Noise-Resilient Similarity Preserving Network Embedding for Social Networks

Zhenyu Qiu[1,2] , **Wenbin Hu**[1,4*] , **Jia Wu**[3] , **ZhongZheng Tang**[2] and **Xiaohua Jia**[2]
[1]School of Computer Science, Wuhan University
[2]Department of Computer Science, City University of Hong Kong
[3]Department of Computing, Macquarie University
[4]Shenzhen Research Institute, Wuhan University, China
{qiuzy, hwb}@whu.edu.cn, jia.wu@mq.edu.au, tangzhongzheng@amss.ac.cn, csjia@cityu.edu.hk

Abstract

Network embedding assigns nodes in a network to low-dimensional representations and effectively preserves the structure and inherent properties of the network. Most existing network embedding methods didn't consider network noise. However, it is almost impossible to observe the actual structure of a real-world network without noise. The noise in the network will affect the performance of network embedding dramatically. In this paper, we aim to exploit node similarity to address the problem of social network embedding with noise and propose a node similarity preserving (NSP) embedding method. NSP exploits a comprehensive similarity index to quantify the authenticity of the observed network structure. Then we propose an algorithm to construct a correction matrix to reduce the influence of noise. Finally, an objective function for accurate network embedding is proposed and an efficient algorithm to solve the optimization problem is provided. Extensive experimental results on a variety of applications of real-world networks with noise show the superior performance of the proposed method over the state-of-the-art methods.

1 Introduction

Network embedding is an important method to learn low-dimensional representations of a network through the study of its high-dimensional representation, while preserving the network's structure and its inherent properties. Based on network embedding, mining information in networks, such as link prediction [Yu *et al.*, 2017], classification [Liu *et al.*, 2017], and network visualization [Tang *et al.*, 2015], can be directly conducted in the low-dimensional space.

Given a network $G = (V, E)$, the network embedding is a mapping $f = v_i \to \mathbf{y}_i \in \mathbb{R}^d$. The function f maps each node v_i in the node set V to d-dimensional space [Goyal and Ferrara, 2018]. Various methods of network embedding have been proposed [Cui *et al.*, 2017] and many network properties such as first-order proximity [Wang *et al.*, 2016], high-order proximity [Zhu *et al.*, 2018], and social community [Zhang *et al.*, 2018] have been considered. Essentially, most of methods didn't consider network noise. However, the observed data always contains noise. That is, false links always exist in observed network data and many actual links are missing during network observation. We call this problem *network embedding with noise*. The noise in the network will affect the performance of network embedding dramatically. Fig. 1 presents an example to illustrate the incorrect embedding of a network due to noise.

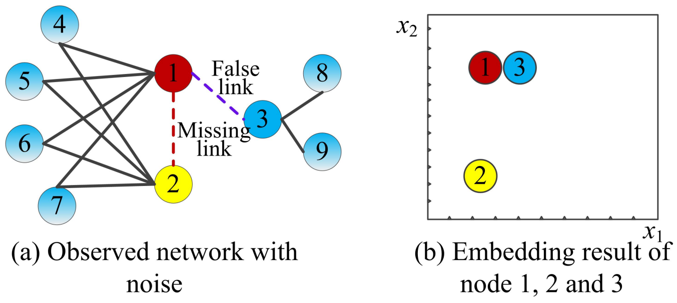
4
5
6
7
1
2
3
8
9
False link
Missing link
x_2
x_1
(a) Observed network with noise
(b) Embedding result of node 1, 2 and 3

Figure 1: Example of network noise affecting embedding. In subgraph (a), the blue dashed line between nodes 1 and 3 is a false link and the red dashed line between nodes 1 and 2 is a missing link. The subgraph (b) presents the incorrect embedding results.

To some extent, the effect of network noise can be reduced and the actual network structure can be recovered by node similarity. In Fig. 1 (a), according to the common neighbor (CN) index method [Liben-Nowell and Kleinberg, 2007], nodes 1 and 2 have high similarity because they share many common neighbors, thus there is a high probability that a link exists between nodes 1 and 2. On the contrary, the probability of the existence of a link between nodes 1 and 3 is low because they have low similarity according to the CN index. As a result, the false link and the missing link in Fig. 1 (a) can be detected by node similarity. Intuitively, we aim to exploit node similarity to quantify the authenticity of the observed network structure and reduce network noise. The greater the similarity between two nodes, the higher the authenticity of the link between them; therefore, they should be represented closely to each other, and vice versa.

In this paper, we propose a node similarity preserving (NSP) embedding method to reduce the influence of network

*Corresponding Author

noise through the analysis of node similarity. In general, node similarity is quantified by a single similarity index, and many such indices have been proposed [Hu *et al.*, 2017]. However, existing single indices cannot fully describe node similarity in complex real-world social networks [Wang *et al.*, 2017a]. In view of this, we propose a comprehensive similarity index to quantify node similarity accurately. For a given network, a similarity matrix is constructed from the adjacency matrix by using the comprehensive similarity index. Then we propose a correction matrix that can reduce the noise in the adjacency matrix by the analysis of the similarity matrix. Finally, we formulate an objective function for accurate network embedding and propose an efficient algorithm based on interactive search to solve the optimization problem. As a result, the learned network representation can preserve the actual network structure well. The contributions of this paper can be summarized as follows:

- We propose a comprehensive similarity index to measure node similarity and compute a node similarity matrix. Compared to existing single similarity indices, the proposed index is more accurate.
- We propose an algorithm to construct a correction matrix to reduce the influence of noise on a given network and recover the actual network structure.
- We formulate an objective function for accurate network embedding by introducing a weight matrix and a penalty function. We also propose an efficient algorithm to solve the optimization problem of network embedding.

