IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received June 16, 2019, accepted July 1, 2019, date of publication July 9, 2019, date of current version July 26, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2927496

Network Embedding Using Semi-Supervised Kernel Nonnegative Matrix Factorization

CHAOBO HE[1], QIONG ZHANG[2,4], YONG TANG[3], SHUANGYIN LIU[1], AND HAI LIU[3]

[1]School of Information Science and Technology, Zhongkai University of Agriculture and Engineering, Guangzhou 510225, China
[2]School of Computer Science and Information Engineering, Hubei University, Wuhan 430062, China
[3]School of Computer Science, South China Normal University, Guangzhou 510631, China
[4]Hyperhunch Technologies Ltd., Shenzhen 518000, China

Corresponding authors: Qiong Zhang (qiong.zhang.1@outlook.com) and Hai Liu (liuhai@m.scnu.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 61772211 and Grant 61871475, in part by the Humanity and Social Science Youth Foundation of Ministry of Education of China under Grant 19YJCZH049, in part by the Science and Technology Support Program of Guangdong Province of China under Grant 2017A040405057 and Grant 2016A030313441, and in part by the Science and Technology Support Program of Guangzhou City of China under Grant 201807010043 and Grant 201803020033.

ABSTRACT Network embedding, aiming to learn low-dimensional representations of nodes in networks, is very useful for many vector-based machine learning algorithms and has become a hot research topic in network analysis. Although many methods for network embedding have been proposed before, most of them are unsupervised, which ignores the role of prior information available in the network. In this paper, we propose a novel method for network embedding using semi-supervised kernel nonnegative matrix factorization (SSKNMF), which can incorporate prior information and thus to learn more useful features from the network through introducing kernel methodology. Besides, it can improve robustness against noises by using the objective function based on $L_{2,1}$ norm. Efficient iterative update rules are derived to resolve the network embedding model using the SSKNMF, and the convergence of these rules are strictly proved from the perspective of mathematics. The results from extensive experiments on several real-world networks show that our proposed algorithm is effective and has better performance than the existing representative methods.

INDEX TERMS Kernel method, network analysis, network embedding, nonnegative matrix factorization, semi-supervised learning.

I. INTRODUCTION

With the rapid growth of various network mechanisms (e.g., online social networks, WWW hyperlink networks and co-authorship networks), network analysis has attracted much attention from researchers in extensive fields, such as node classification [1], node clustering (a.k.a. community detection) [2], link prediction [3] and visualization, etc., in which the research tasks are significantly dependent on how the networks are represented.

A network is represented normally by an adjacency matrix, which is usually very sparse and suffers from overwhelming dimensionality. Thus, it cannot capture more complex and higher-order structural relationships hidden in the network, which as a result makes many tasks of network analysis costly in computation and ineffective in performance. To deal with these problems, network embedding has emerged and become a popular solution. Network embedding pursues the aim of learning low-dimensional latent representation of each node in a network, while still preserving structural and inherent properties of the network itself. After new representations of nodes have been learned, many tasks of network analysis can be effectively carried out by using conventional vector-based machine learning algorithms, such as K-means and support vector machine (SVM). The existing works have also demonstrated that effective network embedding methods can help to improve the performance of different tasks on network analysis [4]–[6].

Over the past few years, network embedding has become a significant topic in the research of network analysis, while many methods have been proposed such as DeepWalk [7], LINE (Large-scale information network embedding) [8], GraRep (Graph representation) [9], node2vec (Node to vector) [10], and so on. Generally, most of the existing methods for network embedding are unsupervised, where prior information from the nodes are not considered. However, the prior information has to be taken into consideration

The associate editor coordinating the review of this manuscript and approving it for publication was Pasquale De Meo.

usually in practice, because it has strong relationships with network representations. For example, we can assign community (a.k.a. cluster or group) labels to a portion of nodes manually, of which the ones with the same community label are kept closer to each other in new representation space. Therefore, by integrating prior information, network embedding can be expected to obtain a more informative and discriminative representation for every node. To achieve this goal, we propose a method for network embedding using semi-supervised kernel nonnegative matrix factorization (SSKNMF), and the work is summarized as follows:

- A network embedding model based on SSKNMF is proposed, which is capable of integrating available prior information in the network. Meanwhile, we use $L_{2,1}$ norm to devise the objective function for the model to improve the robustness concerning noises.
- We derive efficient iterative update rules as the optimal solution to the network embedding model and provide strict proof for the convergence.
- We evaluate our method on four real-world network datasets under different network analysis tasks, including node clustering and visualization. The results demonstrates its effectiveness and superiority.

The rest of this paper is organized as follows. A brief review of the related work on KNMF and network embedding are given in Section II. In Section III, we present our method for network embedding in detail, including the model, solution and convergence proof. Experimental results are reported in Section IV. Finally, conclusions are drawn and future work is discussed in Section V.

II. RELATED WORK

A. KERNEL NMF

Nonnegative matrix factorization (NMF) [11] is a popular low-rank matrix decomposition model that focuses on the analysis of data matrices whose elements are nonnegative. Given a data matrix $\mathbf{X} = [\mathbf{x}_1, \ldots, \mathbf{x}_n] \in \mathbb{R}_+^{m\times n}$ composed of n columns with desired dimension $d \ll \min(m, n)$, where m is the dimension of data and n is the number of samples, NMF aims to find two nonnegative matrices $\mathbf{W} = [w_{il}]^{m\times d} \in \mathbb{R}_+^{m\times d}$ and $\mathbf{H} = [h_{lj}]^{d\times n} \in \mathbb{R}_+^{d\times n}$, whose product can approximate to the original matrix $\mathbf{X}$: $\mathbf{X} \approx \mathbf{WH}$. Here, $\mathbf{W}$ and $\mathbf{H}$ are the basis matrix and the coefficient matrix, respectively. By imposing nonnegativity constraints on $\mathbf{W}$ and $\mathbf{H}$, each data sample $\mathbf{x}_i$ can be represented as an additive linear combination of the nonnegative basis vectors. Namely, we have $\mathbf{x}_j \approx \sum_{l=1}^{d} \mathbf{w}_l h_{lj}$, where $\mathbf{w}_l$ is the l-th column vector of $\mathbf{W}$. NMF provides more interpretable parts based decompositions, because it naturally conforms to intuitive human cognition of ''combining parts to form a whole''. This feature makes NMF widely used in various data representation tasks, including image representation [12], [13], microbiome data representation [14] and attribute representation [15].

NMF is essentially a linear model, thus it cannot represent more useful features hidden in the data, especially the nonlinear features. To overcome this limitation, Zhang *et al.* [16] proposed the kernel NMF (KNMF), which is a combination of NMF and the kernel method. As a first step, it maps each data sample into a higher or infinite dimensional feature space via a nonlinear mapping function $\phi(\cdot)$, so that a new data matrix $\phi(\mathbf{X}) = [\phi(\mathbf{x}_1), \ldots, \phi(\mathbf{x}_n)]$ can be obtained. Then, two nonnegative matrices $\mathbf{W}_\phi$ and $\mathbf{H}$ satisfying $\phi(\mathbf{X}) \approx \mathbf{W}_\phi\mathbf{H}$ are attempted to be acquired. Although direct factorization of $\phi(\mathbf{X})$ is impractical due to the complicacy of $\phi(\cdot)$, KNMF can solve this problem smartly by introducing the kernel matrix as $\mathbf{K} = \phi(\mathbf{X})^T\phi(\mathbf{X}) = [k_{ij}]^{n\times n}$, where $k_{ij} = \phi(\mathbf{x}_i)^T\phi(\mathbf{x}_j)$ can be computed using a given kernel function, such as Linear kernel, Polynomial kernel, Gaussian kernel and Sigmoid kernel mentioned in [17]. This solution is also called kernel trick. Compared with NMF, KNMF can extract more useful features hidden in the data, which has been validated by many existing works, including spectral EEG feature extraction [18], face recognition [19], and hyperspectral image reconstruction [20].

