SSNE: Effective Node Representation for Link Prediction in Sparse Networks

MING-REN CHEN¹, PING HUANG¹, YU LIN⁴, SHI-MIN CAI¹,²,³

¹ School of Computer Science and Engineering, University of Electronic Science and Technology of China, Chengdu 611731, China
² Institute of Fundamental and Frontier Sciences, University of Electronic Science and Technology of China, Chengdu 611731, China
³ Big data research center, University of Electronic Science and Technology of China, Chengdu 611731, China
⁴ Research School of Computer Science, Australian National University, Canberra, Australian

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

This work is supported by National Natural Science Foundation of China (No. 61673086) and the Science Promotion Programme of UESTC, China (No. Y0311023901014006).

ABSTRACT

Graph embedding is gaining its popularity for link prediction in complex networks and achieving excellent performance. However, limited work has been done in sparse networks that represent most of real networks. In this paper, we propose a model, Sparse Structural Network Embedding (SSNE), to obtain node representation for link prediction in sparse networks. The SSNE first transforms the adjacency matrix into the Sum of Normalized $H$-order Adjacency Matrix (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 variation of singular value decomposition. Finally, we calculate nodal similarities for link prediction based on such feature matrix. By extensive testing experiments bases on synthetic and real sparse network, we show that the proposed method presents better link prediction performance in comparison of 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 real 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 like social networks, traffic networks, brain and biological networks, infrastructure networks, etc [1]–[3]. Complex networks are constantly evolving over time and new connections between entities may occur in the future. Therefore, link prediction becomes an important task to study the dynamic evolution of network structure [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 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 huge 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 compare 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.

Link prediction has been studied by Markov chain and machine learning methods. For example, Sarukkai used Markov chain for link prediction and path analysis [16], and Zhu et al. extended Markov chain method to link prediction for adaptive networks [17]. With the development of representation learning, graph embedding (i.e., graph represen-
Graph embedding is able to map graph into low-dimension vector space, and at the same time, keep the structure feature and inherent attribute of graph. Commonly, its pivotal is to sample enough structural information by random walks on graph (or network). DeepWalk is one of the most popular random-walk-based graph embedding models. The link prediction method based on DeepWalk is shown to better predict the possible incidence of MicroRNA genetic disease, as well as individual multiple interests or attributes.

Although these random-walk-based graph embedding models succeed in link prediction in many real networks, they involve critical experience-tuned parameters, such like the sampling length of a random walk and the number of random walks. In a typical scenario, they may only specify a locally maximum value within a finite interval of experience-walks. In a typical scenario, they may only specify a locally maximum value within a finite interval of experience-walks. DeepWalk is shown to be better than matrix optimization and other graph embedding models, respectively. Meanwhile, we testify the mapping operation of SNHAM algorithm to be an equivalent variation of the SNHAM algorithm and is defined by the probability of random walk between a pair of nodes in the steady state. The SimRank index can distinguish different influences of different neighboring nodes 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. The SimRank index also involves with 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 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 the 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 the number of their common neighbors. The AA index and RA index, 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 can distinguish different influences of different neighboring nodes 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. The SimRank index also involves with 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 index for link prediction. In [28], Zhou et al. replaced the degree with H index to form H-index-based link prediction methods,
III. PROBLEM DEFINITION

A. PRE-DEFINITION

An undirected unweighted network is represented by $G = (V, E)$ where the node set $V = \{v_1, v_2, ..., v_n\}$ and the edge set $E = \{e_{ij}\} (i, j \in V)$. The dynamic evolution of network structure is represented by multiple snapshots of network, i.e., $G = \{G_1, G_2, ..., G_{t-1}, G_t, ..., G_N\}$. At current time $t$, $G_t = \langle V_t, E_t \rangle$ denotes a snapshot of network. Naturally, $G_{t-1} = \langle V_{t-1}, E_{t-1} \rangle$ shows a previous snapshot of 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 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 adjacency matrix. In the 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 are respectively set as $W_1$ and $f_1(z)$; 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 are respectively set as $W_2$ and $f_2(z)$; 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.