2 Related Work

Various methods of network embedding have been proposed in the literature. For example, DeepWalk [Perozzi *et al.*, 2014] aims to preserve the neighbor structure of nodes and uses the SkipGram model to learn the representations of nodes. Node2Vec [Grover and Leskovec, 2016] is able to learn the representations that closely embed nodes that belong to the same network community. LINE [Tang *et al.*, 2015] is intended for large-scale network embedding, and can preserve both first-order and second-order proximities. Community structure is also an important property in network representation. [Wang *et al.*, 2017c] propose a modularized nonnegative matrix factorization (M-NMF) model for network embedding that preserves both the microscopic structure and the mesoscopic (community) structure. In addition, some other network properties such as non-transitivity [Ou *et al.*, 2015], asymmetric transitivity [Ou *et al.*, 2016], node side information [Yang *et al.*, 2015], and structure balance [Wang *et al.*, 2017b] have been embedded into representation vectors. Despite the success of these network embedding approaches, they all assume that there is no noise in the observed network data. Therefore, a method to reduce the influence of noise is still lacking.

3 Problem Definition

A social network is defined as $G = (V, E)$, where $V = \{v_1, ..., v_n\}$ is the node set and $E = \{e_1, ..., e_m\}$ is the link set. The adjacency matrix is denoted as $\mathbf{A} \in \mathbb{R}^{n\times n}$, where $\mathbf{A}_{i,j} \in [0, 1]$ is the link weight between nodes v_i and v_j. The value of $\mathbf{A}_{i,j}$ represents the connection strength between nodes v_i and v_j. If there is no link between v_i and v_j, then $\mathbf{A}_{i,j} = 0$. Our task is to find an approach to reduce the influence of noise and embed each node in V into a point in the embedding space. Let $\mathbf{S} \in \mathbb{R}^{n\times n}$ denote the node similarity matrix: $\mathbf{S}_{i,j} \geq 0$ denotes the similarity between v_i and v_j. Let $\mathbf{C} \in \mathbb{R}^{n\times n}$ denote the correction matrix. Let $\mathbf{U} \in \mathbb{R}^{n\times d}$ denote the embedding matrix. The i-th row of $\mathbf{U}$, $\mathbf{U}_{i,*}$, is the embedding vector of v_i. We first need to derive $\mathbf{S}$ from $\mathbf{A}$ by using a comprehensive similarity index, so that $\mathbf{S}$ can preserve the accurate similarity between nodes. Since $\mathbf{A}$ contains noise, we need to compute $\mathbf{C}$ based on $\mathbf{S}$ and $\mathbf{A}$ for noise reduction. Finally, we compute $\mathbf{U}$, such that the learned representations can preserve the actual network structure.

4 Node Similarity Preserving Embedding

The NSP method has three modules: 1) node similarity construction, which generates a comprehensive similarity index to evaluate node similarity and constructs the node similarity matrix $\mathbf{S}$; 2) after node similarity constriction, a noise reduction module is proposed, which computes a correction matrix $\mathbf{C}$ based on $\mathbf{S}$ to reduce network noise; 3) finally, a node similarity embedding module is proposed, which formulates an objective function for accurate network embedding by approximating the $\mathbf{S}$ and $\mathbf{C}$, and proposes an efficient algorithm to solve the optimization problem.

4.1 Node Similarity Construction

Node similarity is an important measurement to quantify the strength of node relationship and is used to predict links between nodes [Yu *et al.*, 2017]. In general, node similarity is quantified by a similarity index. Most similarity indices are derived from specific evolution mechanisms. However, real-world networks are dynamic and jointly driven by a hybrid mechanism [Zhang *et al.*, 2014]. A single index cannot fully describe the node similarity. To evaluate node similarity accurately, we propose a comprehensive similarity index S that considers multiple single similarity indices. The definition of S is as follows:

Definition 1. (Similarity index vector). *The similarity index vector* $S_v = \{s_1, ..., s_i, ..., s_\gamma\}$ *is a vector consisting of* γ *single similarity indices. For example,* s_i *could be the CN index. For any node pair* $\{v_i, v_j\}$*, the similarity index vector is* $S_v(v_i, v_j) = \{s_1(v_i, v_j), ..., s_\gamma(v_i, v_j)\}$.

Since each single similarity index s_i in S_v plays different weight in the overall similarity measurement, we introduce a weight vector for S_v.

Definition 2. (Similarity index weight). *The similarity index weight* $\phi = \{\varphi_1, ..., \varphi_i, ..., \varphi_\gamma\}$ *is the weight vector of* S_v. φ_i *denotes the weight of index* s_i *in* S_v.

Definition 3. (Comprehensive similarity index). *Given* S_v *and* ϕ*, the comprehensive similarity index* $S(v_i, v_j)$ *between nodes* v_i *and* v_j *is defined as:*

$$S(v_i, v_j) = \sum_{s_k\in S_v} \varphi_k \frac{s_k(v_i, v_j) - min(s_k)}{max(s_k) - min(s_k)}, \qquad (1)$$

where $min(s_k)$ is the minimum s_k value of all node pairs in the network, and $max(s_k)$ is the maximum value.

A real-world social network is driven by a hybrid mechanism. Experimental analysis in [Zhang *et al.*, 2014] indicates that the relative contribution of each mechanism varies in the evolution of different social networks. While a single similarity index s_i is derived from a specific evolution mechanism. When a similarity index is consistent with the network evolution mechanism, the index will provide accurate node similarity evaluation [Hu *et al.*, 2017]. Therefore, to make the comprehensive similarity index S more consistent with the hybrid network evolution mechanism, we need to determine the optimal weight value of each single similarity index s_i in S_v. A ϕ whose corresponding S achieves the most accurate node similarity evaluation is defined as the optimal index weight, ϕ^*. For a given social network, we need to optimize the ϕ^* to evaluate node similarity accurately.

Note that the S_v can include any existing single similarity index. To enable the S to be applied to large-scale networks, a sampling step is included in NSP. In detail, the node similarity construction phase consists of the following two steps:

Step1: Network Sampling

We sample from the original network with the alias method [Li *et al.*, 2014] according to the link weights, and treat the sampled links as binary links. The sampling is conducted by the following steps.