B. NETWORK EMBEDDING

Recently, many methods on network embedding have been proposed to learn compact and low-dimensional vector representations of nodes, and the existing methods can be divided into three categories in general: random walk based methods, matrix factorization based methods and deep learning based methods.

The random walk based methods can preserve higher-order proximity between two nodes globally, where the network is transformed into a collection of node sequences at first by using random walk, then structural relationships between nodes are captured from these node-context pairs. DeepWalk [7], node2vec [10] and DDRW (Discriminative deep random walk) [21] are representatives of random walk based methods.

Matrix factorization based methods represent the connections between nodes in the form of matrix and factorize this matrix to obtain the embedding, which varies accordingly on the basis of matrix properties. If the obtained matrix is positive semi-definite, one can execute eigenvalue decomposition, such as GraRep [9] and LLE (Locally linear embedding) [22]. For unstructured matrices, one can devise alternative optimization methods to obtain the embedding, such as M-NMF (Modularized nonnegative matrix factorization) [23] and DMF (Discriminative matrix factorization) [24].

Deep learning based methods mainly use varieties of neural network models to learn feature representations of networks. Representative methods include DNGR (Deep neural networks for graph representations) [25], SDNE (Structural deep network embedding) [26] and DDNE (Deep dynamic network embedding) [27]. All of them can learn complex and highly nonlinear node representations from networks.

On the other hand, from the perspective that whether the prior information attached to nodes is considered, the network embedding methods also can be classified into two groups: unsupervised network embedding and

semi-supervised embedding. At present, most of the existing methods fall into the unsupervised category, in which the node representation can be learned without incorporating any available prior information, such as DeepWalk [7], GraRep [9] and SDNE [26] as mentioned above. Compared with the unsupervised methods, only a few works focus on the semi-supervised methods, and the representatives include DMF [24], MMDW (Max-margin deepwalk) [28] and LANE (Label informed attributed network embedding) [29], all of which are only based on matrix factorization other than random walk or deep learning. These semi-supervised methods are more flexible to incorporate prior information by adding constraint terms, and complex learning framework with too many parameters to be tuned does not need to be designed at all. In fact, they also perform better than some mainstream semi-supervised network embedding methods based on random walk or deep learning, such as DDRW (Discriminative deep random walk) [21], Planetoid (Predicting labels and neighbors with embeddings transductively or inductively from Data) [30] and TriDNR (Tri-party deep network representation) [31]. In short, matrix factorization is a simple and effective solution to be implemented in semi-supervised network embedding.

In this paper, our proposed method is also built on matrix factorization. However, it has some differences from DMF, MMDW and LANE. Firstly, we impose nonnegative constraint on feature representation, which makes the experimental results more intuitive and interpretable. Secondly, we introduce the kernel method, which can help to learn more useful features, especially the nonlinear features. Finally, we use $L_{2,1}$ norm to devise the objective function, which makes the embedding model more robust against noises.

III. METHODOLOGY

A. PROBLEM STATEMENT

Without loss of generality, a given network can be formally represented as a directed and unweighted graph as $G = (V, E)$, where $V = \{v_1, \ldots, v_n\}$ is the set of n nodes, and $E = \{e_{ij} | v_i \in V \wedge v_j \in V\}$ denotes the set of all the edges between two nodes. Typically, we can use a simple adjacent matrix $\mathbf{A} = [a_{ij}]^{n\times n}$ to represent the topological features of G. If $e_{ij} \in E$, then $a_{ij} = 1$, else $a_{ij} = 0$. Assuming that the given network has prior information that some nodes are assigned with community labels, we can construct a corresponding constrained symmetric matrix $\mathbf{M} = [m_{ij}]^{n\times n}$ as:

$$m_{ij} = \begin{cases} 1 & \text{if } v_i \text{ and } v_j \text{ have the same community label} \\ 0 & \text{otherwise.} \end{cases} \quad (1)$$

With the aforementioned notations and definitions, the problem of network embedding here can be formally stated as follows:

Network embedding. Given a network G with matrices $\mathbf{A}$ and $\mathbf{M}$, we aim to learn a low-dimensional vector representation for each node as

$$f(\mathbf{A}, \mathbf{M}) : v_i \rightarrow \mathbf{h}_i \in \mathbb{R}_+^d, \quad (2)$$

where $f(\cdot)$ maps each node v_i to a d-dimensional vector $\mathbf{h}_i$ with $d \ll n$. This new representation form should be consistent with both the structure of the network and the prior information of the nodes. Namely, if nodes v_i and v_j are similar to each other in terms of their topographical features, or if their corresponding element m_{ij} in $\mathbf{M}$ is equal to 1, they should have similar representations.

B. NETWORK EMBEDDING MODEL

The adjacent matrix $\mathbf{A}$ only presents first-order proximity information between two nodes and cannot present global structure information about the network, thereby we do not factorize $\mathbf{A}$ directly to obtain the low-dimensional representations of the nodes. Motivated by the work about equivalence of DeepWalk and matrix factorization described in [32], we choose to factor the following matrix instead of the adjacent matrix $\mathbf{A}$:

$$\mathbf{X} = \frac{\mathbf{A} + \mathbf{A}^2 + \ldots + \mathbf{A}^p}{p}, \quad (3)$$

where $\mathbf{A}^p$ denotes the p-th order proximity information between nodes. By combing multiple high-order proximities, $\mathbf{X}$ can be expected to capture more global structure features. In [32], p is recommended to be set as 2 to make a trade-off between speed and accuracy.

Assuming that the expected dimension of representation is d, satisfying $d \ll n$, we can learn new representations from $\mathbf{X}$ by using NMF to decompose $\mathbf{X}$ approximately into the product of $\mathbf{Y}$ and $\mathbf{H}$, written as $\mathbf{X} \approx \mathbf{YH}$, where $\mathbf{Y} = [\mathbf{y}_1, \ldots, \mathbf{y}_d] \in \mathbb{R}_+^{n\times d}$ and $\mathbf{H} = [\mathbf{h}_1, \ldots, \mathbf{h}_n] \in \mathbb{R}_+^{d\times n}$. Each column of $\mathbf{H}$ can be regarded as low-dimensional representation of the corresponding node. Namely, $\mathbf{h}_i$ is the new vector representation of node v_i in d-dimensional space. It is intuitive that, if any two nodes v_i and v_j have the same community label (i.e., $m_{ij} = 1$), they should be spatially close to each other in high-dimensional space. To keep their consistence in intrinsic geometric structure in both high and low dimensional space, their corresponding new representations $\mathbf{h}_i$ and $\mathbf{h}_j$ should also get spatially close. Formally, the constraints from these prior information can be denoted as:

$$\begin{aligned} R(\mathbf{X}) &= \sum_{i=1}^{n}\sum_{j=1}^{n} ||\mathbf{h}_i - \mathbf{h}_j||^2 m_{ij} \\ &= 2\sum_{i=1}^{n} \mathbf{h}_i^T \mathbf{h}_i d_{ii} - 2\sum_{i,j}^{n} \mathbf{h}_i^T \mathbf{h}_j m_{ij} \\ &= 2tr(\mathbf{HDH}^T) - 2tr(\mathbf{HMH}^T) \\ &= 2tr(\mathbf{HLH}^T), \end{aligned} \quad (4)$$

where $tr(\cdot)$ denotes the trace of a matrix, $\mathbf{D} = [d_{ii}]^{n\times n}$ is a diagonal matrix ($d_{ii} = \sum_{j=1}^{n} m_{ij}$ and $d_{ij} = 0$ if $i \neq j$), and $\mathbf{L} = \mathbf{D} - \mathbf{M}$ is the Laplacian matrix of $\mathbf{M}$. By employing $R(\mathbf{X})$ as the regularization term, we can naturally incorporate prior information into the NMF-based model to guide the learning process of network embedding. If we select the widely

used Frobenius norm to construct the objective function, this semi-supervised network embedding model based on NMF can be represented as:

$$\begin{aligned}\min_{\mathbf{Y}\geq 0,\mathbf{H}\geq 0} J(\mathbf{Y},\mathbf{H}) &= ||\mathbf{X}-\mathbf{YH}||_F^2+\frac{\lambda}{2}R(\mathbf{X})\\ &= ||\mathbf{X}-\mathbf{YH}||_F^2+\lambda tr(\mathbf{HLH}^T)\\ &= \sum_{i=1}^{n}||\mathbf{x}_i-\mathbf{Yh}_i||^2+\lambda tr(\mathbf{HLH}^T),\end{aligned} \tag{5}$$

where, λ controls the contribution of prior information, which will be analyzed in the experiments. From Eq. (5), we can see that noises with large errors in $\mathbf{X}$ are prone to dominate the objective function in the form of squared errors, which means that this network embedding model based on NMF with Frobenius norm is not robust enough regarding noises. Besides, by reason that KNMF has better feature learning capability than NMF, which has been introduced in Section 2.1, it will be better to use KNMF to construct the network embedding model. Based on the analysis, we are inspired to use KNMF and $L_{2,1}$ norm to construct the following semi-supervised network embedding model:

$$\begin{aligned}\min_{\mathbf{Y}_\phi\geq 0,\mathbf{H}\geq 0} J(\mathbf{Y}_\phi,\mathbf{H}) &= ||\phi(\mathbf{X})-\mathbf{Y}_\phi\mathbf{H}||_{2,1}+\lambda tr(\mathbf{HLH}^T)\\ &= \sum_{i=1}^{n}\sqrt{\sum_{j=1}^{n}(\phi(\mathbf{X})-\mathbf{Y}_\phi\mathbf{H})_{ji}^2}\\ &\quad+\lambda tr(\mathbf{HLH}^T)\\ &= \sum_{i=1}^{n}||\phi(\mathbf{x}_i)-\mathbf{Y}_\phi\mathbf{h}_i||\\ &\quad+\lambda tr(\mathbf{HLH}^T),\end{aligned} \tag{6}$$

where $\phi(\cdot)$ is the nonlinear mapping function and $\mathbf{Y}_\phi$ is the basic matrix in new feature space. Obviously, this objective function using $L_{2,1}$ norm does not square the errors, therefore it can be used to reduce the impact of noises and improve the robustness of the network embedding model.

C. THE OPTIMIZATION SOLUTION

We employ convex NMF mentioned in [33] to solve the model described as Eq. (6). Firstly, taking into account the interpretability, we impose the constraint that the vectors of $\mathbf{Y}_\phi$ lie within the column space of $\phi(\mathbf{X})$, which can be obtained by expressing column vector $\mathbf{y}_l \in \mathbf{Y}_\phi$ as convex combination of input data vectors:

$$\mathbf{y}_l = w_{1l}\phi(\mathbf{x}_1)+\ldots+w_{nl}\phi(\mathbf{x}_n), \tag{7}$$

This can be written as in the matrix form: $\mathbf{Y}_\phi = \phi(\mathbf{X})\mathbf{W}$, where $\mathbf{W} = [w_{il}]^{n\times d}$. Replacing $\mathbf{Y}_\phi$ with $\phi(\mathbf{X})\mathbf{W}$, we can rewrite Eq. (6) to be:

$$\begin{aligned}\min_{\mathbf{W}\geq 0,\mathbf{H}\geq 0} J(\mathbf{W},\mathbf{H}) &= ||\phi(\mathbf{X})-\phi(\mathbf{X})\mathbf{WH}||_{2,1}\\ &\quad+\lambda tr(\mathbf{HLH}^T).\end{aligned} \tag{8}$$

The minimization of Eq. (8) is a typical optimization problem, which can be solved to obtain $\mathbf{W}$ and $\mathbf{H}$ by using iterative update rules. To attain the optimization, we need to transform the objective function $J(\mathbf{W},\mathbf{H})$ through the following steps at first:

$$\begin{aligned}&J(\mathbf{W},\mathbf{H})\\ &= \sum_{i=1}^{n}||\phi(\mathbf{x}_i)-\phi(\mathbf{X})\mathbf{Wh}_i||^2\frac{1}{||\phi(\mathbf{x}_i)-\phi(\mathbf{X})\mathbf{Wh}_i||}\\ &\quad+\lambda tr(\mathbf{HLH}^T)\\ &= \sum_{i=1}^{n}((\phi(\mathbf{X})-\phi(\mathbf{X})\mathbf{WH})^T(\phi(\mathbf{X})-\phi(\mathbf{X})\mathbf{WH}))_{ii}s_{ii}\\ &\quad+\lambda tr(\mathbf{HLH}^T)\\ &= \sum_{i=1}^{n}\sum_{j=1}^{n}((\phi(\mathbf{X})-\phi(\mathbf{X})\mathbf{WH})^T(\phi(\mathbf{X})-\phi(\mathbf{X})\mathbf{WH}))_{ij}s_{ji}\\ &\quad+\lambda tr(\mathbf{HLH}^T)\\ &= tr((\phi(\mathbf{X})-\phi(\mathbf{X})\mathbf{WH})\mathbf{S}(\phi(\mathbf{X})-\phi(\mathbf{X})\mathbf{WH})^T)\\ &\quad+\lambda tr(\mathbf{HLH}^T)\\ &= tr((\mathbf{I}-\mathbf{WH})^T\phi(\mathbf{X})^T\phi(\mathbf{X})(\mathbf{I}-\mathbf{WH})\mathbf{S})+\lambda tr(\mathbf{HLH}^T)\\ &= tr((\mathbf{I}-\mathbf{WH})^T\mathbf{K}(\mathbf{I}-\mathbf{WH})\mathbf{S})\\ &\quad+\lambda tr(\mathbf{HLH}^T),\end{aligned} \tag{9}$$

where, $\mathbf{K} = [k_{ij}]^{n\times n}$ is the kernel matrix and its element k_{ij} is only dependent on the inner product form $\phi(\mathbf{x}_i)^T\phi(\mathbf{x}_j)$, $\mathbf{I}$ is an $n\times n$ identity matrix and $\mathbf{S} = [s_{ii}]^{n\times n}$ is an $n\times n$ diagonal matrix defined as:

$$s_{ii} = \frac{1}{||\phi(\mathbf{x}_i)-\phi(\mathbf{X})\mathbf{Wh}_i||}. \tag{10}$$

Besides, it is noted that

$$\begin{aligned}||\phi(\mathbf{x}_i)-\phi(\mathbf{X})\mathbf{Wh}_i|| &= \sqrt{\sum_{j=1}^{n}(\phi(\mathbf{X})-\phi(\mathbf{X})\mathbf{WH})_{ji}^2}\\ &= \sqrt{((\mathbf{I}-\mathbf{WH})^T\phi(\mathbf{X})^T\phi(\mathbf{X})(\mathbf{I}-\mathbf{WH}))_{ii}}\\ &= \sqrt{((\mathbf{I}-\mathbf{WH})^T\mathbf{K}(\mathbf{I}-\mathbf{WH}))_{ii}},\end{aligned} \tag{11}$$

which is also only dependent on the kernel matrix $\mathbf{K}$. For $\forall w_{il} \in \mathbf{W}$ $(i = 1\ldots n, l = 1\ldots d)$ and $\forall h_{lj} \in \mathbf{H}$ $(l = 1\ldots d, j = 1\ldots n)$, we have $w_{il}\geq 0$ and $h_{lj}\geq 0$. Therefore, the minimization of $J(\mathbf{W},\mathbf{H})$ can be solved by using the Lagrange multiplier method. Letting $\Phi = [\Phi_{il}]^{n\times d}$ and $\Omega = [\Omega_{lj}]^{d\times n}$ respectively denote the Lagrange multipliers for $\mathbf{W}$ and $\mathbf{H}$, the corresponding Lagrange function of $J(\mathbf{W},\mathbf{H})$ is:

$$F(\mathbf{W},\mathbf{H}) = J(\mathbf{W},\mathbf{H})+tr(\Phi\mathbf{W}^T)+tr(\Omega\mathbf{H}^T). \tag{12}$$