B. PROBLEM STATEMENT

Obviously, studying the whole dynamic evolution of network $G$ 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 $G = \{G_{t-1}, G_t\}$. 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 train set and test set can be set by $G_{t-1}$ and $G_t$, 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 which meets the lower dimension, but involves with large number of topological information of network structure, and then apply
the feature matrix for predicting the hidden edges.

IV. SSNE FOR LINK PREDICTION

In this section, we describe the model of SSNE in details. 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 error of loss function is established. By 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 to 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 parameters of the random walk, 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 empirically optimal parameters of the random walk (i.e., local best at a finite parameter interval). In further, the finite-length linear sequences collected by the random walks have large errors in the process of representing the boundary nodes. Thus, the accumulative errors of the multiple parameters greatly affect the accuracy of link prediction in complex networks.

To solve the above problem existing in the graph embedding models based on random walk, we propose the SNHAM algorithm which can directly capture the topological information of network structure. We label the nodes in the network and order them to obtain a 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 $\text{Normal}(X)$, so the above operation can be expressed by

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

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

$$S_h = \text{Normal}(A^h), \quad \text{where } A^h = \underbrace{A \times A \cdots \times A}_{h}. \quad (2)$$

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

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

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

$$SPCO_h = \sum_{i=1}^{h} S_i. \quad (4)$$

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:

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

The SNHAM algorithm can efficiently obtain the locally topological information of network structure, and effectively solve the drawbacks of random walk sampling process. As the restart process is excluded, the adjusting parameter in the SNHAM algorithm is only the order $h$. Obviously, the single parameter can avoid the accumulative errors of multiple parameters in the random walk sampling process. At the same time, the SNHAM algorithm to obtain the network structure is no longer to transform the network structure into linear sequences of nodes, so that there doesn’t exist the large 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 $G_{i-1}$; order index $h$

Output: SNHAM matrix $SNHAM$

1: Initializing an $n \times n$ matrix $SPCO$;
2: for each $i \in [1, h]$ do
3: \hspace{1em} Calculating $A^i = \underbrace{A \times A \cdots \times A}_{i}$;
4: \hspace{1em} Normalizing matrix $A^i$ by row, $S_i = \text{Normal}(A^i)$;
5: \hspace{1em} $SPCO_h = \sum_{i=1}^{h} S_i$
6: end for
7: Normalizing matrix $SPCO_h$ by row, $SNHAM = \text{Normal}(SPCO_h)$;

B. NEURAL NETWORK MODEL

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 different fields. Herein, we use the single-hidden layer feedforward
a. Network Topology

SNHAM

b. SNHAM Matrix

Input Layer
Hidden Layer
Output Layer

The input vector $X_u$ is n dimensional

The hidden vector $R_u$ is d dimensional

The output vector $Y_u$ is n dimensional

c. Neural Network

neural network model to construct high-quality and low-dimensional feature representation based on the SNHAM matrix because it is assumed to be a potential nonlinear mapping relationship between the vector representation space of 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 realize the calculation of co-occurrence probability matrix.

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 $f_1(X) = X$,

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

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 $f_2(X) = \text{Softmax}(X) = \frac{e^x}{\sum e^x}$.

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

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.

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

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

$$R_u^{n \times d} = X_u^{n \times n} \cdot W_1^{n \times n}. \quad (9)$$

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

C. SVD FOR LOG(SNHAM)

The above-mentioned optimization procedure of minimizing the loss function $L(W_1, W_2)$ is actually equivalent to make $\text{Softmax}((X \cdot W_1) \cdot W_2)$ 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

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

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

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

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

$$W_i = \text{Softmax}^{-1}(SNHAM_i). \quad (12)$$

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

$$y_{i,j} = \text{Softmax}(x_{i,j}) = \frac{e^{x_{i,j}}}{\sum e^{x_{i,j}}} \quad (13)$$

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

$$\sum_{j=1}^{n} y_{i,j} = 1\quad \text{s.t.} \quad \sum_{j=1}^{n} e^{x_{i,j}} = k_i \quad (14)$$

and used to obtain a variation of equation (13).

