249

SSNE: Effective Node Representation for Link Prediction in Sparse Networks

MING-REN CHEN[1], PING HUANG[1], YU LIN[2], AND SHI-MIN CAI[1,3,4]
[1]School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu 611731, China
[2]Research School of Computer Science, Australian National University, Canberra, ACT 2601, Australia
[3]Big Data Research Center, University of Electronic Science and Technology of China, Chengdu 611731, China
[4]Institute of Fundamental and Frontier Sciences, University of Electronic Science and Technology of China, Chengdu 611731, China

Corresponding author: Shi-Min Cai (shimin.cai81@gmail.com)

This work was supported in part by the National Natural Science Foundation of China under Grant 61673086, and in part by the Science Promotion Program of University of Electronic Science and Technology of China (UESTC), China, under Grant Y03111023901014006.

ABSTRACT Graph embedding is gaining popularity for link prediction in complex networks. However, few works focus on the effectiveness of graph embedding models on link prediction in sparse networks. This paper proposes a novel graph embedding model, **S**parse **S**tructural **N**etwork **E**mbedding (SSNE), to obtain node representation for link predication in sparse networks. The SSNE first transforms the adjacency matrix into the **S**um of **N**ormalized **H**-order **A**djacency **M**atrix (SNHAM) and then maps the SNHAM matrix into a d-dimensional feature matrix for node representation via a neural network model. The mapping operation is proved to be an equivalent variety of singular value decomposition. Finally, we calculate nodal similarities for link prediction based on the d-dimensional feature matrix. The extensive testing experiments based on artificial and real sparse networks suggest that the SSNE shows the effective node representation for link prediction in sparse networks, supported by the better link prediction performance compared to those of structural similarity indexes, matrix optimization, and other graph embedding models.

INDEX TERMS Link prediction, graph embedding, node representation, sparse network.

I. INTRODUCTION

IN natural complex systems, there are many entities, which interact with each other in a complicated way. By treating these entities as nodes and the corresponding interactive relationships as edges, we can abstract such systems into the network (or graph) model. Naturally, diverse types of complex networks are available to represent real complex systems, such as social networks, traffic networks, brain and biological networks, infrastructure networks, etc. [1]–[3]. Complex networks are continually evolving, and new connections between entities may occur in the future. Therefore, link prediction becomes an important task to study network structure's dynamic evolution [4]–[9].

In previous researches, a relatively simple link prediction framework is proposed based on the assumption that the greater the similarity between two nodes in the network, the greater the possibility of a connection between them [5]. Then, many similarity measurements of nodes have been proposed to compute similarity-based indexes for link prediction. A network contains a massive amount of structural information, which has been modeled as many similarity-based indexes, including the common neighbor (CN) index [10], the Adamic-Adar (AA) index [11], the resource allocation (RA) index [12], the Katz index [13], the restarted random walk (RWR) index [14], and the SimRank index [15], etc. These indexes can mainly be divided into two categories, local and global structural similarity indexes. The local structural similarity indexes (e.g., CN, AA, RA) only use the local topological information of nodes, which benefit from low computational complexity and become suitable for large-scale networks. However, their accuracy is slightly lower compared to that of the global structural similarity indexes (e.g., Katz, RWR, and SimRank), which considers the global topological information at a higher computational cost.

Graph embedding (i.e., graph representation) has been widely used in link prediction problems with representation learning development [16], [17]. Graph embedding can map a graph into low-dimension vector space, and at the

The associate editor coordinating the review of this manuscript and approving it for publication was Huiling Chen.

same time, keep the structure feature and inherent attribute of the graph [18]–[22]. Commonly, its pivotal is to sample enough structural information by random walks on a graph (or network). For example, DeepWalk [23] is one of the most popular random-walk-based graph embedding models. The link prediction method based on DeepWalk is shown to predict better the possible incidence of MicroRNA genetic disease [24], [25], as well as individual multiple interests or attributes [26], [27]. Although these embedding models succeed in link prediction in many natural networks, they involve critical experience-tuned parameters, such as the sampling length of a random walk and the number of random walks [23]. A typical scenario may only specify a locally maximum value within a finite interval of experience-tuned parameters. The error accumulation of multiple parameters would also hinder link prediction performance in sparse networks.

Therefore, in the framework of graph embedding, we propose a novel graph embedding model, **S**parse **S**tructure **N**etwork **E**mbedding (**SSNE**), to solve the problems mentioned above of random-walk-based graph embedding models. The SSNE includes two essential operations. The first is to transform the adjacency matrix of a sparse network into a general mutual information matrix based on the algorithm of the **S**um of **N**ormalized **H**-order **A**djacency **M**atrix (**SNHAM**), and the second is to map the SNHAM matrix into a d-dimensional feature matrix for effective node representation via a neural network model. The details will be introduced in **Section IV**. In further, we design experiments over various datasets to verify the effectiveness of SSNE for link prediction. The experimental results based on sparse networks show that the link prediction method based on SSNE outperforms other methods based on structural similarity indexes, matrix optimization, and other graph embedding models. As for relatively dense and better-structured networks, SSNE still shows comparable performance as structural similarity indexes, which is better than matrix optimization.

In short, in this paper, we make the following contributions:

- We propose a novel graph embedding model that overcomes the drawbacks in the prevail random-walk-based graph embedding models. The SNHAM algorithm is used to transform the adjacency matrix into a new matrix of theoretical co-occurrence probability between each pair of nodes, which substitutes the sampling method in random-walk-based graph embedding models. Meanwhile, we testify the mapping operation of the SNHAM algorithm to be an equivalent variation of the singular value decomposition (SVD), which significantly improves the computational efficiency of obtaining the feature matrix.
- We construct a link prediction method based on SSNE. The testing experiments' results based on six real networks and two types of artificial network models show the excellent performance of link prediction in sparse networks.
- We verify the algorithmic stability of link prediction method based SSNE by selecting different combinations of parameters. The results show that the proposed method is generally insensitive to parameters.

The remainder of this paper is organized as follows. In **Section II**, we briefly survey related work. **Section III** gives the problem definition. **Section IV** presents the whole framework of the link prediction method based on SSNE in detail. **Section V** contains experimental material and evaluation. **Section VI** presents the experimental result and discusses the effectiveness of adjustable parameters in link prediction performance. We finally conclude the paper in **Section VII**.

II. RELATED WORKS

This section briefly illustrates the related works in two aspects. On the one hand, we introduce some classical link prediction methods based on structural similarity index and discuss corresponding research achievements in recent years. On the other hand, we also discuss some popular graph embedding models based on representation learning for link prediction in complex networks.

A. LINK PREDICTION BASED ON STRUCTURAL SIMILARITY INDEX

The structural similarity indexes are defined by the similarity between two nodes based on their corresponding local or global structural measurements. The common structural similarity indexes include the CN index, the AA index, the RA index, the Katz index, the RWR index, the SimRank index, *etc.* (refer to [7] for more indexes). The CN index calculates the similarity of a pair of nodes by counting their number of common neighbors. The AA index [11] and RA index [12], based on the CN index, punish the common neighbors with large degrees respectively by the inverse logarithm and the reciprocal of common neighbors' degrees. The Katz index [13] can distinguish different neighboring nodes' influences and assign different weights to the neighboring nodes, causing the weights to be biased towards short paths. The RWR index is an alternative method of the PageRank algorithm and is defined by the probability of random walk between a pair of nodes in the steady-state [14]. The SimRank index [15] also involves the random walk process and measures how soon two random walkers, respectively starting from the endpoints, are expected to meet a certain node. These structural similarity indexes have been widely used to infer linking probability for link prediction in complex networks. Herein, we illustrate some very recent works on the link prediction methods based on structural similarity indexes.

Inspired by the above common structural similarity indexes, sever recent works synthesized more structural measurements to form complicated structural similarity indexes for link prediction. In [28], Zhou *et al.* replaced the degree with H index to form H-index-based link prediction methods, which significantly improve link prediction accuracy.

In [29], [30], Zhu *et al.* discussed the roles of degree, H-index, and coreness in link prediction in complex networks and formed a hybrid similarity index synthesizing degree and H-index. In [31], Pech *et al.* proposed a simple assumption that the likelihood of the existence of a link between two nodes can be unfolded by a linear summation of neighboring nodes' contributions and obtained the optimal likelihood matrix that shows remarkably better-predicting prediction performance.