1) Initialize the sampled network $G' = (V', E')$, where $v' \subset V$, $E' \subset E$.
2) According to the link weights, sample a link e_i from G by the alias method [Li *et al.*, 2014]. If e_i is in E', then discard it, else set the weight of e_i to 1 and add e_i to E'. For the nodes v_i, v_j linked by e_i, if they do not exist in V', then add them to V'.
3) Repeat 2) until $|E'| = K$. K is a control parameter of the sampled network size.

Step2: Searching For Optimal Index Weight

After the network sampling, we need to search the optimal ϕ^* according to the obtained G'.

A good similarity index should provide accurate link prediction [Hu *et al.*, 2017]. To search for the optimal ϕ^*, we adopt the AUC [Hu *et al.*, 2017] as the metric of prediction accuracy. Let $AUC(\phi)$ denote the prediction accuracy of the S corresponding to the ϕ. The problem turns into searching for the ϕ^* such that the corresponding $AUC(\phi^*)$ is maximized. Since this is a random search problem, we use the quantum-behaved particle swarm optimization method (QPSO) [Tang *et al.*, 2014] to search ϕ^*.

Once the S is determined by ϕ^* for the given network G', we can calculate the $\mathbf{S}_{i,j}$ for each node pair $\{v_i, v_j\}$ in G to construct the node similarity matrix $\mathbf{S}$.

4.2 Noise Reduction

In this section, we introduce how to construct the correction matrix $\mathbf{C}$ based on $\mathbf{S}$ to reduce the noise in $\mathbf{A}$. According to the experimental analysis in [Lu and Zhou, 2010], two nodes are more likely to be connected if they have high similarity. Inspired by this conclusion, we use the matrix $\mathbf{S}$ as a metric to evaluate the confidence level of the observed $\mathbf{A}$, and propose the construct correction matrix (CCM) algorithm to compute the matrix $\mathbf{C}$ to reduce network noise.

Algorithm 1 Construct Correction Matrix

```
Input: adjacency matrix $\mathbf{A}$, similarity matrix $\mathbf{S}$.
Output: correction matrix $\mathbf{C}$.
1: Initialize correction matrix $\mathbf{C}$;
2: if $\mathbf{A}_{i,j} > 0$ and $\mathbf{S}_{i,j} < avg(\mathbf{S}) \times \alpha$ then
3:     calculate $\mathbf{C}_{i,j}$ by Equation (2);
4: else if $\mathbf{A}_{i,j} = 0$ and $\mathbf{S}_{i,j} \geq avg(\mathbf{S}) \times (\alpha + 1)$ then
5:     calculate $\mathbf{C}_{i,j}$ by Equation (3);
6: else
7:     $\mathbf{C}_{i,j} = \mathbf{A}_{i,j}$;
8: end if
9: return $\mathbf{C}$.
```

There are two types of network noise: false links and missing links. The CCM algorithm checks the node similarity $\mathbf{S}_{i,j}$ and link weight $\mathbf{A}_{i,j}$ of each node pair $\{v_i, v_j\}$ to detect these two types of noise, and computes the $\mathbf{C}_{i,j}$ to reduce them.

In terms of false links, if $\mathbf{A}_{i,j} > 0$ and $\mathbf{S}_{i,j} < avg(\mathbf{S}) \times \alpha$, where $avg(\mathbf{S})$ is the average of all the elements in $\mathbf{S}$ and α is a threshold parameter such that $0 < \alpha < 1$, this means that there is a link between v_i and v_j in the observed network but their similarity $\mathbf{S}_{i,j}$ is very small. In this case, we regard this link as a false link. To reduce the influence of such false links, we need to reduce the original $\mathbf{A}_{i,j}$ to reflect the actual connection strength. Thus, the value of $\mathbf{C}_{i,j}$ is defined as follows:

$$\mathbf{C}_{i,j} = \mathbf{A}_{i,j} \times \frac{\mathbf{S}_{i,j}}{avg(\mathbf{S})} \times \alpha. \tag{2}$$

On the contrary, if $\mathbf{A}_{i,j} = 0$ and $\mathbf{A}_{i,j} \geq avg(\mathbf{S}) \times (\alpha+1)$, this means that there is no link between v_i and v_j in the observed network but the similarity $\mathbf{S}_{i,j}$ is large enough. In this case, we believe that there is a link between v_i and v_j in the actual network, but it has been lost in the observation process. In this case, we need to increase the $\mathbf{A}_{i,j}$ to compensate for the missing link. Therefore, the value of $\mathbf{C}_{i,j}$ is defined as follows:

$$\mathbf{C}_{i,j} = \frac{\mathbf{S}_{i,j}}{avg(\mathbf{S})} \times (\alpha + 1). \tag{3}$$

Finally, the CCM algorithm is detailed in Algorithm 1.

4.3 Node Similarity Preserving Embedding

After obtaining the correction matrix $\mathbf{C}$, we now construct the network embedding matrix $\mathbf{U}$. The $\mathbf{C}$ is derived from $\mathbf{A}$ after a noise reduction process, which is the most direct expression of network. To make the embedding matrix $\mathbf{U}$ preserve the actual node relationship in the embedding space, we introduce a non-negative basis matrix $\mathbf{M} \in \mathbb{R}^{n \times d}$ according to the non-negative decomposition [Lee and Seung, 2000]. Then we expect to make $\mathbf{M}$ times $\mathbf{U}$ to be as close as possible to $\mathbf{C}$, which leads to the following objective function:

$$\underset{\mathbf{M},\mathbf{U}}{min} \; ||\mathbf{C} - \mathbf{M}\mathbf{U}^{\mathbf{T}}||_F^2, s.t. \mathbf{M} \geq 0, \mathbf{U} \geq 0. \tag{4}$$

Because of the sparsity of real-world networks, the number of zero elements in $\mathbf{C}$ is far greater than the number of non-zero elements. This makes the learned $\mathbf{U}$ more prone to reconstruct the zero elements. Inspired by [Wang *et al.*, 2016], we add a greater penalty to the reconstruction error of the non-zero elements than to that of the zero elements. For this purpose, we add a weight matrix $\mathbf{W} \in \mathbb{R}^{n\times n}$ to Equation (4). Specifically, if $\mathbf{C}_{i,j} = 0$, $\mathbf{W}_{i,j} = 1$, else $\mathbf{W}_{i,j} = \theta > 1$. The revised objective function is as follows:

$$\underset{\mathbf{M},\mathbf{U}}{min} \ ||(\mathbf{C} - \mathbf{M}\mathbf{U}^{\mathbf{T}}) \odot \mathbf{W}||_F^2, s.t.\mathbf{M} \geq 0, \mathbf{U} \geq 0, \quad (5)$$

where $\odot$ means the Hadamard product.