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

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

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

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,

$$\begin{align*}
x_{i,1} &= \log (c_i \cdot y_{i,1}) \\
x_{i,2} &= \log (c_i \cdot y_{i,2}) \\
\vdots \\
x_{i,n} &= \log (c_i \cdot y_{i,n}) \quad (17)
\end{align*}$$

which is equivalent to the following formula,

$$\begin{align*}
e^{x_{i,1}} &= c_i \cdot y_{i,1} \\
e^{x_{i,2}} &= c_i \cdot y_{i,2} \\
\vdots \\
e^{x_{i,n}} &= c_i \cdot y_{i,n} \quad (18)
\end{align*}$$

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

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

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 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

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

Turning to equation (12), it is specified as

$$W_i = \text{Softmax}^{-1}(SNHAM_i) = \log (SNHAM_i). \quad (21)$$

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

$$W = \log(SNHAM + \sigma) = \log(SNHAM'). \quad (22)$$

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

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

$$\log(SNHAM') = U\Sigma V^T \quad (23)$$

We choose the first $d$ largest singular value, and approach $\log(SNHAM')$ to $\log(SNHAM')d$

$$\log(SNHAM') \approx \log(SNHAM')d = U_d\Sigma_d V_d^T \quad (24)$$

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

$$R^{n \times d} = U_d\Sigma_d. \quad (25)$$

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 feature matrix to construct 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 the similarity index of these two node. The similarity measurement is used by the Euclidean distance between the feature vectors of $u$ and $v$, which is described as

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

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

$$S(e_{u,v}) = \frac{1}{1 + D(e_{u,v})} \quad (27)$$
In the link prediction in complex networks, it thinks that higher the similarity index is, the higher potential possibility the unknown edge will be linked with. We show the pseudocode of the link prediction method 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 $\log(SNAMH + \sigma)$

2: $\Sigma V^T = SD\log(SPMI + \sigma)$

3: Choosing $d$ largest singular value, $U_d \Sigma_d V_d^T \approx \Sigma V^T$

4: Obtaining feature matrix $\mathbf{R}$, $R_n \times d = U_d \Sigma_d$

5: Calculating Euclidean distance,

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

7: Calculating similarity index, $S(u,v) = \frac{1}{1 + D(u,v)}$

8: Initializing sampling parameter of AUC, $N' = 672400$

9: for each $i \in [1, N]$ do

10: if $S_i(u,a) > S_i(u,b)$ then $N' = 1$

11: else if $S_i(u,a) = S_i(u,b)$ 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 the link prediction in complex networks. In this section, the specific descriptions of real networks, two types of artificial network models 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 macaques or rhesus. 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. In this network, there are 2,375 proteins and 11,693 known interactions, 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 preferred air route among these airports or service centers recommended by the NFDC (National Flight Data Center), respectively. In this network, there are 1,226 airports or service centers, 2,410 preferred air routes, and the average degree of the network is 3.93.

**WS** [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. In this network, there are 3,212 twitter users, 3,423 retweeting relationships between these users, and the average degree of the network 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. In this network, there are 4,941 substations or converters, 6,594 high-voltage lines, and the average degree of the network is 2.70.

**Road** [34]: It is the road network in Minnesota state. The nodes and edge represents voluntary buildings and 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** [35]: It is the biological network in budding yeast. The nodes and edges represent proteins and interactions among these proteins. In this network, there are 2,375 proteins and 11,693 known interactions, and the average degree of network is 9.85.