B. LINK PREDICTION BASED ON GRAPH EMBEDDING

Graph embedding is used to map network structure into low-dimensional vector space indicated by a feature matrix of nodes reduced from an adjacency matrix. Based on the nodes' feature matrix, the similarity index is defined by the similarity between feature vectors of nodes. Thus, the link prediction method based on graph embedding strongly correlates with the graph embedding models. Herein, we illustrate some previous works on embedding graph models.

In recent years, graph embedding models have attracted more attention. In [23], Perozzi *et al.* proposed DeepWalk, where the random walk sampling processes produce linear sequences of nodes, and these sequences are used to calculate co-occurrence probabilistic matrix of nodes that are mapped into a feature matrix by a neural network model. In [32], Tang *et al.* explicitly defined two objective functions, 1st-order proximity and 2nd-order proximity, to obtain the topological information of network structure. They then used the linear combination of such proximity to represent the global proximity. In [33], Grover *et al.* proposed Node2Vec, which maintained the high order proximity between nodes by maximizing the probability of subsequent nodes in the random traversal graph. Compared with DeepWalk, Node2Vec has made some improvements in the random walk sampling process.

III. PROBLEM DEFINITION

A. PRE-DEFINITION

An undirected unweighted network is represented by $G = \langle V, E\rangle$ where the node set V={ v_1,v_2,...,v_n } and the edge set $E = \left\{e_{i,j}\right\} (i, j \in V)$. The dynamic evolution of network structure is represented by multiple snapshots of network, i.e., $\mathbb{G} = \{G_1, G_2, \ldots, G_{t-1}, G_t, \ldots, G_{N-1}, G_N\}$. At the current time t, $G_t =< V_t, E_t >$ denotes a snapshot of the network. Naturally, $G_{t-1} =< V_{t-1}, E_{t-1} >$ shows a previous snapshot of the network. We assume that the node set is stable, *i.e.,* $V_1 = V_2 = \cdots = V_{t-1} = V_t$, but the edge set is dynamically changing, which characterizes the dynamic evolution of network structure.

For simplicity, any two different nodes are indicated by symbols u and v, and the adjacency matrix of the network is expressed by symbol A. Obviously, if there exists an edge between nodes u and v, $A(u, v) = 1$, otherwise $A(u, v) = 0$. For a node u, its adjacency vector is $A_u^{1\times n} = A(u, :)$. We assume that the feature matrix R for node representations is obtained from the dimensionality reduction of the adjacency matrix. In a similar way, for a node u, its d-dimensionality vector is $R_u^{1\times d} = R(u, :)$.

We illustrate the important symbols involved in the model of SSNE. In the SNHAM algorithm, the output is defined by matrix $SNHAM \in \mathbb{R}^{n\times n}$, and the specific order is set as h. Because the elements of SNHAM matrix can reflect the co-occurrence probability for each pair of nodes, a node u has n-dimension vector of co-occurrence probability, $SNHAM_u^{1\times n} = SNHAM(u, :)$. In the single-hidden layer feedforward neural network model, the input is defined by the matrix $X \in \mathbb{R}^{n\times n}$, and the kernel and activation function between the input and hidden layers is respectively set as W_1 and $f_1(x)$; the output is defined by the matrix $Y \in \mathbb{R}^{n\times n}$, and the kernel and activation function between the hidden and output layers is respectively set as W_2 and $f_2(x)$; the feature matrix $R \in \mathbb{R}^{n\times d}$ of node representations is obtained in the hidden layer. The more details of symbol description are summarized in **Table 1**.

TABLE 1. Notation note of each symbol.

Symbol	Comments
u or v	denote a node
n	the number of nodes, i.e., the full dimension
d	the target of dimensionality reduction
h	the orders of SNHAM algorithm
t	the current time
$t-1$	the previous time
G_t	a snapshot of network at time t
V_t	the node set in network G_t
E_t	the edge set in the network G_t
W_i	the kernel of ith-layer network
$f_i(x)$	the activation function of ith-layer network
$\mathbb{G}$	multiple snapshots of network
A	the adjacency matrix of network
$A_u^{1\times n}$	the adjacency vector for node u
$SNHAM$	the output matrix of the SNHAM algorithm
$SNHAM_u^{1\times n}$	the SNHAM vector for node u
R	the feature matrix of node representations
$R_u^{1\times d}$	the d-dimension vector for node u
X	the input matrix of neural network model
$X_u^{1\times n}$	the input vector for node u
Y	the output matrix of neural network model
$Y_u^{1\times n}$	the output vector for node u

B. PROBLEM STATEMENT

Studying the whole dynamic evolution of network $\mathbb{G}$ is a complicated and challenging task. In order to simplify the process of the derivation, we herein only consider the relationship between the current t and previous time $t-1$, that is $\mathbb{G} = \{G_{t-1}, G_t\}$. Therefore, inferring dynamic evolution of network structure from $t-1$ to t realized by the link prediction based on G_{t-1} and G_t. The training set and test set can be set by G_{t-1} and $G_t - G_{t-1}$, respectively. Note that the real (benchmark) networks in the testing experiments aren't temporal (i.e., absent of the time scale). We thus assume the original network as G_t and hide a proportion of its edges to assume the residual network as G_{t-1}. Based on G_{t-1}, our task is to get the feature matrix of node representations that

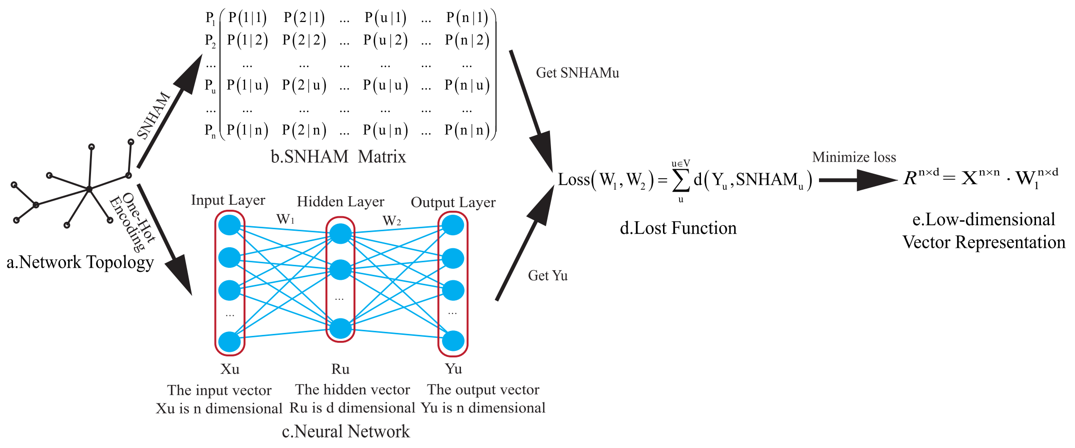

a.Network Topology
SNHAM
One-Hot Encoding
b.SNHAM Matrix
Get SNHAMu
Input Layer
Hidden Layer
Output Layer
W1
W2
Xu
Ru
Yu
The input vector Xu is n dimensional
The hidden vector Ru is d dimensional
The output vector Yu is n dimensional
c.Neural Network
Get Yu
Loss(W1, W2) = Σ d(Yu, SNHAMu)
d.Lost Function
Minimize loss
R^{n×d} = X^{n×n} · W1^{n×d}
e.Low-dimensional Vector Representation

FIGURE 1. (Color online) The schematic description of the SSNE. (a) An example of network topology; (b) The construction of SNHAM matrix; (c) The neural network model that is used to acquires the low-dimensional feature matrix; (d) The loss function of neural network model; (e) The acquirement of low-dimensional vector representation by minimizing the loss function.

meets the lower dimension. Still, it involves a large number of topological information of network structure and then applies the feature matrix to predict the hidden edges.