To further reduce the impact of noise, we add a penalty function to the objective function (5). Since the matrix $\mathbf{S}$ accurately evaluates node similarity, it is well-suited for this purpose. We adopt the theory of Laplacian Eigenmaps [Belkin and Niyogi, 2003], which adds a penalty when similar nodes are mapped far apart in the embedding space. Let $||\mathbf{U}_{i,*} - \mathbf{U}_{j,*}||^2$ denote the distance between nodes v_i and v_j in the embedding space. The penalty function is as follows:

$$\underset{\mathbf{U}}{min} \ \sum_{i,j=1}^{n} \mathbf{S}_{i,j}||\mathbf{U}_{i,*} - \mathbf{U}_{j,*}||^2. \quad (6)$$

Based on Laplacian Eigenmaps, we can reformulate Equation (6) to the following objective function:

$$\underset{\mathbf{U}}{min} \ tr(\mathbf{U}^{\mathbf{T}}\mathbf{L}\mathbf{U}), s.t.\mathbf{U} \geq 0, \quad (7)$$

where Laplacian $\mathbf{L} = \mathbf{D} - \mathbf{S}$ and $\mathbf{D}$ is a diagonal matrix with the sum of each row of $\mathbf{S}$ on its diagonal, and $tr(\mathbf{X})$ is the trace of matrix $\mathbf{X}$.

Finally, the actual network structure can be preserved by combining Equations (5) and (7) and jointly minimizing the following objective function:

$$\begin{aligned}&\underset{\mathbf{M},\mathbf{U}}{min} \ ||(\mathbf{C} - \mathbf{M}\mathbf{U}^{\mathbf{T}}) \odot \mathbf{W}||_F^2 + \beta tr(\mathbf{U}^{\mathbf{T}}\mathbf{L}\mathbf{U}),\\ &s.t.\mathbf{M} \geq 0, \mathbf{U} \geq 0.\end{aligned} \quad (8)$$

Because the objective function (8) is not convex, it is impossible to give closed-form solutions. Inspired by [Wang *et al.*, 2017c], we propose an algorithm to find an approximate solution. Our method iteratively updates $\mathbf{U}$ and $\mathbf{M}$ until the function (8) converges. The updating steps for $\mathbf{U}$ and $\mathbf{M}$ are as follows.

Update $\mathbf{M}$: When matrix $\mathbf{U}$ is fixed during the update process of $\mathbf{M}$, Equation (8) can be revised to $\underset{\mathbf{M},\mathbf{U}}{min}||(\mathbf{C} - \mathbf{M}\mathbf{U}^{\mathbf{T}}) \odot \mathbf{W}||_F^2, s.t.\mathbf{M} \geq 0$. According to [Wang *et al.*, 2017c], the updating of $\mathbf{M}$ can be obtained as:

$$\mathbf{M} \leftarrow \mathbf{M} \odot \frac{(\mathbf{C} \odot \mathbf{W} \odot \mathbf{W})\mathbf{U}}{[(\mathbf{M}\mathbf{U}^{\mathbf{T}}) \odot \mathbf{W} \odot \mathbf{W}]\mathbf{U}}. \quad (9)$$

Update $\mathbf{U}$: When matrix $\mathbf{M}$ is fixed during the update process of $\mathbf{U}$, the Lagrange multiplier matrix $\psi = [\psi_{i,j}]$ is introduced to guarantee that $\mathbf{U}$ is nonnegative. Then Equation (8) is equivalent to:

$$\begin{aligned}\underset{\mathbf{U}}{min} \ L(\mathbf{U}) &= tr([(\mathbf{C} - \mathbf{M}\mathbf{U}^{\mathbf{T}}) \odot \mathbf{W}][(\mathbf{C} - \mathbf{M}\mathbf{U}^{\mathbf{T}}) \odot \mathbf{W}]^{\mathbf{T}})\\ &+ \beta tr(\mathbf{U}^{\mathbf{T}}\mathbf{L}\mathbf{U}) + tr(\psi\mathbf{U}^{\mathbf{T}}).\end{aligned} \quad (10)$$

Setting derivatives of $L(\mathbf{U})$, with respect to $\mathbf{U}$, to 0, we have:

$$\begin{aligned}\psi = &-2(\mathbf{C}^{\mathbf{T}} \odot \mathbf{W}^{\mathbf{T}} \odot \mathbf{W}^{\mathbf{T}})\mathbf{M} + 2\beta\mathbf{L}\mathbf{U}\\ &+ 2[(\mathbf{U}\mathbf{M}^{\mathbf{T}}) \odot \mathbf{W}^{\mathbf{T}} \odot \mathbf{W}^{\mathbf{T}}]\mathbf{M}.\end{aligned} \quad (11)$$

Following the Karush-Kuhn-Tucker (KKT) [Wang *et al.*, 2017c] condition for the nonnegativity of $\mathbf{U}$, we have the following equation:

$$\begin{aligned}&([(\mathbf{C}^{\mathbf{T}} \odot \mathbf{W}^{\mathbf{T}} \odot \mathbf{W}^{\mathbf{T}})]\mathbf{M})_{i,j}\mathbf{U}_{i,j} - \beta(\mathbf{L}\mathbf{U})_{i,j}\mathbf{U}_{i,j}\\ &- ([(\mathbf{U}\mathbf{M}^{\mathbf{T}}) \odot \mathbf{W}^{\mathbf{T}} \odot \mathbf{W}^{\mathbf{T}}]\mathbf{M})_{i,j}\mathbf{U}_{i,j} = 0.\end{aligned} \quad (12)$$