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 the stylized facts of real complex networks. 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 an scale 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.

| Nets   | $|V|$ | $|E|$ | $\langle k \rangle$ | $ES$ | $|d|$ | $C$ | $H$ |
|-------|------|------|----------------------|-----|------|-----|-----|
| 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 |
well as the rewired probability are adjustable according to the actual need.

C. EVALUATION

The common measuring index for evaluating link prediction method is AUC that refers to the area under the curve of the receiver operating characteristic (ROC) [39]. In the calculation of AUC, we don’t draw the specific ROC curve, especially when the samples are very large. Alternatively, we generally use the sampling method to obtain its approximate value. Once the partition of training set and the testing set is determined, there are two kinds of unknown edges in the training set, that is, one corresponds to the nonexistent edges (i.e., they don’t exist in both training and testing sets) and 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 randomly selected hidden edge in the testing set is higher than that of randomly selected nonexistent edge [40].

That is to say, we randomly select a hidden edge and a nonexistent edge in the testing set. If the similarity index of hidden edge is higher than that of nonexistent edge, the AUC value is added by 1, and 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 above-mentioned two cases, respectively. The AUC value is calculated as,

$$AUC = \frac{N' + 0.5 \cdot N''}{N}$$  \hspace{1cm} (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.

VI. EXPERIMENTAL RESULT AND DISCUSSION

In this section, we mainly present the performance of link prediction method based on SSNE, and compare the proposed method with other baselines. The 20% edges of 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 performance of link prediction method based on SSNE, and compare the proposed method with several mainstream methods based on structural similarity index and graph embedding, such as CN, AA, RA, RWR and DeepWalk. More other methods are shown in the following summary of experimental result. AUC is used to evaluate the link prediction performance of these methods. The order $h$ and dimension $d$ are considered as the 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., $d = p \times n$. 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.

![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.](image)

More concretely, as shown in Figure 2, it is found that in these network with relatively large average degree (e.g., Brain, Yeast), the link prediction performance of the proposed method is similar to that of method based on DeepWalk, and both of them do not significantly outperform other methods based on 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 network. Note that such observation will be further verified by the artificial networks 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, which suggests that the size of dimension reduction has little influence on the proposed
method. While for a given $p$, the link prediction performance changes largely when $h$ gradually increases in a small range and then becomes approximately stable with the convergence of 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, the results of each network reveal a similar trend, which verify the stability of the proposed method. After the above-mentioned analysis, we observe that when $h = 10$ and $p = 0.1$, the proposed method almost converges to the 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 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 different average degrees and sizes of nodes. Specifically, the sizes of nodes in the BA (or WS) networks vary from 1000 to 5000, and for the BA (or WS) networks with the fixed size, their average degrees vary from 2 to 10 with a step-length 2 by adding edges which indicates the changes of edge sparsity. We try to study the 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 relatively smaller average degree and lower edge sparsity (e.g. $\langle k \rangle = 2$ and $N = 5000$). In particular, no matter of 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 become much better, which is contrary to the results found in the BA networks. Nevertheless, when the average degrees increase, the differences of link prediction performance between the proposed method and those methods based on structural
similarity indexes become smaller (see in Table 3). That is, in the sparse WS networks \((k) = 2\), the proposed method show much better link prediction performance. Thus, these results to some extent show that the proposed method is more suitable for the link prediction in sparse networks.

C. DISCUSSION

As we comprehensively analyze the link prediction performance of the proposed method 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, each of them is based on structural similarity indexes including CN, Salton, Jaccard, AA, RA, RWR LHN-I, LHN-II, Katz, SimRank, matrix optimization including CLMC [42] and other graph embedding models including DeepWalk (DW), Diff2Vec (D2V) [43], Struc2Vec (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 relatively large average degree (e.g., Brain and Yeast) the link prediction performance obtained by structural similarity indexes is better than other two types of link prediction methods. However, except 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 with relatively small average degree. And, in artificial networks, we can see that the proposed method performs the best AUC values for the link prediction in these artificial sparse networks (i.e., \((k) = 2\)).