IV. SSNE FOR LINK PREDICTION

In this section, we describe the model of SSNE in detail. As shown in **Figure 1**, the SSNE consists of two steps. First, we introduce the SNHAM algorithm to obtain its corresponding matrix *SNHAM* that can reflect the theoretical values of co-occurrence probability between each pair of nodes. Then, we design a neural network model to calculate the corresponding co-occurrence probability (i.e., the output matrix Y). According to the difference between the matrices *SNHAM* and Y, the loss function is established. Using the stochastic gradient descent approach to minimize the loss function, we can get the optimal kernels and determine the feature matrix R in the hidden layer. However, the stochastic gradient descent approach has high time complexity in its iterative operation. We then find an alternative method to directly map the log(SNHAM) matrix into the feature matrix of node representations and demonstrate that the mapping operation is an equivalent variation of SVD. Finally, we apply the results of the SSNE to construct the similarity index for link prediction.

A. SNHAM MATIRX

Although using the random walk sampling process is effective in converting the topological information of network structure into linear sequences, it has been found that this method has obvious drawbacks. As mentioned above, graph embedding models based on random walk need to determine some random walk parameters, such as the sampling length of a random walk and the number of random walks, so that they are sensitive to such parameters. More importantly, we can only determine the random walk's empirically optimal parameters (i.e., local best at a finite parameter interval). Further, the finite-length linear sequences collected by the random walks have vital errors in representing the boundary nodes. Thus, the multiple parameters' accumulative errors significantly affect the accuracy of link prediction in complex networks. To solve the above problem existing in the graph embedding models based on a random walk, we propose the SNHAM algorithm to capture the network structure's topological information directly. We label the nodes in the network and order them to obtain an adjacency matrix A. First, we normalize the adjacency matrix by row to get the 1st-order transition probability matrix S_1. The row normalization function is set as Normal(X), so the above operation can be expressed by

$$S_1 = \text{Normal}(A). \tag{1}$$

In a similar way, we calculate the h-order transition probability matrix S_h by the h-order reachable matrix A^h ($h = 1, 2, \cdots, h$),

$$S_h = \text{Normal}\left(A^h\right), \text{ where } A^h = \overbrace{A \times A \cdots \times A}^{h}. \tag{2}$$

Then, we define the h-order similar probabilistic co-occurrence matrix $SPCO_h$, which is the sum of probability transition moments considering a restart. The restart probability is set as α. Thus, $SPCO_h$ is described by

$$SPCO_h = \sum_{i=1}^{h} \left((1-\alpha)S_i + \alpha S_1\right). \tag{3}$$

We consider that the restart process is excluded (i.e., $\alpha = 0$), that is, $SPCO_h$ can be reduced to the following form,

$$SPCO_h = \sum_{i=1}^{h} S_i. \tag{4}$$

Finally, we normalize the rows of $SPCO_h$ matrix, and the final result is denoted as the SNHAM matrix, which can be expressed as follows:

$$SNHAM = \text{Normal}\,(SPCO_h)\,. \tag{5}$$

The SNHAM algorithm can efficiently obtain the locally topological information of network structure and effectively solve the random walk sampling process's drawbacks. As the restart process is excluded, the adjusting parameter in the SNHAM algorithm is only the order h. The single parameter can avoid the random walk sampling process's accumulative errors of multiple parameters. Simultaneously, the SNHAM algorithm to obtain the network structure is no longer transforming the network structure into linear sequences of nodes. There don't exist errors in the process of representing the boundary node of each linear sequence. We show the pseudocode of the SNHAM algorithm in **Algorithm 1**.

Algorithm 1 SNHAM

```
Input: adjacency matrix $A$ of $G_{t-1}$; order index $h$
Output: SNHAM matrix $SNHAM$
1: Initializing an $n \times n$ matrix $SPCO$;
2: for each $i \in [1, h]$ do
3:     Calculating $A^i = \overbrace{A \times A \cdots \times A}^{i}$;
4:     Normalizing matrix $A^i$ by row, $S_i = \text{Normal}\left(A^i\right)$;
5:     $SPCO_h = \sum_{i=1}^{h} S_i$
6: end for
7: Normalizing matrix $SPCO_h$ by row, SNHAM= Normal
   ($SPCO_h$);
```

B. NEURAL NETWORK MODEL

The neural network is widely used to study multi-level feature representation, and the results obtained from representation learning are proved to be successfully applied in various fields. Herein, we use the single-hidden layer feedforward neural network model to construct high-quality and low-dimensional feature representation based on the SNHAM matrix. It is assumed to be a potential nonlinear mapping relationship between the vector representation space of the SNHAM matrix and the low-dimensional feature representation space. In SSNE, the single-hidden layer feedforward neural network model based on the SNHAM matrix is designed to calculate the co-occurrence probability matrix calculation.

Specifically, for a given node u, we input its one-hot coding vector $X_u^{1\times n}$, and map $X_u^{1\times n}$ into a low-dimensional vector $R_u^{1\times d}$ through the kernel W_1 and activation function $f_1(X) = $ X,

$$R_u^{1\times d} = f_1\left(X_u^{1\times n} \cdot W_1^{n\times d}\right) = X_u^{1\times n} \cdot W_1^{n\times d}. \tag{6}$$

Referring to the neural network model, the low-dimensional vector $R_u^{1\times d}$ is able to be mapped into the co-occurrence probabilistic vector $Y_u^{1\times n}$ through the kernel W_2 and activation function $f_2(X) = \text{Softmax(X)} = \frac{e^x}{\Sigma_i e^i}$,

$$Y_u^{1\times n} = \text{Softmax}\left(R_u^{1\times d} \cdot W_2^{d\times n}\right). \tag{7}$$

We use the theoretical co-occurrence probabilistic vector $SNHAM_u^{1\times n}$ of node u obtained from the SNHAM matrix and compare it with $Y_u^{1\times n}$ via the Euclid measurement. The loss function $L\,(W_1, W_2)$ is built by summing the errors across over all nodes,

$$L\,(W_1, W_2) = \sum_{u\in V} \text{d}\,(SNHAM_u, \text{Softmax}\,((X_u \cdot W_1) \cdot W_2))\,. \tag{8}$$

The kernels W_1 and W_2 are obtained through the stochastic gradient descent approach by minimizing the loss function. We focus the low-dimensional feature matrix in the hidden layer, which is described by

$$\text{R}^{n\times d} = X^{n\times n} \cdot W_1^{n\times d}. \tag{9}$$

As the stochastic gradient descent approach is high computational complexity, we provide an alternative method in the following subsection to improve the computational efficiency of obtaining a feature matrix significantly.

C. MINIMIZING $L(W_1, W_2)$ BY SVD

The above-mentioned optimization procedure of minimizing the loss function $L\,(W_1, W_2)$ is actually equivalent to make Softmax $((X \cdot W_1) \cdot W_2)$ approximate $SNHAM$ by adjusting the kernels W_1 and W_2. An ideal situation is that $L\,(W_1, W_2) = 0$, which satisfies the condition,

$$SNHAM_i = \text{Softmax}\,((X_i \cdot W_1) \cdot W_2)\,, \quad \text{where } i = 1, 2, \cdots, n. \tag{10}$$

We further simplify the variable $(X_i \cdot W_1) \cdot W_2$. Since the input matrix X encoded for the one-hot form is actually an identity matrix, we can write $W_1 \cdot W_2$ as the product matrix Z. Then, equation (10) can be rewritten as

$$SNHAM_i = \text{Softmax}\,(Z_i)\,, \text{ where } i = 1, 2, \cdots, n. \tag{11}$$

Supposing equation (11) has an inverse function, Z_i can be written as

$$Z_i = \text{Softmax}^{-1}\,(SNHAM_i)\,. \tag{12}$$

Naturally, the main task turns to obtain such inverse function. We set a input vector $x_i = \left(x_{i,1}, x_{i,2}, \ldots, x_{i,j}, \ldots, x_{i,n}\right)$, and the output vector via Softmax function is denoted as $y_i = \left(y_{i,1}, y_{i,2}, \ldots, y_{i,j}, \ldots, y_{i,n}\right)$. Without loss of generality, each value $x_{i,j}$ producing a corresponding $y_{i,j}$ satisfies an equation,

$$y_{i,j} = \text{Softmax}\left(x_{i,j}\right) = \frac{e^{x_{i,j}}}{\sum e^{x_{i,j}}}. \tag{13}$$