This equation leads to the following updating formula:

$$\mathbf{U} \leftarrow \mathbf{U} \odot \frac{(\mathbf{C}^{\mathbf{T}} \odot \mathbf{W}^{\mathbf{T}} \odot \mathbf{W}^{\mathbf{T}})\mathbf{M} + \beta\mathbf{S}\mathbf{U}}{[(\mathbf{U}\mathbf{M}^{\mathbf{T}}) \odot \mathbf{W}^{\mathbf{T}} \odot \mathbf{W}^{\mathbf{T}}]\mathbf{M} + \beta\mathbf{D}\mathbf{U}}. \quad (13)$$

To prove the convergence of the updating rule (13), we introduce an auxiliary function as in [Lee and Seung, 2000]. The definition of the auxiliary function is as follows.

Definition 4. A function $V(x, x')$ is an auxiliary function of function $F(x)$ where $V(x, x') \geq F(x)$ and $V(x, x) = F(x)$ for any x, x'.

Based on the auxiliary function, [Lee and Seung, 2000] proposed the following lemma.

Lemma 1. If V is an auxiliary function of F, then F is nonincreasing under the updating rule:

$$x^{(t+1)} = \underset{x}{\arg\min} \ V(x, x^{(t)}). \quad (14)$$

Let $F_{i,j}$ denote the part of objective function (10) which is only relevant to $\mathbf{U}_{i,j}$. Then we have:

$$\begin{aligned}\frac{\partial F_{i,j}}{\partial \mathbf{U}_{i,j}} = F'_{i,j} = &-2(\mathbf{C}^{\mathbf{T}} \odot \mathbf{W}^{\mathbf{T}} \odot \mathbf{W}^{\mathbf{T}})_{i,j} + 2\beta(\mathbf{L}\mathbf{U})_{i,j}\\ &+ 2(((\mathbf{U}\mathbf{M}^{\mathbf{T}}) \odot \mathbf{W}^{\mathbf{T}} \odot \mathbf{W}^{\mathbf{T}})\mathbf{M})_{i,j},\end{aligned} \quad (15)$$

$$\frac{\partial^2 F_{i,j}}{\partial \mathbf{U}_{i,j}^2} = F''_{i,j} = \sum_{k=1}^{n} \mathbf{W}_{i,k}^2\mathbf{M}_{k,j}^2 + 2\beta\mathbf{L}_{i,i}. \quad (16)$$

Now we propose a specific auxiliary function $V(x, x')$ for $F_{i,j}$ based on Lemma 2.

Lemma 2. The function

$$\begin{aligned}V(\mathbf{U}_{i,j}, \mathbf{U}_{i,j}^{(t)}) &= F_{i,j}(\mathbf{U}_{i,j}^{(t)}) + F'_{i,j}(\mathbf{U}_{i,j}^{(t)})(\mathbf{U}_{i,j} - \mathbf{U}_{i,j}^{(t)})\\ &+ \frac{([(\mathbf{U}\mathbf{M}^{\mathbf{T}}) \odot \mathbf{W}^{\mathbf{T}} \odot \mathbf{W}^{\mathbf{T}}]\mathbf{M} + \beta\mathbf{D}\mathbf{U})_{i,j}}{\mathbf{U}_{i,j}^{(t)}}(\mathbf{U}_{i,j} - \mathbf{U}_{i,j}^{(t)})^2,\end{aligned} \quad (17)$$

is an auxiliary function for $F_{i,j}$.

Proof. It is obvious that $F_{i,j}(\mathbf{U}_{i,j}) = V(\mathbf{U}_{i,j}, \mathbf{U}_{i,j})$, so we need to prove $V(\mathbf{U}_{i,j}, \mathbf{U}_{i,j}^{(t)}) \geq F_{i,j}(\mathbf{U}_{i,j})$. The Taylor series expansion of $F_{i,j}(\mathbf{U}_{i,j})$ at $\mathbf{U}_{i,j}^{(t)}$ is

$$\begin{aligned}F_{i,j}(\mathbf{U}_{i,j}) &= F_{i,j}(\mathbf{U}_{i,j}^{(t)}) + F'_{i,j}(\mathbf{U}_{i,j}^{(t)})(\mathbf{U}_{i,j} - \mathbf{U}_{i,j}^{(t)})\\ &+ (\sum_{k=1}^{n} \mathbf{W}_{i,k}^2\mathbf{M}_{k,j}^2 + \beta\mathbf{L}_{i,i})(\mathbf{U}_{i,j} - \mathbf{U}_{i,j}^{(t)})^2.\end{aligned} \quad (18)$$

Compared to Equation (17), we need to prove that

$$\frac{([(\mathbf{UM^T}) \odot \mathbf{W^T} \odot \mathbf{W^T}]\mathbf{M} + \beta\mathbf{DU})_{i,j}}{\mathbf{U}_{i,j}^{(t)}} \geq \sum_{k=1}^{n} \mathbf{W}_{i,k}^2 \mathbf{M}_{k,j}^2 + 2\beta\mathbf{L}_{i,i}. \tag{19}$$

We have

$$\begin{aligned}([(\mathbf{UM^T}) \odot \mathbf{W^T} \odot \mathbf{W^T}]\mathbf{M})_{i,j} &= \sum_{a=1}^{n} \mathbf{W}_{i,a}^2 \mathbf{M}_{a,j} \sum_{b=1}^{m} \mathbf{U}_{i,b}^{(t)} \mathbf{M}_{a,b} \\ &\geq \sum_{a=1}^{n} \mathbf{W}_{i,a}^2 \mathbf{M}_{a,j}^2 \mathbf{U}_{i,j}^{(t)},\end{aligned} \tag{20}$$

$$\begin{aligned}\beta(\mathbf{DU})_{i,j} &= \beta \sum_{k=1}^{n} \mathbf{D}_{i,k} \mathbf{U}_{i,j}^{(t)} \geq \beta \mathbf{D}_{i,i} \mathbf{U}_{i,j}^{(t)} \\ &\geq \beta(\mathbf{D} - \mathbf{S})_{i,i} \mathbf{U}_{i,j}^{(t)} = \beta \mathbf{L}_{i,i} \mathbf{U}_{i,j}^{(t)}.\end{aligned} \tag{21}$$

Therefore, Equation (19) is proved.