The derivatives of $F(\mathbf{W},\mathbf{H})$ with respect to $\mathbf{W}$ and $\mathbf{H}$ are:

$$\frac{\partial F(\mathbf{W},\mathbf{H})}{\partial\mathbf{W}} = -2\mathbf{KSH}^T+2\mathbf{KWHSH}^T+\Phi, \tag{13}$$

$$\begin{aligned}\frac{\partial F(\mathbf{W},\mathbf{H})}{\partial\mathbf{H}} &= -2\mathbf{W}^T\mathbf{KXS}+2\mathbf{W}^T\mathbf{KWHS}\\ &\quad+2\lambda\mathbf{HL}+\Omega.\end{aligned} \tag{14}$$

Using the Karush-Kuhn-Tucker (KKT) conditions $\Phi_{il}w_{il} = 0$ and $\Omega_{lj}h_{lj} = 0$, we can obtain the following equations for w_{il} and h_{lj}:

$$(-2\mathbf{KSH}^T + 2\mathbf{KWHSH}^T)_{il}w_{il} + \Phi_{il}w_{il} = 0, \quad (15)$$

$$(-2\mathbf{W}^T\mathbf{KS} + 2\mathbf{W}^T\mathbf{KWHS} + 2\lambda\mathbf{HL})_{lj}h_{lj} + \Omega_{lj}h_{lj} = 0. \quad (16)$$

These equations can lead to the following iterative update rules:

$$w_{il} = w_{il}\frac{(\mathbf{KSH}^T)_{il}}{(\mathbf{KWHSH}^T)_{il}}, \quad (17)$$

$$h_{lj} = h_{lj}\frac{(\mathbf{W}^T\mathbf{KS})_{lj}}{(\mathbf{W}^T\mathbf{KWHS} + \lambda\mathbf{HL})_{lj}}. \quad (18)$$

$J(\mathbf{W}, \mathbf{H})$ is non-increasing and can converge to obtain the minimum under the above iterative updating rules. The proof will be presented in the next section.

D. CONVERGENCE PROOF

In this section, we will prove the convergence of iterative updating rules described in the following Theorems.

Theorem 1: Updating $\mathbf{W}$ using the rule of Eq. (17) while fixing $\mathbf{H}$, the objective function of Eq. (8) monotonically decreases, which can be written as

$$||\phi(\mathbf{X}) - \phi(\mathbf{X})\mathbf{W}^{t+1}\mathbf{H}||_{2,1} \leq ||\phi(\mathbf{X}) - \phi(\mathbf{X})\mathbf{W}^{t}\mathbf{H}||_{2,1}, \quad (19)$$

where t means the number of iterations.

Theorem 2: Updating $\mathbf{H}$ using the rule of Eq. (18) while fixing $\mathbf{W}$, the objective function of Eq. (8) monotonically decreases, which can be written as

$$||\phi(\mathbf{X}) - \phi(\mathbf{X})\mathbf{W}\mathbf{H}^{t+1}||_{2,1} + \lambda tr(\mathbf{H}^{t+1}\mathbf{L}(\mathbf{H}^{t+1})^T) \leq ||\phi(\mathbf{X}) - \phi(\mathbf{X})\mathbf{W}\mathbf{H}^{t}||_{2,1} + \lambda tr(\mathbf{H}^{t}\mathbf{L}(\mathbf{H}^{t})^T). \quad (20)$$

The proof of convergence for $\mathbf{W}$ is identical to that for $\mathbf{H}$, thus we only focus on $\mathbf{H}$ here (i.e., the proof of Theorem 2). To prove Theorem 2, we need to employ the following two Lemmas.

Lemma 1: Given nonnegative matrices $\mathbf{C}$, $\mathbf{N}$ and $\mathbf{B}$, where $\mathbf{C} = \mathbf{C}^T$ and $\mathbf{N} = \mathbf{N}^T$, we have

$$tr(\mathbf{B}^T\mathbf{CBN}) \leq \sum_{i,j}(\mathbf{CB}'\mathbf{N})_{ij}\frac{\mathbf{B}_{ij}^2}{\mathbf{B}'_{ij}}, \quad (21)$$

and the equality holds when $\mathbf{B} = \mathbf{B}'$. The detailed proof of this Lemma can be found in [33].

Lemma 2: Under the updating rule of Eq. (18), the following inequation holds

$$tr((\mathbf{I} - \mathbf{WH}^{t+1})^T\mathbf{K}(\mathbf{I} - \mathbf{WH}^{t+1})\mathbf{S}) + \lambda tr(\mathbf{H}^{t+1}\mathbf{L}(\mathbf{H}^{t+1})^T) \leq tr((\mathbf{I} - \mathbf{WH}^{t})^T\mathbf{K}(\mathbf{I} - \mathbf{WH}^{t})\mathbf{S}) + \lambda tr(\mathbf{H}^{t}\mathbf{L}(\mathbf{H}^{t})^T), \quad (22)$$

where the diagonal element s_{ii} in $\mathbf{S}$ is defined as $s_{ii} = 1/||\phi(\mathbf{x}_i) - \phi(\mathbf{X})\mathbf{W}\mathbf{h}_i^t||$.

Proof: Lemma 2 is proved by using the auxiliary function approach proposed in [34]. First of all, we define

$$G(\mathbf{H}) = tr((\mathbf{I} - \mathbf{WH})^T\mathbf{K}(\mathbf{I} - \mathbf{WH})\mathbf{S}) + \lambda tr(\mathbf{HLH}^T). \quad (23)$$

Then, Eq. (22) can be reformulated as

$$G(\mathbf{H}^{t+1}) \leq G(\mathbf{H}^t). \quad (24)$$

According to Lemma 1, we can obtain

$$\begin{aligned} G(\mathbf{H}) &= tr((\mathbf{I} - \mathbf{WH})^T\mathbf{K}(\mathbf{I} - \mathbf{WH})\mathbf{S}) \\ &\quad + \lambda tr(\mathbf{HLH}^T) \\ &= tr(\mathbf{KS} - 2\mathbf{H}^T\mathbf{W}^T\mathbf{KS}) + tr(\mathbf{H}^T\mathbf{W}^T\mathbf{KWHS}) \\ &\quad + \lambda tr(\mathbf{HLH}^T) \\ &\leq tr(\mathbf{KS} - 2\mathbf{H}^T\mathbf{W}^T\mathbf{KS}) + \sum_{l,j}(\mathbf{W}^T\mathbf{KWH}'\mathbf{S})_{lj}\frac{h_{lj}^2}{h'_{lj}} \\ &\quad + \lambda\sum_{l,j}(\mathbf{H}'\mathbf{L})_{lj}\frac{h_{lj}^2}{h'_{lj}} \\ &= Z(\mathbf{H}, \mathbf{H}'). \end{aligned} \quad (25)$$

The equality holds when $\mathbf{H} = \mathbf{H}'$, hence $Z(\mathbf{H}, \mathbf{H}')$ is an auxiliary function of $G(\mathbf{H})$. The 1st order and 2nd order derivatives of $Z(\mathbf{H}, \mathbf{H}')$ with regard to $\mathbf{H}$ are respectively expressed as follows:

$$\frac{\partial Z(\mathbf{H}, \mathbf{H}')}{\partial h_{lj}} = 2(-\mathbf{W}^T\mathbf{KS})_{lj} + \frac{2(\mathbf{W}^T\mathbf{KWH}'\mathbf{S} + \lambda\mathbf{H}'\mathbf{L})_{lj}h_{lj}}{h'_{lj}}, \quad (26)$$

$$\frac{\partial Z^2(\mathbf{H}, \mathbf{H}')}{\partial h_{lj}\partial h_{lj}} = \frac{2(\mathbf{W}^T\mathbf{KWH}'\mathbf{S} + \lambda\mathbf{H}'\mathbf{L})_{lj}}{h'_{lj}}. \quad (27)$$