When the input vector is determined, $\sum e^{x_{i,j}}$ is a constant that is set as k_i. The conditions are satisfied with

$$\text{s.t.} \quad \begin{cases} \sum_{j=1}^{n} y_{i,j} = 1 \\ \sum_{j=1}^{n} e^{x_{i,j}} = k_i, \end{cases} \tag{14}$$

and used to obtain a variation of equation (13), then we can get the following formula,

$$x_{i,j} = \log\left(k_i \cdot y_{i,j}\right). \tag{15}$$

Inspired by equation (15), we assume the inverse function with a formula,

$$\text{Softmax}^{-1}\left(y_{i,j}\right) = x_{i,j} = \log\left(c_i \cdot y_{i,j}\right). \tag{16}$$

For a certain x_i, equation (16) is determined only when c_i is constant. In further, we verify the above-mentioned assumption. Equation (16) is generalized as

$$\begin{cases} x_{i,1} = \log\left(c_i \cdot y_{i,1}\right) \\ x_{i,2} = \log\left(c_i \cdot y_{i,2}\right) \\ \ldots\ldots \\ x_{i,n} = \log\left(c_i * y_{i,n}\right) \end{cases}, \tag{17}$$

which is equivalent to the following formula,

$$\begin{cases} e^{x_{i,1}} = c_i \cdot y_{i,1} \\ e^{x_{i,2}} = c_i \cdot y_{i,2} \\ \ldots\ldots \\ e^{x_{i,n}} = c_i \cdot y_{i,n} \end{cases}, \tag{18}$$

We sum the left and right terms in equation (18) and obtain the following formula,

$$\sum e^{x_{i,j}} = c_i \cdot \sum y_{i,j}. \tag{19}$$

According to the conditions in equation (14), we obtain $c_i = k_i$ from equation (19), which implies that for a certain x_i, c_i is a constant. Thus, the specific formula of the inverse function is independent of c_i. To make it easy to calculate, we set all k_i to 1 by assuming the independence of input vectors so that the inverse function is simplified as

$$\text{Softmax}^{-1}\left(y_{i,j}\right) = x_{i,j} = \log\left(y_{i,j}\right). \tag{20}$$

Turning to equation (12), it is specified as

$$Z_i = \text{Softmax}^{-1}\left(SNHAM_i\right) = \log\left(SNHAM_i\right). \tag{21}$$

Considering the zero value of co-occurrence probability in the SNHAM matrix, we uniformly add a very small positive σ ($\sigma = 10^{-8}$ in the testing experiments). We finally obtain the inverse function with the formula,

$$Z = \log(SNHAM + \sigma) = \log(SNHAM'). \tag{22}$$

Through equation (22), the specific matrix Z is also acquired.

We have known $Z = W_1 \cdot W_2$, and divide the matrix Z by SVD to get W_1, W_2 easily. The SVD procedure of $\log(SNHAM')$ is approximately equivalent to the optimization procedure of the neural network model. Without loss of generality, we denote the decomposition process as

$$\log(SNHAM') = U\Sigma V^T. \tag{23}$$

We choose the first d largest singular values, and approach $log(SNHAM')$ to $log(SNHAM')_d$, according to the following formula,

$$\log(SNHAM') \approx \log(SNHAM')_d = U_d\Sigma_d V_d^T \tag{24}$$

According to equation (24), we easily obtain $W_1 = U_d\Sigma_d$ and $W_2 = V_d^T$. Finally, according to equation (9), the d-dimensional feature matrix R can be expressed as

$$\mathrm{R}^{n\times d} = U_d\Sigma_d. \tag{25}$$

D. SIMILARITY INDEX BASED ON FEATURE MATRIX

After the original network topology is represented by the d-dimension feature matrix by the SSNE, we use such a feature matrix to construct a similarity index for link prediction. For any unknown edge $e_{u,v}$ between a pair of nodes u and v, its potential probability is quantified by these two nodes' similarity index. The similarity measurement is used by the Euclidean distance between the feature vectors of u and v, which is described as

$$D(e_{u,v}) = \sqrt{(x_{1v}-x_{1u})^2 + (x_{2v}-x_{2u})^2 + \cdots + (x_{dv}-x_{du})^2} \tag{26}$$

Considering the inverse correlation that the greater the distance is, the lower the similarity is, we take its reciprocal and add 1 to $D(e_{u,v})$ to prevent the case that $D(e_{u,v})$ is zero or too small. Finally, the similarity index is constructed by

$$S(e_{u,v}) = \frac{1}{1 + D(e_{u,v})}. \tag{27}$$

In the link prediction in complex networks, the higher the similarity index, the higher the potential possibility the unknown edge will be linked. We show the link prediction method's pseudocode based on SSNE in **Algorithm 2**.

Algorithm 2 Link Prediction Based on SSNE

```
Input: SNHAM Matrix $SNHAM$; dimension $d$
Output: Evaluation Index $AUC$
 1: Calculating $\log(\text{SNAHM} + \sigma)$;
 2: $\text{U}\Sigma V^T = SVD(\log(SPMI + \sigma))$
 3: Choosing $d$ largest singular value, $\text{U}_d\Sigma_d V_d^T \approx \text{U}\Sigma V^T$
 4: Obtaining feature matrix R, $R^{n\times d} = U_d\Sigma_d$
 5: Calculating Euclidean distance,
 6: $D(e_{u,v}) = \sqrt{(x_{1v}-x_{1u})^2 + \cdots + (x_{dv}-x_{du})^2}$
 7: Calculating similarity index, $S(e_{u,v}) = \frac{1}{1+D(e_{u,v})}$
 8: Initializing sampling parameter of AUC, $N = 672400$,
    $N' = 0, N'' = 0$;
 9: for each $i \in [1, N]$ do
10:     if $S_i\left(e_{a,b}\right) > S_i\left(e_{c,d}\right)$ then $N' += 1$
11:     else if $S_i\left(e_{a,b}\right) = S_i\left(e_{c,d}\right)$ then $N'' += 1$
12:     else other cases
13:     end if
14: end for
15: Calculating AUC, $AUC = \frac{N'+0.5\cdot N''}{672400}$
```

V. EXPERIMENTAL MATERIAL AND EVALUATION

We design testing experiments based on six real networks and two types of artificial network models to validate the effectiveness of SSNE for link prediction in complex networks.

In this section, the specific descriptions of real networks, two types of artificial networks, and the evaluation are illustrated, respectively.

A. REAL NETWORKS

We show six real networks that are described as:

Brain [34]: It is the neuronal connection network of a rhesus macaque. The nodes and edges represent neurons and fiber bundles among these neurons, respectively. In this network, there are 242 neurons, 3,054 fiber bundles, and the average degree of the network is 25.24.

Yeast [35]: It is the biological network in budding yeast. The nodes and edges represent proteins and interactions among these proteins. There are 2,375 proteins and 11,693 known interactions in this network, and the average degree of network is 9.85.

Air [34]: It is the traffic network of air control. The nodes and edges represent airports or service centers and the preferred air route among these airports or service centers recommended by the NFDC (National Flight Data Center). In this network, there are 1,226 airports or services centers, 2,410 preferred air routes, and the average degree of the network is 3.93

Road [34]: It is the road network in Minnesota state. The nodes and edge represent voluntary buildings and the direct road between these buildings. In this network, there are 2,642 buildings, and there are 3,303 direct roads, and the average degree of the network is 2.50.

Twitter [36]: It is the forwarding network of Twitter users about Obama's re-election as President of the United States in 2012. The nodes and edges represent twitter users and retweeting relationships between these users, respectively. There are 3,212 Twitter users in this network, 3,423 retweeting connections, and the network's average degree is 2.13.

Power [34]: It is the west power grid in the U.S. The nodes and edges represent substations or converters and high-voltage lines among these substations or converters. There are 4,941 substations or converters in this network in this network, 6,594 high-voltage lines, and the average degree of the network is 2.70.

We also summarize the basic topological information of six real networks, including the number of nodes and edges, the edge sparsity, the average degree, the clustering coefficient, and the degree heterogeneity, which are shown in **Table 2**.

TABLE 2. Basic topological information of six real networks. $|V|$ and $|E|$ indicate the number of nodes and edges, $\langle k \rangle$ is the average degree, ES is the edge sparsity, $\langle d \rangle$ is the average distance, C is the clustering coefficient, and $H = \frac{\langle k^2 \rangle}{\langle k \rangle^2}$ is the degree heterogeneity.