Based on Lemmas 1 and 2, we can show the convergence of the update rule (13).

Theorem 1. The objective function (10) is nonincreasing under the update rule of Equation (13) for $\mathbf{U}$.

Proof. According to Lemma 2, $V(\mathbf{U}_{i,j}, \mathbf{U}_{i,j}^{(t)})$ in Equation (17) is an auxiliary function of $F_{i,j}$. Replacing the $V(\mathbf{U}_{i,j}, \mathbf{U}_{i,j}^{(t)})$ in Equation (14) by Equation (17), we have the following update rule:

$$\mathbf{U}_{i,j}^{(t+1)} \leftarrow \mathbf{U}_{i,j}^{(t)} \frac{[(\mathbf{C^T} \odot \mathbf{W^T} \odot \mathbf{W^T})\mathbf{M} + \beta\mathbf{SU}]_{i.j}}{([(\mathbf{UM^T}) \odot \mathbf{W^T} \odot \mathbf{W^T}]\mathbf{M} + \beta\mathbf{DU})_{i.j}}, \tag{22}$$

which is the same as (13). Following Lemma 1, under this update rule, the objective function (10) will be nonincreasing.

Finally, the objective function (8) can be achieved by updating $\mathbf{M}$ and $\mathbf{U}$ in Equations (9) and (13), respectively. The complexity of the updating rules in (9) and (13) is $O(n^2d + nd^2)$, since $n \gg d$, so the overall complexity of the update rules of NSP is $O(n^2d)$.

5 Experimental Evaluation

In this section, we conduct experiments to validate the effectiveness of NSP on several real-world applications: node classification, node clustering, and link prediction. We evaluated the method on four social networks. 1) BlogCatalog is a network of social relationships of bloggers listed on the BlogCatalog website (10312 nodes, 333983 links, and 39 different labels). 2) Cora is a citation network of scientific publications (2708 nodes, 5278 links, and 7 different labels). 3) EmailEu is an email network of a large European research institution (1005 nodes, 25571 links, and 42 different labels). 4) Polblogs is a blog network about US politics recorded in 2005 (1490 nodes, 16715 links, and 2 different labels).

The following five network embedding algorithms are used in our experiments for comparison: DeepWalk [Perozzi *et al.*, 2014], LINE [Tang *et al.*, 2015], Node2vec [Grover and Leskovec, 2016], M-NMF [Wang *et al.*, 2017c], and GraRep [Cao *et al.*, 2015]. The parameters of the comparison algorithms are set to their default values, and the parameters of NSP are tuned by using grid search on the validation set. The S_v of NSP is consists of two local similarity indexes CN and AA, and two global similarity indexes KI and SimRank [Lu and Zhou, 2010]. We uniformly set the representation dimension $d = 128$.

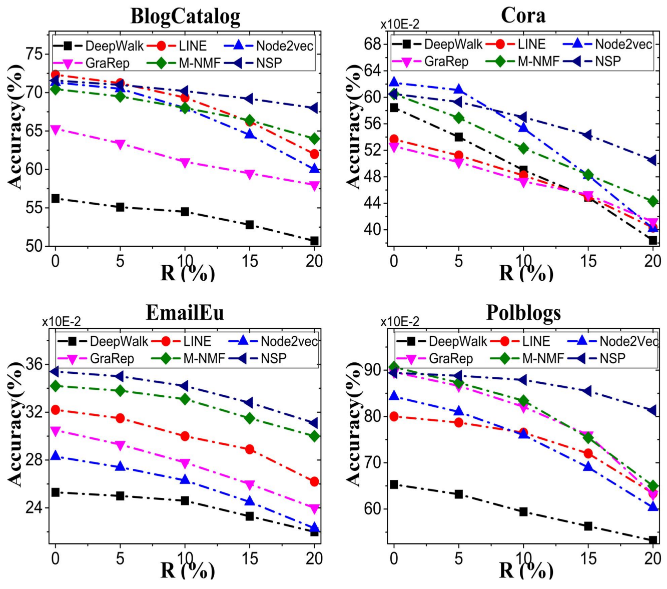

Figure 2: The accuracy of node classification with varying R.

5.1 Node Classification

This section presents the performance of all algorithms on the node classification task with varying proportions of noise in the network datasets. Let R denote the proportion of noise added. We first delete $|E|/R$ links from the experimental network, to simulate the missing links, and then we randomly add $|E|/R$ links to the network to simulate false links. For node classification, the learned representations of nodes are used to classify these nodes into a set of labels. Following [Wang *et al.*, 2017c], we used the KNN algorithm to train the classifiers. For each class of a given network, we randomly selected 80% of the nodes as the training nodes and the rest as the testing nodes. The prediction accuracy is used as the evaluation metric.