Due to the nonnegative property of each matrix involved, we have $\frac{\partial Z^2(\mathbf{H}, \mathbf{H}')}{\partial h_{lj}\partial h_{lj}} \geq 0$, which makes the Hessian matrix of $Z(\mathbf{H}, \mathbf{H}')$ be positive semi-definite. This indicates that $Z(\mathbf{H}, \mathbf{H}')$ is a convex function and has a global minimum, which can be obtained by setting the gradient of $Z(\mathbf{H}, \mathbf{H}')$ to zero and solving it for $\mathbf{H}$. Thus, we set Eq. (26) to zero and can obtain

$$h_{lj} = h'_{lj}\frac{(\mathbf{W}^T\mathbf{KS})_{lj}}{(\mathbf{W}^T\mathbf{KWH}'\mathbf{S} + \lambda\mathbf{H}'\mathbf{L})_{lj}}. \quad (28)$$

Eq. (28) will be as same as Eq. (18) if letting $\mathbf{H}^{t+1} = \mathbf{H}$ and $\mathbf{H}^t = \mathbf{H}'$. Under this updating rule, $G(\mathbf{H})$ decreases monotonically. Namely, we have $G(\mathbf{H}^{t+1}) \leq G(\mathbf{H}^t)$. Therefore, Lemma 2 holds.

From Eq. (9), we can deduce that the left-hand side of Eq. (20) is actually equal to $G(\mathbf{H}^{t+1})$ and the right-hand side of Eq. (19) is actually equal to $G(\mathbf{H}^t)$. According to Lemma 2, $G(\mathbf{H}^{t+1}) \leq G(\mathbf{H}^t)$, Eq. (20) holds. Hence, Theorem 2 holds.

IV. EXPERIMENTS

In this section, we introduce the datasets for experiments and the baseline methods used for comparisons at first. Afterwards, we evaluate our proposed method (called SSKNMF for short hereafter) on two network analysis tasks: node clustering and visualization. Finally, we test the robustness and the sensitivity of SSKNMF parameters.

A. DATASETS

We select four real-world network datasets to evaluate our method SSKNMF. Their details are introduced as follows and their statistics are listed in Table 1.

TABLE 1. Statistics of selected datasets.

Dataset	$\|V\|$	$\|E\|$	$\#Communities$
Polblog	1224	19025	2
Email	1005	25571	42
DBLP	4985	24954	201
YouTube	11297	841340	137

- Political blog network (Polblog) [35]. Polblog is a directed hyperlink network of websites, which is composed of blogs about US politics and the hyperlinks related to them. This network has 1224 nodes (blogs) and 19025 links, and the nodes are divided into 2 communities according to their political labels (liberal and conservative).
- Email-Eu-core network (Email) [36]. Email is a directed communication network generated by email data from a large European research institution. It has 1005 nodes (institution members) and 25571 links (email communication relationships from the members), and the nodes are divided into 42 communities according to their department labels.
- DBLP [37]. DBLP is an undirected co-authorship network extracted from DBLP computer science bibliography. It has 4985 nodes (authors) and 24954 links (co-authorships from authors), and the nodes are divided into 201 communities according to their research interest labels.
- YouTube [37]. YouTube is an undirected social network extracted from YouTube video-sharing website. It contains 11297 nodes (users) and 841340 links (friendships between users), and the nodes are divided into 137 communities according to their interest labels.

Note that both the DBLP and YouTube datasets here are much smaller than the original ones in [37], because the nodes are only selected from some non-overlapped communities for the convenience of subsequent performance comparisons of node hard clustering. Besides, although these two datasets are undirected networks, we can treat them as directed networks with bidirections along the connection edges, which should be applicable to our SSKNMF model.

All of these selected datasets have ground-truth results of community partition, which can provide reliable sources for setting the known prior information. Assuming that the ground-truth set of communities is $C = \{c_1, \ldots, c_r\}$, the possible number (ML_{pairs}) of node pairs having the same community label can be expressed as:

$$ML_{pairs} = \sum_{i=1}^{r} \frac{|c_i|(|c_i|-1)}{2}. \tag{29}$$

In our experiments, we randomly extract these labeled node pairs as prior information according to the pre-assigned ratio, and then construct the pairwisely constrained symmetric matrix $\mathbf{M}$ correspondingly.

B. BASELINE METHODS

In order to demonstrate the advantages of our method, three types of unsupervised methods and four types of semi-supervised methods are selected to be compared in the experiments. These methods are simply introduced as follows:

- DeepWalk [7]. DeepWalk is a random walk based unsupervised method for network embedding. Its parameters are set as: $\gamma = 80$ (walks per vertex) and $\omega = 10$ (window size) on Polblog and Email datasets, $\gamma = 100$ and $\omega = 12$ on DBLP and YouTube datasets.
- GraRep [9]. GraRep is a matrix factorization based unsupervised method. It uses k-steps probability transition matrix of the network to learn node representations. In this paper, we set $k = 3$ on Polblog and Email datasets, $k = 6$ on DBLP and YouTube datasets.
- DNGR [25]. DNGR is a deep learning based unsupervised method. It applies the stacked denoising autoencoders (SDAE) to learn node representations from the high-dimensional positive pointwise mutual information matrix of the network. We set the number of layers of the neural networks for all the datasets to be 4, and besides, all the neurons are activated by the sigmoid function.
- MMDW [28]. MMDW is also a matrix factorization based semi-supervised method. It provides a unified network embedding learning framework that jointly optimizes the max-margin classifier and the aimed embedding learning model. Therefore, it needs a certain proportion of labeled nodes for training. MMDW has a hyper-parameter η to balance the biased gradient and the original gradient of the joint objective function. In this paper, we set $\eta = 10^{-2}$ for Polblog and Email datasets, $\eta = 10^{-3}$ for DBLP and YouTube datasets.
- TriDNR [31]. TriDNR is a deep learning based semi-supervised method. It applies a coupled deep natural language module to learn node representations, which can enhance the performance by integrating information from three parties: node structure, node content (if available) and node labels (if available). The parameters are set as follows: $\omega = 10$ (window size) and $\alpha = 0.6$ (weight parameter) on Polblog and Email datasets, $\omega = 12$ and $\alpha = 0.8$ on DBLP and YouTube datasets.
- ANRL [38]. ANRL (Attributed network representation learning) is also a deep learning based semi-supervised

method. It designs an attribute-aware skip-gram model to learn node representations, which can also integrate node label information into node structures. The settings of parameters γ (walks per vertex) and ω (window size) are as same as those of DeepWalk.

- SSNMF. SSNMF is actually a short version of SSKNMF without using kernel method. Its corresponding network embedding model is:

$$\min_{\mathbf{Y}\geq 0,\mathbf{H}\geq 0} J(\mathbf{Y},\mathbf{H}) = ||\mathbf{X}-\mathbf{YH}||_{2,1} + \lambda tr(\mathbf{HLH}^T). \quad (30)$$

By comparing SSKNMF with SSNMF, we can validate the effectiveness of kernel method used for network embedding. In our experiments, we set the λ parameter both in SSKNMF and SSNMF to be 1 according to the analysis results of parameter sensitivity described in Section 4.6. By the way, the kernel function we use in SSKNMF is Gaussian kernel.

All of the baselines mentioned above are implemented according to the programs released by the original authors, meanwhile the parameters have been tuned to be optimal. For fair comparisons, all these methods use the same representation dimension as $d = 10 * (\#Communities)$ on the same dataset. Besides, under each kind of setting of parameters for different methods, the experiments are repeated for 20 times and the average results are reported.