Nets	$\|V\|$	$\|E\|$	$\langle k \rangle$	ES	$\langle d \rangle$	C	H
Brain	242	3,054	25.24	0.1047	2.22	0.450	1.53
Yeast	2,375	11,693	9.85	0.0041	5.09	0.306	3.48
Air	1,226	2,410	3.93	0.0032	5.92	0.068	1.88
Road	2,642	3,303	2.50	0.0009	35.35	0.016	1.09
Twitter	3,212	3,423	2.13	0.0006	7.31	0.004	19.16
Power	4,941	6,594	2.67	0.0005	18.99	0.080	1.45

B. ARTIFICIAL NETWORK MODELS

We have known that the BA and WS networks models are widely used to simulate real complex networks because they characterize real complex networks' stylized facts. Herein, we show two types of artificial network models that are used in the following research, which are described as:

Barabasi-Albert network model [37]: The BA network model proposed by Barabasi and Albert characterizes the scale-free property of real complex networks. By using mean filed approximation, it can be proved that the resulted BA network has a power-law degree distribution with a scaling exponent 3. In the simulating process, the number of nodes and edges are adjustable according to the actual need.

Watts-Strogatz network model [38]: The WS network model proposed by Watts and Strogatz characterizes the small-world property of real complex networks. The resulted WS network has a larger cluster coefficient and shorter average distance. However, its degree distribution is Poisson. In the simulating process, the number of nodes and edges and the rewired probability are adjustable according to the actual need.

C. EVALUATION

The common measuring index for evaluating the link prediction method is AUC, which refers to the area under the receiver operating characteristic curve (ROC) [39]. In the AUC calculation, we needn't draw the specific ROC curve, especially when the samples are enormous. Rather than, we generally use the sampling method to obtain its approximate value. Once the partition of the training set and the testing set is determined, there are two kinds of unknown edges in the training set. One corresponds to the nonexistent edges (i.e., they don't exist in both training and testing sets). The other corresponds to the hidden edges (i.e., they only exist in the testing set). For a given link prediction method, each unknown edge is given a similarity index. AUC is equivalent to the probability that the similarity index of the randomly selected hidden edge in the testing set is higher than that of randomly chosen nonexistent edges [40].

So, we randomly select a hidden edge and a nonexistent edge in the testing set. If the similarity index of the hidden edge is higher than that of the nonexistent edge, the AUC value is added by 1. If these two similarity indexes are equal, the AUC value is added by 0.5. The sampling process is repeated with N times. We assume that there are N' and N'' times of the sampling processes that meet the two cases mentioned above, respectively. The AUC value is calculated as

$$AUC = \frac{N' + 0.5 \cdot N''}{N} \tag{28}$$

Note that a larger N makes the higher confidence of the measurement of AUC in equation (28). According to [41], when $N \geq 672400$, we can guarantee with 90% confidence that the absolute error of AUC will not exceed one-thousandth no matter of the network size. Thus, we set $N = 672400$ in the measurement of AUC.

VI. EXPERIMENTAL RESULT AND DISCUSSION

This section presents the link prediction method's performance based on SSNE and compares the proposed method with other baselines. The 20% edges of the current network G_t is hidden to obtain the previous network G_{t-1}. There are no isolated nodes in both G_{t-1} and G_t. Furthermore, we explore the effectiveness of adjustable parameters in the proposed method according to the experimental results based on real networks and artificial networks. Finally, we summarize the optimal AUC values obtained from the proposed method and the mainstream methods based on six real networks and two types of artificial network models.

A. LINK PREDICTION IN REAL NETWORKS

Herein, we first examine the link prediction method's performance based on SSNE and compare the proposed method with several mainstream methods based on structural similarity indexes and graph embeddings, such as CN, AA, RA, RWR, and DeepWalk. More other methods are shown in the following summary of the experimental result. AUC is used to evaluate the link prediction performance of these methods. The order h and dimension d are considered adjustable parameters, which regulate the link prediction method based on SSNE. Because the full dimension n is different from each network, d is dependent on n, *i.e.*, $d = p \cdot n$ for $p \in (0, 1)$. Note that p is an alternative parameter of d that indicates the proportion of dimension reduction to network size. **Figure 2** presents the performance comparison of different link prediction methods for six real networks. It suggests that except the Yeast, the link prediction method based on SSNE (short of $SSNE(h, p)$) behaves better than these mainstream methods.

More concretely, as shown in **Figure 2**, it is found that in these networks with the relatively large average degree

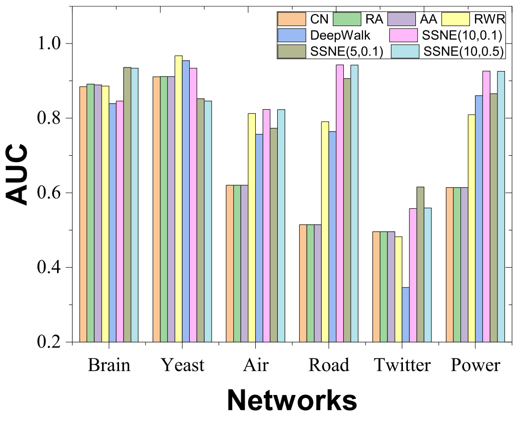


FIGURE 2. (Color online) Performance comparison of different link prediction methods for six real networks. Except the Yeast, the link prediction method based on SSNE (short of $SSNE(h, p)$) behaves better than these mainstream methods.

(e.g., **Brain**, **Yeast**), the link prediction performance of the proposed method is similar to that of the method based on DeepWalk. Both of them do not significantly outperform other methods based on the structural similarity index. However, when the average degree is relatively small (e.g., **Road**, **Twitter**, **Power**), the proposed method performs the best. Thus, we think that the proposed method is more suitable to solve the link prediction problem of sparse networks. Note that the artificial networks will further verify such observation in the following subsection.

At the same time, it is also found that the proposed method is affected by the adjustable parameters. We use different combinations of order h and proportion p to comprehensively analyze the link prediction for six real networks. **Figure 3** presents the influence of both h and p on the link prediction performance based on six real networks. The best AUC values of six real networks are 0.938 of **Yeast**, 0.856 of **Brain**, 0.834 of **Air**, 0.952 of **Road**, 0.616 of **Twitter**, and 0.928 of **Power**. We find that the proposed method is not particularly sensitive to the changes of h and p. More concretely, for a given h, the link prediction performance is nearly unchanged when p varies from 0.1 to 0.9. We can easily understand that the operation of SVD in SSNE causes such a phenomenon. There exists a critical d_c in each sparse network. The d_c-dimension feature matrix nicely represents the structural information. However, the acquirement of d_c in each sparse network brings high computational costs. We use p to uniformly set the corresponding dimensions of sparse networks for the simplicity of parameter computation. Even for $p = 0.1$, the corresponding d-dimension feature matrices can well represent the complete structural information of these sparse networks. While for a given p, the link prediction performance changes primarily when h gradually increases in a small range and then becomes approximately stable with the convergence of the SNHAM matrix, which implies that the SNHAM matrix with a small order (at least 10) contains most of the topological information of network structure. Furthermore, each network's results reveal a similar trend, which verifies the proposed method's stability. After the analysis mentioned above, we observe that when $h = 10$ and $p = 0.1$, the proposed method almost converges to the nearly optimal link prediction performance. It roughly suggests the default set of adjustable parameters can correspond to $h = 10$ and $p = 0.1$ for obtaining better link prediction performance.