As illustrated in Fig. 2, in these four networks, the classification accuracy of all algorithms decreases with the increase of noise ratio R, especially in the Cora and Polblogs networks. This is because the network noise disturbs the network structure and leads to inferior performance in all of the algorithms. In the absence of artificial noise ($R = 0$), the accuracy of NSP is similar to the optimal results in BlogCatalog, Cora, and Polblogs, while NSP achieves the best accuracy in EmailEu. With increased network noise, $R \in [5\%, 20\%]$, the advantage of NSP becomes increasingly apparent: when $R > 5\%$, NSP outperforms other algorithms in all networks; when $R = 20\%$, the accuracy of NSP is about $5\% - 15\%$

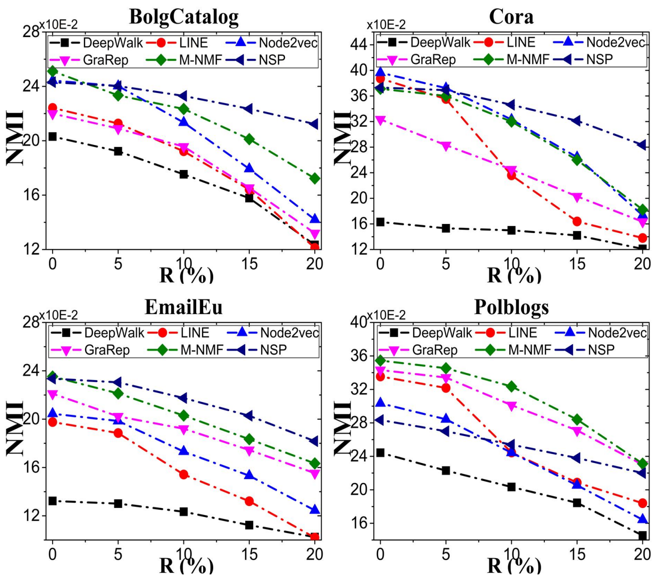

Figure 3: The NMI of node clustering with varying R.

Algorithm	P@10	P@100	P@500	P@1000	P@2000
DeepWalk	0	0	0.052	0.034	0.045
LINE	0	0.15	0.122	0.103	0.202
Node2Vec	0.2	0.05	0.056	0.088	0.132
GraRep	0	0.12	0.026	0.108	0.152
M-NMF	0	0.11	0.034	0.056	0.144
NSP	**0.2**	**0.18**	**0.136**	**0.115**	**0.154**

Table 1: $precision@k$ on BlogCatalog network for link prediction. The best performing algorithm is highlighted in bold.

Algorithm	P@10	P@100	P@500	P@1000	P@2000
DeepWalk	0.3	0.29	0.293	0.158	0.167
LINE	0.4	0.38	0.432	0.432	0.312
Node2Vec	0.2	0.16	0.116	0.105	0.182
GraRep	0.2	0.04	0.032	0.038	0.045
M-NMF	1	0.96	0.884	0.798	0.564
NSP	**1**	**1**	**0.892**	**0.808**	**0.576**

Table 2: $precision@k$ on EmailEu network for link prediction. The best performing algorithm is highlighted in bold.

in BlogCatalog, Cora, and Polblogs. This is because NSP adopts more node similarity measures; hence, it is more effective in eliminating part of the noise.

5.2 Node Clustering

In this section, we evaluate the performance of all algorithms on the node clustering task. Following [Wang *et al.*, 2017c], we adopt the K-means algorithm in this experiment to perform the node clustering and use normalized mutual information (NMI) [Goyal and Ferrara, 2018] for evaluating the clustering results. Since K-means is sensitive to the initialization of the centroids, we adopt k-means++ [Bachem *et al.*, 2016] for centroids initialization and run each clustering 10 times.

As we can see, the NMI of all algorithms decreases as the noise rate R increases in these four networks, just like the results in Section 5.1. This again suggests that the existence of noise has a significant impact on the effectiveness of embedding algorithms. By considering the node similarity, NSP has a stronger noise tolerance than the other algorithms. NSP outperforms the other algorithms when $R > 5\%$ in the BlogCatalog, Cora, and EmailEu networks. Although the NMI of NSP is not the largest in the Polblogs network, its rate of decline is the smallest. When R increases from 0% to 20%, the NMI of NSP only decreases about 0.04 in Polblogs, in contrast to M-NMF (decreases 0.12), Node2vec (0.14), and LINE (0.15). These results again show the advantages of introducing node similarity to network embedding.

5.3 Link Prediction

In this section, we concentrate on the link prediction task and experiment on the BlogCatalog and EmailEu networks. In the link prediction task, for each dataset we first randomly hide 20% of the network links and learn the embedding using the remaining 80% of the links, to predict from the learned embedding the most likely links that are not observed in the training data. The $precision@k$ [Goyal and Ferrara, 2018] is adopted as the evaluation metric of predicting the hidden links. The results of all embedding algorithms on BlogCatalog and EmailEu are presented in Tables 1 and 2.

The results in the BlogCatalog and EmailEu networks show that as k increases, the performance of our method is consistently better than other network embedding algorithms, especially in the EmailEu network. (When $k = 1000$, the precision of NSP is still greater than 0.8, while other competitors are all less than 0.8.) This is because the idea of NSP preserves the node similarity quantified by the hybrid similarity index in the network embedding process, thereby giving NSP better link prediction power.

6 Conclusion

We propose the node similarity preserving (NSP) method in this paper to perform network embedding for social networks with noise. NSP uses a comprehensive similarity index to construct the node similarity matrix $\mathbf{S}$. $\mathbf{S}$ can quantify the credibility of the observed network structure. Based on $\mathbf{S}$, we obtain a corrected matrix $\mathbf{C}$, which has reduced noise. NSP also includes a objective function that preserves the constructed node similarity $\mathbf{S}$ and the corrected network structure $\mathbf{C}$. Finally, efficient updating rules, with correctness and convergence guarantees, are also provided in NSP. Comprehensive experiments comparing NSP with state-of-the-art approaches on four real-world social networks demonstrated the substantial gains of the proposed method for node classification, node clustering, and link prediction tasks.

Acknowledgments

This work was supported in part by the National Natural Science Foundation of China (No. 61572369), in part by the Key Projects of Guangdong Natural Science Foundation (No. 2018B030311003).

References

[Bachem *et al.*, 2016] Olivier Bachem, Mario Lucic, Seyed Hamed Hassani, and Andreas Krause. Fast and provably good seedings for k-means. In *Proceedings of the NIPS 2016*, pages 55–63, 2016.