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 that is 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 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 specific network structure. Most importantly, in the SSNE, the adjustable parameters have been extremely reduced into two variables. Thus, the SSNE overcomes the critical drawbacks of the random-walk-based graph embedding models by avoiding directly random walk sampling process.

Meanwhile, to reduce the computational complexity of neural network model, we assume that the optimization procedure of mining the loss function is actually equivalent to make the output matrix approximate the SNHAM matrix by adjusting the kernels of neural network model. The output matrix is denoted by the product matrix of the kernels. Then, we formalize the association between the SNHAM matrix and the product matrix through the Softmax function. By verifying the assumption of the inverse Softmax function, we obtain the product matrix indicated by the logarithm SNHAM matrix. Finally, we use the SVD to solve the product matrix and obtain the \(d\)-dimensional feature matrix.

The link prediction method based on 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 link prediction performance of the proposed method. The testing experiments are designed in three aspects. We first verify the effectiveness of the proposed method on the link prediction in diverse real networks and the sensitivity of the adjustable parameters to the proposed method. It has found that the proposed method is more suitable for the link prediction in relatively sparse network and only partially sensitive to the order of the SNHAM matrix. Then, the effectiveness of the proposed method on the link prediction in sparse network is further verified based on artificial networks. Finally, we discuss the results of the comparison of 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 show better link prediction performance in relatively sparse network.

REFERENCES

[1] S. Boccaletti, V. Latora, Y. Moreno, M. Chavez, and D.-U. Hwang, “Complex networks: Structure and dynamics,” Physics Reports, vol. 424, no. 4-5, pp. 175–308, 2006.
[2] E. Bullmore and O. Sporns, “Complex brain networks: graph theoretical analysis of structural and functional systems,” Nature Reviews Neuroscience, vol. 10, no. 3, pp. 186–198, 2009.
[3] S. Boccaletti, G. Bianconi, R. Criado, C. I. Del Genio, J. Gómez-Gardenes, M. Romance, I. Sendina-Nadal, Z. Wang, and M. Zanin, “The structure and dynamics of multi-layer networks,” Physics Reports, vol. 544, no. 1, pp. 1–122, 2014.
[4] L. Getoor and C. P. Diehl, “Link mining: a survey,” ACM SIGKDD Explorations Newsletter, vol. 7, no. 2, pp. 3–12, 2005.
[5] D. Liben-Nowell and J. Kleinberg, “The link-prediction problem for social networks,” Journal of the American Society for Information Science and Technology, 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 Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. ACM, 2010, pp. 243–252.
[7] L. Li and T. Zhou, “Link prediction in complex networks: A survey,” Physica A, vol. 390, no. 6, pp. 1150–1170, 2011.
TABLE 3: The performance comparison of link prediction methods

| Net  | CN  | Salton | Jaccard | AA  | RA  | RWR  | LHN-I | LHN-II | Katz | SimRank | CLMC | DW  | 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 |
| East  | 0.911 | 0.910 | 0.913 | 0.912 | 0.912 | **0.907** | 0.905 | 0.962 | **0.907** | 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 | **0.983** | 0.350 | **0.923** |
| Road  | 0.514 | 0.514 | 0.514 | 0.514 | 0.779 | 0.779 | 0.931 | 0.202 | 0.764 | 0.940 | 0.529 | **0.942** | 0.505 | 0.779 | 0.884 |
| 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** |
| B(A)2 | 0.499 | 0.499 | **0.500** | 0.499 | 0.499 | 0.497 | 0.499 | 0.477 | 0.477 | 0.346 | 0.354 | 0.374 | 0.370 | 0.181 | **0.581** |
| B(A)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 |
| W(S)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** |
| W(S)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 | **0.543** |