C. NODE CLUSTERING

In this subsection, we evaluate the performance of each network embedding method comparatively from applying K-means algorithm to cluster the learned representations of nodes. As each node has a unique ground-truth community label (i.e., cluster label), we use the widely used accuracy (AC) as the evaluation metric. Given a node v_i, let l_i' and l_i be its cluster label and the ground-truth label respectively, then AC can be defined as follows:

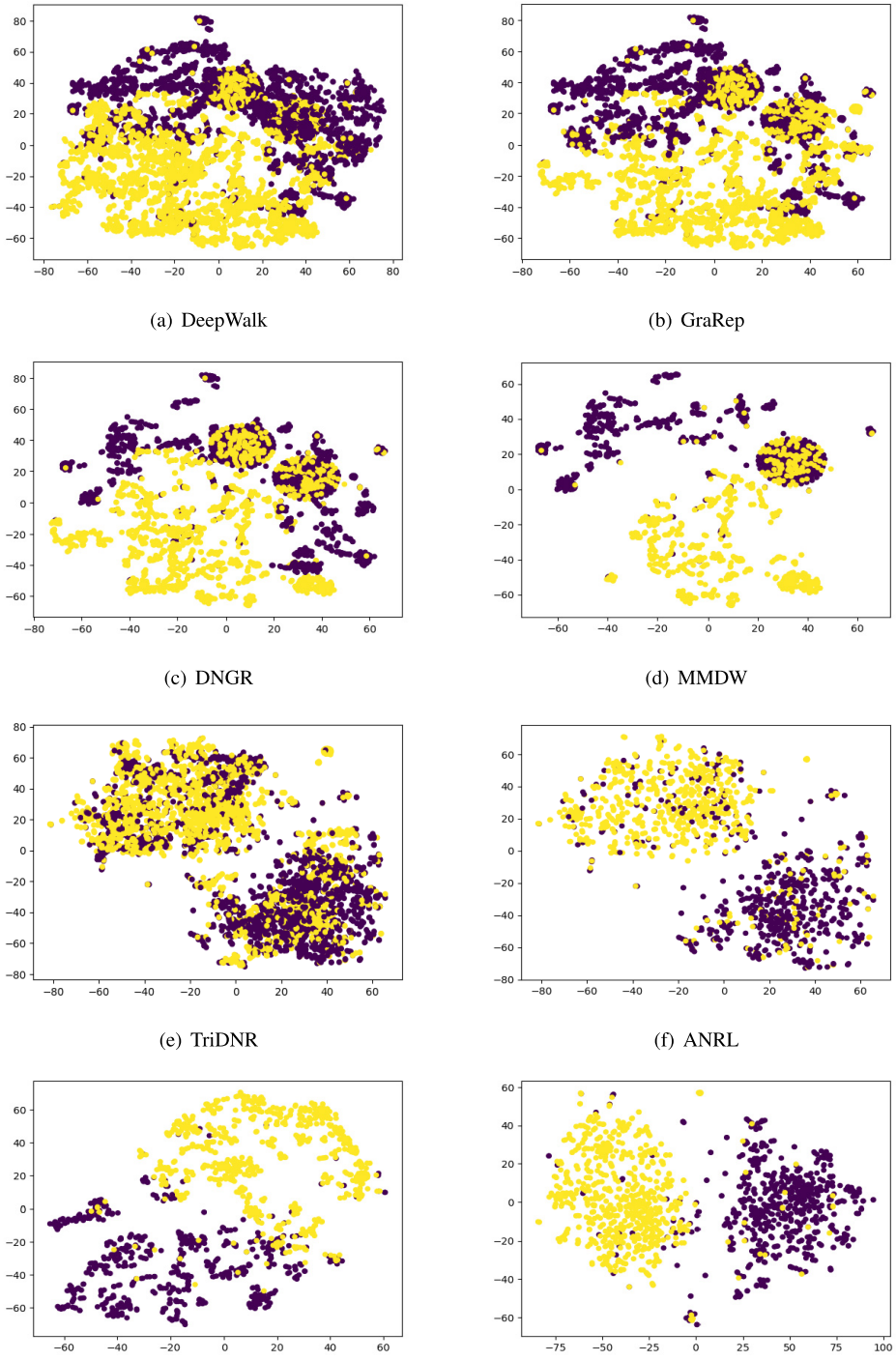
$$AC = \frac{\sum_{i=1}^{n} \delta(l_i, map(l_i'))}{n}, \quad (31)$$

where, n denotes the total number of nodes in the network, $\delta(x, y)$ is the delta function that equals 1 if $x = y$ and equals 0 otherwise, and $map(l_i')$ is the mapping function that maps each cluster label l_i' to equivalent ground-truth label from the network. The best mapping can be efficiently obtained by the Hungarian algorithm [39]. The range of AC is (0, 1], while the larger the AC is, the better the performance of node clustering will be. In our experiments, for a specified dataset, we randomly select different ratios of labeled node pairs as prior information, including 0%, 5%, 10%, 15% and 20% respectively. Note that DeepWalk, GraRep and DNGR are all unsupervised methods that do not consider any prior information, hence we specially set the ratio as 0%. Based on the settings mentioned above, we run every method on four datasets and the results are shown in Table 2. From Table 2, we have the following observations:

TABLE 2. Node clustering results (note that the results of unsupervised methods DeepWalk, GraRep and DNGR keep the same along with different ratios of labeled node pairs, because they do not use any prior information).

Dataset	Method	Ratio of labeled node pairs				
		0%	5%	10%	15%	20%
Polblog	DeepWalk	0.652	0.652	0.652	0.652	0.652
	GraRep	0.631	0.631	0.631	0.631	0.631
	DNGR	0.643	0.643	0.643	0.643	0.643
	MMDW	0.571	0.622	0.681	0.734	0.801
	TriDNR	0.562	0.596	0.634	0.681	0.725
	ANRL	0.569	0.597	0.651	0.704	0.749
	SSNMF	0.572	0.633	0.701	0.762	0.822
	SSKNMF	0.592	0.663	0.734	0.824	0.911
Email	DeepWalk	0.561	0.561	0.561	0.561	0.561
	GraRep	0.521	0.521	0.521	0.521	0.521
	DNGR	0.542	0.542	0.542	0.542	0.542
	MMDW	0.456	0.480	0.502	0.523	0.572
	TriDNR	0.443	0.467	0.491	0.519	0.565
	ANRL	0.451	0.475	0.496	0.522	0.568
	SSNMF	0.467	0.482	0.513	0.534	0.577
	SSKNMF	0.513	0.571	0.625	0.656	0.706
DBLP	DeepWalk	0.553	0.553	0.553	0.553	0.553
	GraRep	0.511	0.511	0.511	0.511	0.511
	DNGR	0.532	0.532	0.532	0.532	0.532
	MMDW	0.419	0.452	0.513	0.566	0.601
	TriDNR	0.403	0.431	0.485	0.526	0.569
	ANRL	0.411	0.443	0.487	0.531	0.573
	SSNMF	0.432	0.464	0.515	0.573	0.612
	SSKNMF	0.491	0.532	0.595	0.667	0.721
YouTube	DeepWalk	0.403	0.403	0.403	0.403	0.403
	GraRep	0.363	0.363	0.363	0.363	0.363
	DNGR	0.371	0.371	0.371	0.371	0.371
	MMDW	0.308	0.331	0.349	0.391	0.436
	TriDNR	0.284	0.302	0.329	0.368	0.411
	ANRL	0.317	0.328	0.331	0.372	0.421
	SSNMF	0.322	0.331	0.353	0.404	0.452
	SSKNMF	0.363	0.394	0.426	0.485	0.551

(1) Prior information can improve the performance of network embedding. Although five semi-supervised methods MMDW, TriDNR, ANRL, SSNMF and SSKNMF are not as good as the unsupervised methods DeepWalk, GraRep and DNGR when the ratio of labeled node pairs is 0%, their performance values improve significantly with the increase of the ratio and eventually surpass those of DeepWalk, GraRep and DNGR. For example, when the ratio of labeled node pairs is 20% on Polblog, the performance values of MMDW, TriDNR, ANRL, SSNMF and SSKNMF improve by 22.9%, 11.2%, 14.9%, 26.1%, and 39.7% respectively compared with the best unsupervised method,