B. LINK PREDICTION IN ARTIFICIAL NETWORKS

We also testify to the proposed method based on artificial networks. The artificial networks are generated by the BA and WS network models, respectively. Each type is composed of multiple artificial networks with various average degrees and sizes of nodes. Specifically, the sizes of nodes in the BA (or WS) networks vary from 1000 to 5000. For the BA (or WS) networks with the fixed size, their average degrees vary from 2 to 10 with a step-length two by adding edges which indicates the changes of edge sparsity. We try to study the

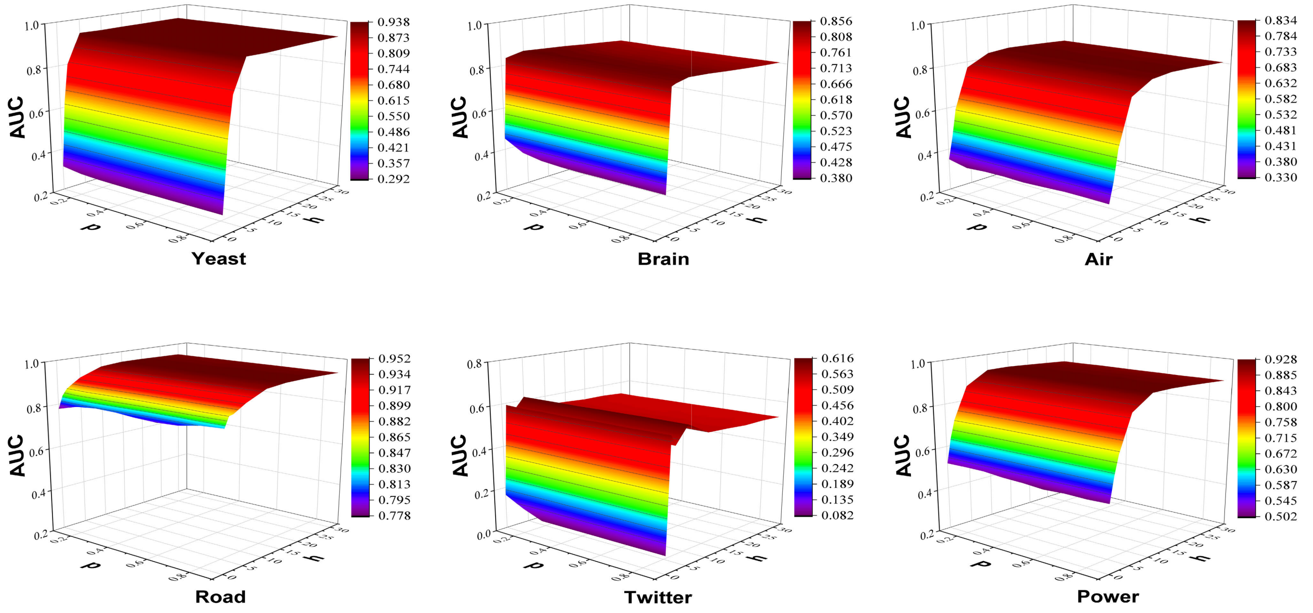


FIGURE 3. (Color online) The Influence of both the order *h* and the proportion *p* on the link prediction method based on SSNE. For each network, we show the AUC values in respect to different combinations of *h* and *p*. The results verify the stability of the proposed method because they share a similar trend in respect to *h* and *p*.

TABLE 3. The performance comparison of link prediction methods. The first and second-best AUC values are underlined by bold characters.

Nets	CN	Salton	Jaccard	AA	RA	RWR	LHN-I	LHN-II	Katz	SimRank	CLMC	DW	D2V	S2V	SSNE
Brain	0.884	0.885	**0.905**	**0.891**	0.889	0.886	0.771	0.651	0.886	0.754	0.784	0.839	0.793	0.652	0.846
Yeast	0.911	0.910	0.913	0.912	0.912	**0.967**	0.905	0.962	**0.967**	0.674	0.928	0.954	0.947	0.271	0.934
Air	0.620	0.619	0.626	0.620	0.620	0.813	0.619	0.781	0.813	0.674	0.621	0.757	**0.863**	0.350	**0.823**
Road	0.514	0.514	0.515	0.514	0.514	0.791	0.514	0.779	0.779	0.931	0.202	0.764	**0.940**	0.529	**0.942**
Twitter	0.496	0.496	**0.507**	0.496	0.496	0.483	0.496	0.465	0.483	0.432	0.479	0.346	0.469	0.135	**0.558**
Power	0.614	0.614	0.614	0.614	0.614	0.809	0.614	0.809	0.809	**0.919**	0.505	0.860	0.907	0.450	**0.927**
$BA(2)$	0.499	0.499	**0.500**	0.499	0.499	0.477	0.499	0.477	0.477	0.346	0.354	0.374	0.370	0.181	**0.581**
$BA(6)$	0.519	0.518	0.526	0.519	0.519	0.599	0.518	0.497	0.599	**0.646**	0.507	0.582	**0.640**	0.242	0.499
$WS(2)$	0.501	0.501	0.501	0.501	0.501	0.498	0.501	0.498	0.498	0.551	0.516	0.645	**0.693**	0.561	**0.713**
$WS(6)$	0.769	0.770	0.771	0.770	0.770	0.826	0.770	0.828	0.826	0.306	0.753	0.832	**0.835**	0.482	**0.843**

relationship between the network sparsity and link prediction performance (i.e., AUC) obtained from the proposed method.

Figure 4 presents the AUC values as a function of the average degree, which are obtained from the link prediction in the BA and WS networks. As shown in the left panel of **Figure 4**, we can see that the link prediction performance is better realized by the proposed method when the BA networks have a relatively smaller average degree and lower edge sparsity (e.g., $\langle k \rangle = 2$ and $N = 5000$). In particular, no matter the sizes of nodes, the AUC values are optimal when the average degrees of BA networks are $\langle k \rangle = 2$, which suggests that the proposed method is sensitive to the average degree. Meanwhile, as shown in the right panel of **Figure 4**, we can see that when the average degrees in the WS networks increase, the link prediction performance becomes much better, which is contrary to the results found in the BA networks. Nevertheless, when the average degree rises, the differences in link prediction performance between the proposed method and those based on structural similarity indexes become smaller (see in **Table 3**). In the sparse WS networks ($\langle k \rangle = 2$), the proposed method shows much better link prediction performance. Thus, to some extent, these results show that the proposed method is more suitable for link prediction in sparse networks.

C. DISCUSSION

As we comprehensively analyze the proposed method's link prediction performance based on real networks and artificial networks, we further discuss the performance comparison between the proposed method and more mainstream methods by conducting extensive experiments. Note that the average degrees of artificial networks are set as 2 and 6, and their sizes are both 5000 nodes.

Table 3 presents the performance comparison of all link prediction methods by the AUC values. The mainstream methods are divided into three types: structural similarity indexes including CN, Salton, Jaccard, AA, RA, RWR LHN-I, LHN-II, Katz SimRank, matrix optimization including CLMC [42], and graph embedding models including DeepWalk (DW), Diff2Vec (D2V) [43], Struc2Vec

TABLE 4. The running time (in seconds) of link prediction methods.

Nets	CN	Salton	Jaccard	AA	RA	RWR	LHN-I	LHN-II	Katz	SimRank	CLMC	DW	D2V	S2V	SSNE
Brain	105.0	102.0	97.0	107.2	106.9	0.1	0.1	0.1	0.1	20.8	1.3	23.5	14.1	16.6	16.0
Yeast	100.2	100.5	103.5	102.0	101.9	0.8	0.4	1.0	0.6	910.4	245.4	132.4	18.9	22.3	214.9
Air	87.6	87.3	85.2	89.1	88.6	0.3	0.2	0.3	0.2	74.9	34.8	73.0	15.2	17.4	68.3
Road	82.0	83.5	84.5	81.9	81.5	1.5	0.5	1.3	1.2	252.8	361.2	126.0	21.2	24.3	262.0
Twitter	84.2	83.7	85.7	86.1	84.4	1.6	0.7	2.0	1.2	356.7	593.9	141.1	24.1	27.8	384.8
Power	84.9	84.9	85.8	86.2	85.5	5.1	1.4	5.5	3.5	951.5	1832.4	237.8	38.8	42.6	889.8
$BA(2)$	86.1	85.5	88.7	88.5	87.3	6.5	2.2	8.0	5.2	1281.4	2620.8	274.8	51.0	59.3	1312.8
$BA(6)$	97.4	90.9	96.2	97.3	97.5	7.2	2.2	8.5	5.3	2437.2	2585.0	333.4	52.5	56.8	1329.2
$WS(2)$	81.9	83.6	80.8	83.0	81.4	6.5	2.5	8.3	5.3	1254.3	2624.3	254.4	50.5	57.4	1341.8
$WS(6)$	87.1	93.1	88.5	88.8	87.4	5.7	2.1	7.5	4.8	3294.3	2274.6	340.8	49.2	56.3	1328.2