[Belkin and Niyogi, 2003] Mikhail Belkin and Partha Niyogi. Laplacian eigenmaps for dimensionality reduction and data representation. *Neural Computation*, 15(6):1373–1396, 2003.

[Cao *et al.*, 2015] Shaosheng Cao, Wei Lu, and Qiongkai Xu. Grarep: Learning graph representations with global structural information. In *Proceedings of the 24th CIKM*, pages 891–900, 2015.

[Cui *et al.*, 2017] Peng Cui, Xiao Wang, Jian Pei, and Wenwu Zhu. A survey on network embedding. *CoRR*, abs/1711.08752, 2017.

[Goyal and Ferrara, 2018] Palash Goyal and Emilio Ferrara. Graph embedding techniques, applications, and performance: A survey. *Knowl.-Based Syst.*, 151:78–94, 2018.

[Grover and Leskovec, 2016] Aditya Grover and Jure Leskovec. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD*, pages 855–864, 2016.

[Hu *et al.*, 2017] Wenbin Hu, Huan Wang, Zhenyu Qiu, Cong Nie, Liping Yan, and Bo Du. An event detection method for social networks based on hybrid link prediction and quantum swarm intelligent. *World Wide Web*, 20(4):775–795, 2017.

[Lee and Seung, 2000] Daniel D. Lee and H. Sebastian Seung. Algorithms for non-negative matrix factorization. In *Advances in Neural Information Processing Systems 13*, pages 556–562, 2000.

[Li *et al.*, 2014] Aaron Q. Li, Amr Ahmed, Sujith Ravi, and Alexander J. Smola. Reducing the sampling complexity of topic models. In *Proceedings of the 20th ACM SIGKDD*, pages 891–900, 2014.

[Liben-Nowell and Kleinberg, 2007] David Liben-Nowell and Jon M. Kleinberg. The link-prediction problem for social networks. *JASIST*, 58(7):1019–1031, 2007.

[Liu *et al.*, 2017] Weiwei Liu, Ivor W. Tsang, and Klaus-Robert Müller. An easy-to-hard learning paradigm for multiple classes and multiple labels. *Journal of Machine Learning Research*, 18:94:1–94:38, 2017.

[Lu and Zhou, 2010] Linyuan Lu and Tao Zhou. Link prediction in complex networks: A survey. *CoRR*, abs/1010.0725, 2010.

[Ou *et al.*, 2015] Mingdong Ou, Peng Cui, Fei Wang, Jun Wang, and Wenwu Zhu. Non-transitive hashing with latent similarity components. In *Proceedings of the 21st ACM SIGKDD*, pages 895–904, 2015.

[Ou *et al.*, 2016] Mingdong Ou, Peng Cui, Jian Pei, Ziwei Zhang, and Wenwu Zhu. Asymmetric transitivity preserving graph embedding. In *Proceedings of the 22nd ACM SIGKDD*, pages 1105–1114, 2016.

[Perozzi *et al.*, 2014] Bryan Perozzi, Rami Al-Rfou, and Steven Skiena. Deepwalk:online learning of social representations. In *Proceeding of the 20th ACM SIGKDD*, pages 701–710, 2014.

[Tang *et al.*, 2014] Deyu Tang, Yongming Cai, Jie Zhao, and Yun Xue. A quantum-behaved particle swarm optimization with memetic algorithm and memory for continuous non-linear large scale problems. *Inf. Sci.*, 289:162–189, 2014.

[Tang *et al.*, 2015] Jian Tang, Meng Qu, Mingzhe Wang, Ming Zhang, Jun Yan, and Qiaozhu Mei. Line: Large-scale information network embedding. In *Proceedings of the 24th WWW*, pages 1067–1077, 2015.

[Wang *et al.*, 2016] Daixin Wang, Peng Cui, and Wenwu Zhu. Structural deep network embedding. In *Proceedings of the 22nd ACM SIGKDD*, pages 1225–1234, 2016.

[Wang *et al.*, 2017a] Huan Wang, Wenbin Hu, Zhenyu Qiu, and Bo Du. Nodes' evolution diversity and link prediction in social networks. *IEEE Trans. Knowl. Data Eng.*, 29(10):2263–2274, 2017.

[Wang *et al.*, 2017b] Suhang Wang, Jiliang Tang, Charu C. Aggarwal, Yi Chang, and Huan Liu. Signed network embedding in social media. In *Proceedings of the 2017 SIAM*, pages 327–335, 2017.

[Wang *et al.*, 2017c] Xiao Wang, Peng Cui, Jing Wang, Jian Pei, Wenwu Zhu, and Shiqiang Yang. Community preserving network embedding. In *Proceedings of the 31st AAAI*, pages 203–209, 2017.

[Yang *et al.*, 2015] Cheng Yang, Zhiyuan Liu, Deli Zhao, Maosong Sun, and Edward Y. Chang. Network representation learning with rich text information. In *Proceedings of the 24th IJCAI*, pages 2111–2117, 2015.

[Yu *et al.*, 2017] Wenchao Yu, Wei Cheng, Charu C Aggarwal, Haifeng Chen, and Wei Wang. Link prediction with spatial and temporal consistency in dynamic networks. In *Proceedings of the 26th IJCAI*, pages 3343–3349, 2017.

[Zhang *et al.*, 2014] Qian-Ming Zhang, Xiao-Ke Xu, Yu-Xiao Zhu, and Tao Zhou. Measuring multiple evolution mechanisms of complex networks. *CoRR*, abs/1410.3519, 2014.

[Zhang *et al.*, 2018] Yuan Zhang, Tianshu Lyu, and Yan Zhang. COSINE: community-preserving social network embedding from information diffusion cascades. In *Proceedings of the 32nd AAAI*, pages 2620–2627, 2018.

[Zhu *et al.*, 2018] Dingyuan Zhu, Peng Cui, Ziwei Zhang, Jian Pei, and Wenwu Zhu. High-order proximity preserved embedding for dynamic networks. *IEEE Trans. Knowl. Data Eng.*, 30(11):2134–2144, 2018.