(a) DeepWalk (b) GraRep

(c) DNGR (d) MMDW

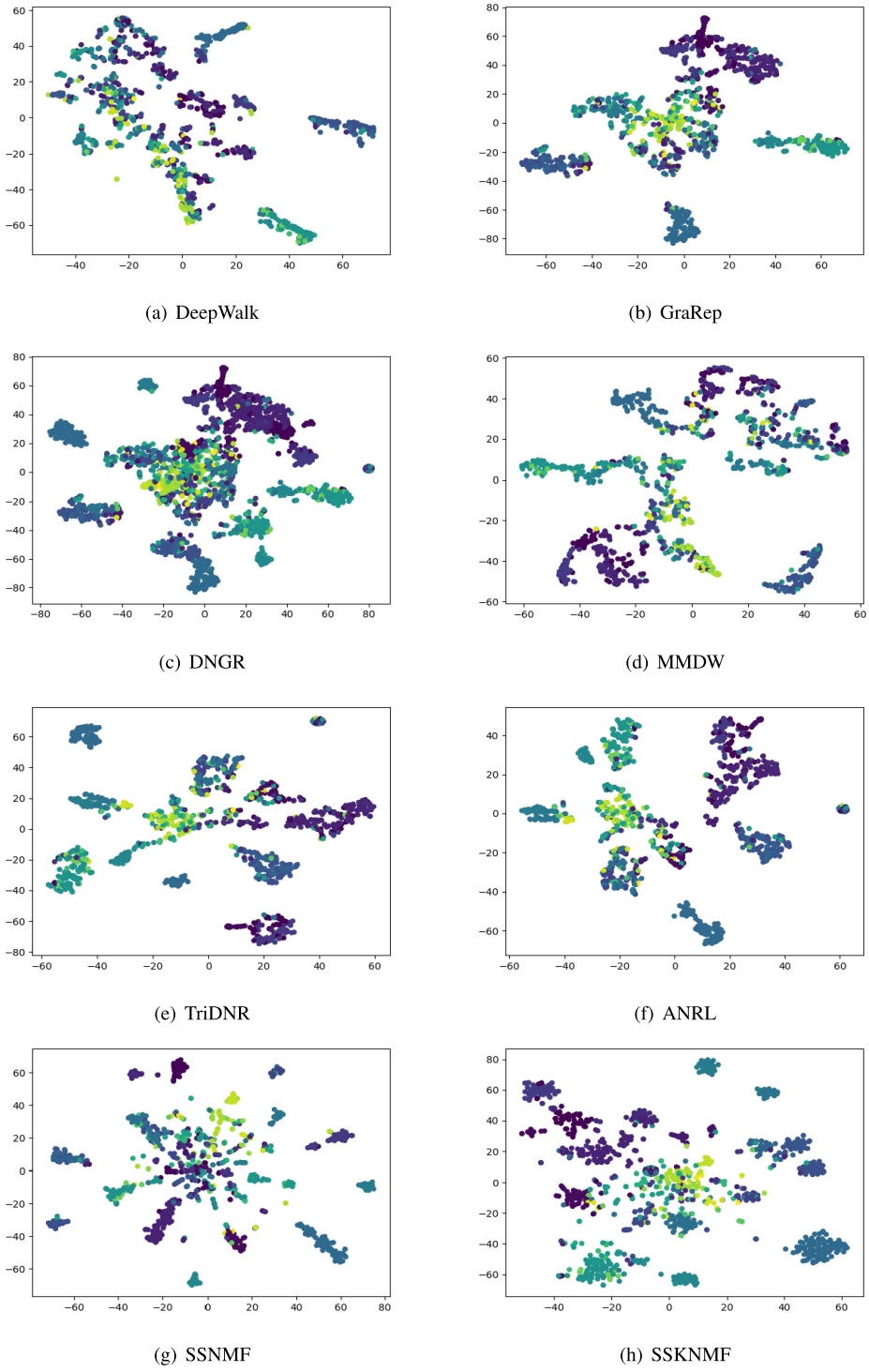
(e) TriDNR (f) ANRL

(g) SSNMF (h) SSKNMF

FIGURE 1. Visualization results of polblog.

DeepWalk. Similar results can also be found on the other three datasets. All these results show that prior information plays a positive role in improving the performance of network embedding.

(2) SSKNMF performs better than the other semi-supervised methods. Comparing with MMDW, TriDNR, ANRL and SSNMF on each dataset, SSKNMF not only maximizes the performance but also performs the best



(a) DeepWalk (b) GraRep

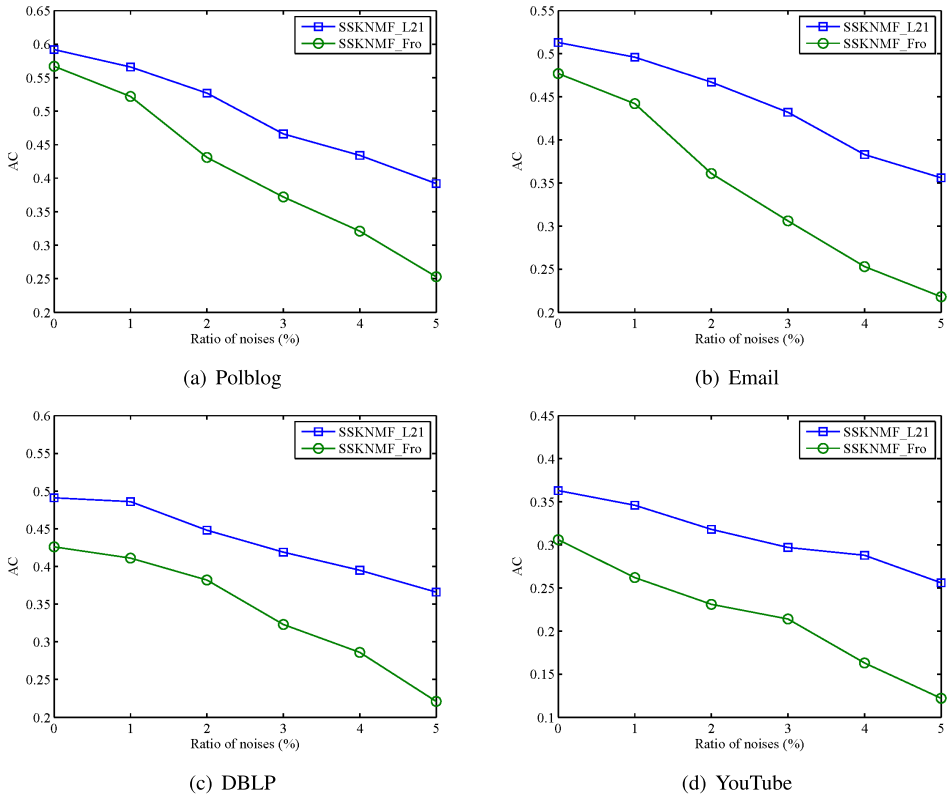
(c) DNGR (d) MMDW

(e) TriDNR (f) ANRL

(g) SSNMF (h) SSKNMF

FIGURE 2. Visualization results of email.

with the increase of the ratio of labeled node pairs. Considering that both SSNMF and MMDW are based on matrix factorization without using kernel method, we can infer that the kernel method effectively helps SSKNMF learn more useful features from the network and thereby helps SSKNMF improve the performance of network embedding. For the other two deep learning based semi-supervised methods TriDNR and ANRL, not only do they perform worse than SSKNMF, they need to spend too much time to tune too many model parameters. On the contrary, SSKNMF has only



(a) Polblog (b) Email

(c) DBLP (d) YouTube

FIGURE 3. Comparisons on robustness.

one parameter λ to be adjusted, which makes it much more efficient.