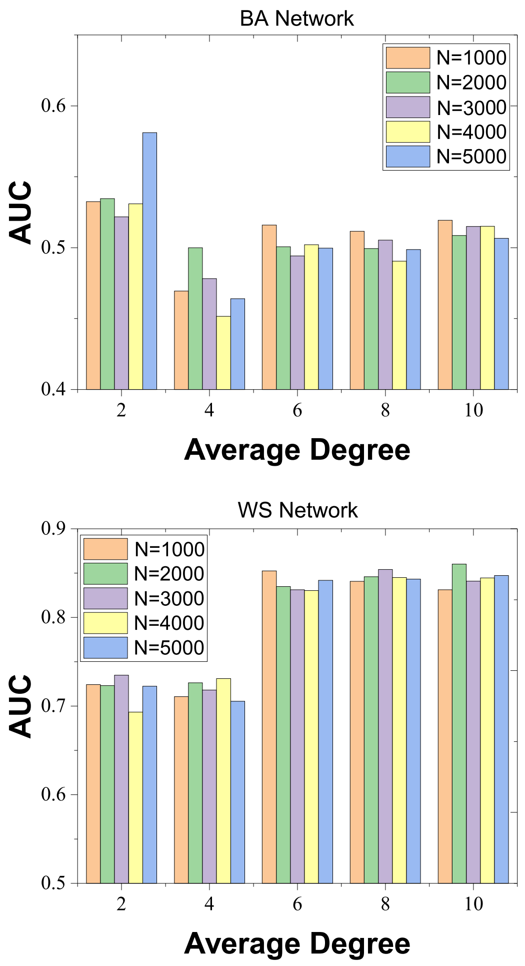


FIGURE 4. (Color online) The link prediction performance of the proposed method based on the BA and WS networks with different average degrees and sizes of nodes. In the upper panel, the AUC values as a function of the average degree show that the proposed method is much more suitable for the BA network with the relatively smaller average degree and lower edge sparsity. In the lower panel, the AUC values as a function of the average degree show that the result is contrary to that found in the BA networks.

(S2V) [44]. The specific parameter sets are illustrated: RWR with parameter $c = 0.8$; SimRank with parameter $\lambda = 0.8$; CLMC with parameters $\alpha_1 = 0.001$, $\alpha_2 = 0.01$ and $\alpha_3 = 100$; DeepWalk with parameters $Windows = 10$, $length = 40$, $times = 30$, $d = 128$; Diff2Vec with parameters $\alpha = 0.025$, $Windows = 10$, $vertexsetcardinality = 40$, $numdiffusions = 10$, $d = 128$; Struc2Vec with parameters $times = 20$, $Windows = 5$, $length = 40$, $d = 64$; SSNE with default parameters $h = 10$, $rate = 0.1$.

More concretely, in **Table 3**, the first and second-best AUC values are underlined by bold characters. We can find that for these networks with the relatively large average degree (e.g., **Brain** and **Yeast** the link prediction performance obtained by structural similarity indexes is better than the other two types of link prediction methods. However, except for the second-best AUC values in the Air network, we can see that the proposed method (i.e., SSNE) achieves the best AUC values for the link prediction in these real networks relatively small average degree. In artificial networks, we can see that the proposed method performs the best AUC values for the link prediction in these sparse artificial networks (i.e., $\langle k \rangle = 2$).

Finally, we quantitatively analyze each link prediction method's running efficiency via a personal computer with the 20 Intel(R) Xeon(R) CPU and 64G DDR4 memory. Note that the running time is directly presented to supplement experimental analysis. **Table 4** shows the running time of link prediction methods based on artificial and real sparse networks. It can be seen that the highest running time is 3294.3 seconds of SimRank, which suggests that realizing the link prediction in both artificial and real sparse networks is feasible. Besides, we discuss the running efficiency of link prediction methods. For these link prediction methods (except SimRank) based on structural similarity indexes, the running time is relatively stable and much less than that of link prediction methods based on matrix optimization and graph embedding models. The high running efficiency is because the running time mostly spends in the computation of structural similarity indexes. For CLMC and SSNE, the low running efficiency is because the running time mostly spends in the multiple iterations of matrix computation (e.g., the matrix computation in SNHAM algorithm). Note that the running time of DeepWalk, Diff2Vec, and Struc2Vec is much less because the pre-training time of node representation is neglected.

VII. CONCLUSION

As graph embedding is recently used for link prediction in complex networks, this paper proposes a novel link prediction method based on SSNE constructed in the framework of graph embedding. We comprehensively describe the procedure of SSNE from two aspects, the SNHAM matrix and the neural network model. The SNHAM matrix contains the h-order structural information of the adjacency matrix, and the neural network model is used to learn the d-dimensional representation of the SNHAM matrix. Through the SSNE, we can effectively obtain the graph representation of network structure. Note that the graph embedding procedure of SSNE is irrelevant to a specific network structure. Most importantly, in the SSNE, the adjustable parameters have been significantly reduced into two variables. Thus, the SSNE overcomes the random-walk-based graph embedding models' critical drawbacks by avoiding a directly random walk sampling process.

Meanwhile, to reduce the computational complexity of the neural network model, we assume that the optimization procedure of mining the loss function is equivalent to making the output matrix approximate the SNHAM matrix by adjusting the kernels of the neural network model. The product matrix of the kernels denotes the output matrix. Then, we formalize the association between the SNHAM matrix and the product matrix through the Softmax function. By verifying the inverse softmax function's assumption, we obtain the product matrix indicated by the logarithmic SNHAM matrix. Finally, we use the SVD to solve the product matrix and get the d-dimensional feature matrix.

The link prediction method based on the feature matrix is constructed by calculating the similarity indexes among feature vectors. We use six real networks and two types of artificial network models to test the proposed method's link prediction performance. The testing experiments are designed in three aspects. We first verify the proposed method's effectiveness on the link prediction in real diverse networks and the adjustable parameters' sensitivity to the proposed method. It has been found that the proposed method is more suitable for the link prediction in the relatively sparse network and only partially sensitive to the order of the SNHAM matrix. Then, the proposed method's effectiveness on the link prediction in the sparse network is further verified based on artificial networks. Finally, we discuss the comparison of the proposed method with a lot of mainstream methods based on structural similarity indexes, matrix optimization, and other graph embedding models. It suggests that the proposed method shows better link prediction performance in a relatively sparse network.

REFERENCES

[1] S. Boccaletti, V. Latora, Y. Moreno, M. Chavez, and D.-U. Hwang, "Complex networks: Structure and dynamics," *Phys. Rep.*, vol. 424, nos. 4–5, pp. 175–308, 2006.

[2] E. Bullmore and O. Sporns, "Complex brain networks: Graph theoretical analysis of structural and functional systems," *Nature Rev. Neurosci.*, vol. 10, no. 3, pp. 186–198, Mar. 2009.

[3] S. Boccaletti, G. Bianconi, R. Criado, C. I. del Genio, J. Gómez-Gardeñes, M. Romance, I. Sendiña-Nadal, Z. Wang, and M. Zanin, "The structure and dynamics of multilayer networks," *Phys. Rep.*, vol. 544, no. 1, pp. 1–122, Nov. 2014.

[4] G. Lise and D. Christopher, "Link mining: A survey," *ACM SIGKDD Explor. Newslett.*, vol. 7, no. 2, pp. 3–12, 2005.

[5] 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.

[6] R. N. Lichtenwalter, J. T. Lussier, and N. V. Chawla, "New perspectives and methods in link prediction," in *Proc. 16th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining (KDD)*, 2010, pp. 243–252.

[7] L. Lü and T. Zhou, "Link prediction in complex networks: A survey," *Phys. A, Stat. Mech. Appl.*, vol. 390, no. 6, pp. 1150–1170, Mar. 2011.

[8] V. Martínez, F. Berzal, and J.-C. Cubero, "A survey of link prediction in complex networks," *ACM Comput. Surv.*, vol. 49, no. 4, p. 69, 2016.

[9] S. Haghani and M. R. Keyvanpour, "A systemic analysis of link prediction in social network," *Artif. Intell. Rev.*, vol. 52, no. 3, pp. 1961–1995, Oct. 2019.

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

[11] L. A. Adamic and E. Adar, "Friends and neighbors on the Web," *Social Netw.*, vol. 25, no. 3, pp. 211–230, Jul. 2003.

[12] T. Zhou, L. Lü, and Y.-C. Zhang, "Predicting missing links via local information," *Eur. Phys. J. B*, vol. 71, no. 4, pp. 623–630, Oct. 2009.

[13] L. Katz, "A new status index derived from sociometric analysis," *Psychometrika*, vol. 18, no. 1, pp. 39–43, Mar. 1953.

[14] S. Brin and L. Page, "The anatomy of a large-scale hypertextual Web search engine," *Comput. Netw. ISDN Syst.*, vol. 30, nos. 1–7, pp. 107–117, Apr. 1998.

[15] G. Jeh and J. Widom, "SimRank: A measure of structural-context similarity," in *Proc. 8th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining (KDD)*, 2002, pp. 538–543.

[16] H. Xiao, M. Huang, and X. Zhu, "From one point to a manifold: Knowledge graph embedding for precise link prediction," 2015, *arXiv:1512.04792*. [Online]. Available: http://arxiv.org/abs/1512.04792

[17] R.-M. Cao, S.-Y. Liu, and X.-K. Xu, "Network embedding for link prediction: The pitfall and improvement," *Chaos, Interdiscipl. J. Nonlinear Sci.*, vol. 29, no. 10, Oct. 2019, Art. no. 103102.

[18] F. Luo, H. Huang, Y. Duan, J. Liu, and Y. Liao, "Local geometric structure feature for dimensionality reduction of hyperspectral imagery," *Remote Sens.*, vol. 9, no. 8, p. 790, Aug. 2017.

[19] Y. Gu, Y. Sun, Y. Li, and Y. Yang, "RaRE: Social rank regulated large-scale network embedding," in *Proc. 27th World Wide Web Conf. World Wide Web (WWW)*, 2018, pp. 359–368.

[20] J. Ni, S. Chang, X. Liu, W. Cheng, H. Chen, D. Xu, and X. Zhang, "Co-regularized deep multi-network embedding," in *Proc. 27th World Wide Web Conf. World Wide Web (WWW)*, 2018, pp. 469–478.

[21] G. H. Nguyen, J. B. Lee, R. A. Rossi, N. K. Ahmed, E. Koh, and S. Kim, "Continuous-time dynamic network embeddings," in *Proc. Companion Web Conf. Web Conf. (WWW)*, 2018, pp. 969–976.

[22] G. Shi, H. Huang, and L. Wang, "Unsupervised dimensionality reduction for hyperspectral imagery via local geometric structure feature learning," *IEEE Geosci. Remote Sens. Lett.*, vol. 17, no. 8, pp. 1425–1429, Aug. 2020.

[23] 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.

[24] G. Li, J. Luo, Q. Xiao, C. Liang, P. Ding, and B. Cao, "Predicting MicroRNA-disease associations using network topological similarity based on DeepWalk," *IEEE Access*, vol. 5, pp. 24032–24039, 2017.

[25] Z. Chen, X. Wang, P. Gao, H. Liu, and B. Song, "Predicting disease related microRNA based on similarity and topology," *Cells*, vol. 8, no. 11, p. 1405, Nov. 2019.

[26] Z. Jin, R. Liu, Q. Li, D. D. Zeng, Y. Zhan, and L. Wang, "Predicting user's multi-interests with network embedding in health-related topics," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Jul. 2016, pp. 2568–2575.

[27] Z. Yu, Z. Zhang, H. Chen, and J. Shao, "Structured subspace embedding on attributed networks," *Inf. Sci.*, vol. 512, pp. 726–740, Feb. 2020.

[28] W. Zhou, J. Gu, and Y. Jia, "H-index-based link prediction methods in citation network," *Scientometrics*, vol. 117, no. 1, pp. 381–390, Oct. 2018.

[29] X. Zhu, Y. Yang, L. Li, and S. Cai, "Roles of degree, H-index and coreness in link prediction of complex networks," *Int. J. Modern Phys. B*, vol. 32, no. 16, Jun. 2018, Art. no. 1850197.

[30] X. Zhu, W. Li, H. Tian, and S. Cai, "Hybrid influence of degree and H-index in the link prediction of complex networks," *EPL*, vol. 122, no. 6, p. 68003, Aug. 2018.

[31] R. Pech, D. Hao, Y.-L. Lee, Y. Yuan, and T. Zhou, "Link prediction via linear optimization," *Phys. A, Stat. Mech. Appl.*, vol. 528, Aug. 2019, Art. no. 121319.

[32] 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*, May 2015, pp. 1067–1077.

[33] 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.

[34] J. Kunegis, "KONECT: The Koblenz network collection," in *Proc. 22nd Int. Conf. World Wide Web*, 2013, pp. 1343–1350.

[35] V. Batagelj and A. Mrvar, "Pajek–program for large network analysis," *Connections*, vol. 21, no. 2, pp. 47–57, 1998.

[36] R. A. Rossi, D. F. Gleich, A. H. Gebremedhin, and M. M. A. Patwary, "What if CLIQUE were fast? Maximum cliques in information networks and strong components in temporal networks," 2012, *arXiv:1210.5802*. [Online]. Available: http://arxiv.org/abs/1210.5802

[37] A.-L. Barabási and R. Albert, "Emergence of scaling in random networks," *Science*, vol. 286, no. 5439, pp. 509–512, Oct. 1999.

[38] D. J. Watts and S. H. Strogatz, "Collective dynamics of 'small-world' networks," *Nature*, vol. 393, no. 6684, p. 440, 1998.

[39] J. A. Hanley and B. J. McNeil, "The meaning and use of the area under a receiver operating characteristic (ROC) curve," *Radiology*, vol. 143, no. 1, pp. 29–36, 1982.

[40] T. Fawcett, "An introduction to ROC analysis," *Pattern Recognit. Lett.*, vol. 27, no. 8, pp. 861–874, Jun. 2006.

[41] L. Lü and T. Zhou, "Link prediction: Appendix A.1 the selection of N in calculating AUC," Higher Educ. Press, pp. 261–264.

[42] Z. Zhang, D. Kang, C. Gao, and J. Shao, "SemiSync: Semi-supervised clustering by synchronization," in *Proc. Int. Conf. Database Syst. Adv. Appl.* Cham, Switzerland: Springer, 2019, pp. 358–362.

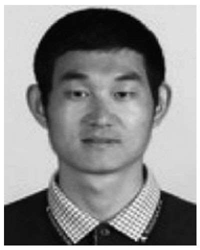
[43] B. Rozemberczki and R. Sarkar, "Fast sequence-based embedding with diffusion graphs," in *Proc. Int. Workshop Complex Netw.* Cham, Switzerland: Springer, 2018, pp. 99–107.

[44] L. F. R. Ribeiro, P. H. P. Saverese, and D. R. Figueiredo, "*struc2vec*: Learning node representations from structural identity," in *Proc. 23rd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2017, pp. 385–394.

MING-REN CHEN received the B.S. degree in information security from the University of Electronic Science and Technology of China, Chengdu, China, in 2019, where he is currently pursuing the M.S. degree in computer science and technology and the degree in computer science and engineering. His research interests include data mining, deep learning, and complex networks. His main awards and honors include Merit Student, the National Encouragement Scholarship, and Outstanding Graduates.

PING HUANG received the B.S. degree in electronic science and technology from Southwest Jiaotong University, Chengdu, China, in 2018. She is currently pursuing the M.S. degree in computer science and technology with the University of Electronic Science and Technology of China, Chengdu, and the degree in computer science and engineering. Her research interests include complex networks, information diffusion, and epidemic spreading. Her main awards and honors include Merit Student, the First Prize Scholarship, and Outstanding Graduates.

YU LIN received the Ph.D. degree in computer science from EPFL, Lausanne, Switzerland. He was a Postdoctoral Scholar with the Department of Computer Science and Engineering, University of California, San Diego. He is currently a Lecturer with the Research School of Computer Science, Australian National University. His research interests include algorithm design and computational biology.



SHI-MIN CAI received the B.S. degree in electrical engineering and automation from the Hefei University of Technology, in 2004, and the Ph.D. degree in circuit and systems from the University of Science and Technology of China, in 2009. He currently serves as an Associate Professor with the University of Electronic Science and Technology of China. He has published more than 100 high-level academic articles and hosted/participated nine national projects mainly supported by the National Natural Science Foundation of China. His research interests include complex network theory and its application for mining and modeling of real large-scale networked systems, time series analysis, and personalized recommendation systems.