D. VISUALIZATION

In this part of experiment, we further compare the performance of network embedding methods by visualizing the learned representations of nodes. If a figure can display clear boundaries between different clusters, the learned representations are much more discriminative, which also indicates the corresponding network embedding method performs better. We use t-SNE visualization tool [40] to map the learned representations of nodes into 2D space, where the nodes with the same community label are highlighted with the same color. For the convenience of presentation and analysis, we only show the visualization results of each method on PolBlog and Email with 20% labeled node pairs, both of which have relatively few nodes and communities. The results are shown in Fig. 1 and Fig. 2, respectively. From Fig. 1 and Fig. 2, we can observe that the visualization outputs from SSKNMF display more clear boundaries: nodes in different colors appear in separate areas, and the majority of the nodes with the same color are clustered altogether. On the contrary, the visualization outputs from the other seven baselines, especially the unsupervised methods DeepWalk, GraRep and DNGR, all show unclear boundaries and diffuse clusters, where many nodes with different colors tend to be mixed together. Generally, these visualization results illustrate that the representations learned by our method SSKNMF are more discriminative, which further demonstrates its advantage.

E. ROBUSTNESS TEST

To improve the robustness against noises, we select $L_{2,1}$ norm instead of the widely used Frobenius norm to devise the network embedding model SSKNMF. To validate the effectiveness of this strategy, we specially conduct comparative experiments between using $L_{2,1}$ norm (SSKNMF_L21) and using Frobenius norm (SSKNMF_Fro). For each dataset, we firstly define a so-called cannot-link pairwise constraints which include pairs of nodes with different community labels. The possible numbers (CL_{pairs}) of cannot-link pairwise constraints on a network with ground-truth communities set $C = \{c_1, \ldots, c_r\}$ can be expressed as:

$$CL_{pairs} = \sum_{i=1}^{r} \sum_{j=i+1}^{r} |c_i||c_j|. \tag{32}$$

Then, we randomly extract the fixed ratios (including 6 levels: 0%, 1%, 2%, 3%, 4% and 5%) of cannot-link pairwise constraints to establish virtual links, which can be regarded as noises. Finally, we evaluate the performances of SSKNMF_L21 and SSKNMF_Fro on node clustering tasks. Note that we do not use any prior information for the convenience of comparisons. The results on all the different datasets are shown in Fig. 3. As we can see From Fig. 3, on each dataset, the performance decline of SSKNMF_L21 is

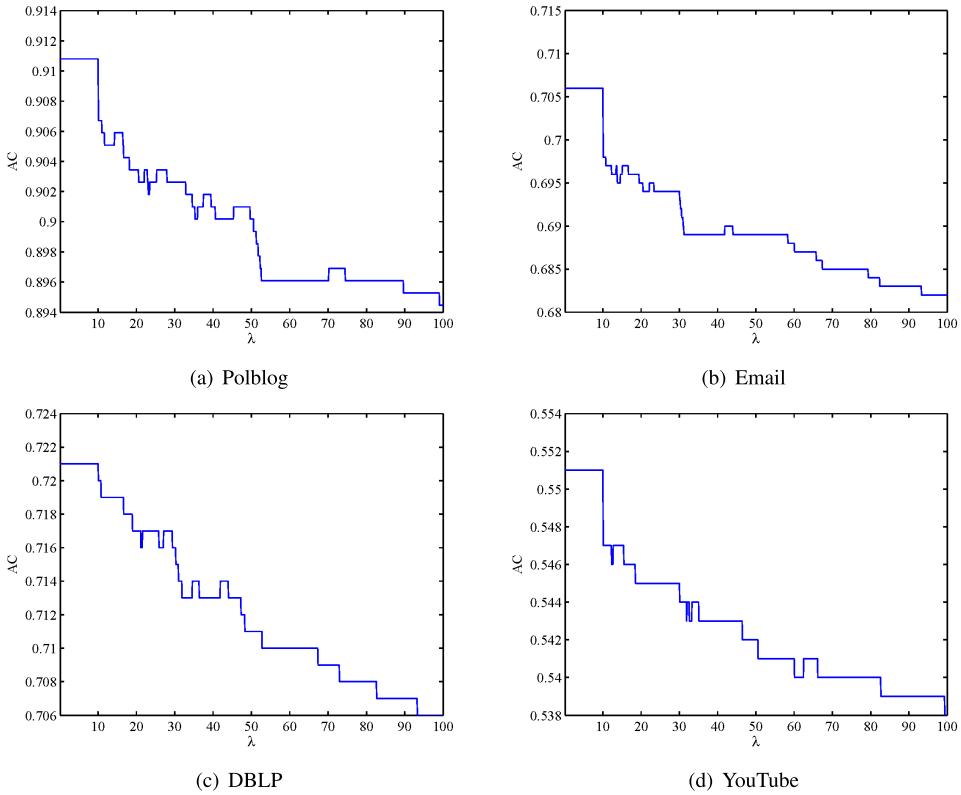


FIGURE 4. AC w.r.t. λ.

much smaller than that of SSKNMF_Fro with the increase of the ratio of noises. For example, when the ratio of noises is 5% on YouTube dataset, the AC value of SSKNMF_L21 is 0.256, which is 29.4% lower than that without noises, but the AC value of SSKNMF_Fro is 0.122, which is 60.1% lower than that without noises. We can also find similar results on the other three datasets. All the results show that SSKNMF_L21 has better robustness than SSKNMF_Fro. This makes SSKNMF_L21 more suitable to be applied to real-world networks that are usually noisy.

F. PARAMETER SENSITIVITY

In the SSKNMF network embedding model, parameter λ is used to balance the contribution of prior information. To test its sensitivity, for each dataset we fix the ratio of labeled node pairs to 20% and test the performance of node clustering with different λ (starting from 0.1 to 100, increasing by 0.1 each time). Results are shown in Fig. 4. We can observe that: on each dataset the performance of SSKNMF is stable and the best when λ ranges from 0.1 to 10. Beyond this range, the performance of SSKNMF generally presents a decline trend with the increase of λ. This phenomenon means that a fixed parameter λ is suitable for different datasets. Therefore, it is reasonable to set λ to be 1 in the experiments described above, which also takes into account the convenience of computation.

V. CONCLUSIONS

In this paper, we propose a network embedding method using semi-supervised kernel NMF (SSKNMF), which specially focuses on how to use prior information available in the network to further improve the performance of network embedding. SSKNMF uses semi-supervised NMF framework to incorporate prior information flexibly, and applies kernel method to strengthen the representations learned from the network. Moreover, SSKNMF improves the robustness against noises by using $L_{2,1}$ norm. Extensive comparative experiments also illustrate its superiority. It can be noticed that the time complexity of SSKNMF is $O(tn^2d)$, which restricts SSKNMF to be applied effectively in large-scale networks. In our future work, we will focus on how to improve the efficiency of SSKNMF, while the implementations on distributed computing frameworks (e.g., Spark) will be carried out similarly as what we have made before in our previous work [41], [42]. Besides, to further explore the kernel of SSKNMF, we will also conduct more comparative experiments by using different kernel functions.

REFERENCES

[1] Z. Zhang, X. Li, and C. Gan, "Multimodality fusion for node classification in D2D communications," *IEEE Access*, vol. 6, pp. 63748–63756, Oct. 2018.

[2] S. Fortunato and D. Hric, "Community detection in networks: A user guide," *Phys. Rep.*, vol. 659, pp. 1–44, Nov. 2016.

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

[4] H. Cai, V. W. Zheng, and K. C.-C. Chang, "A comprehensive survey of graph embedding: Problems, techniques, and applications," *IEEE Trans. Knowl. Data Eng.*, vol. 30, no. 9, pp. 1616–1637, Sep. 2018.

[5] P. Goyal and E. Ferrara, "Graph embedding techniques, applications, and performance: A survey," *Knowl.-Based Syst.*, vol. 151, pp. 78–94, Jul. 2018.

[6] D. Zhang, J. Yin, X. Zhu, and C. Zhang, "Network representation learning: A survey," *IEEE Trans. Big Data*, to be published. doi: 10.1109/TBDATA.2018.2850013.

[7] 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*, New York, NY, USA, Aug. 2014, pp. 701–710.

[8] J. Tang, M. Qu, M. Wang, M. Zhang, Y. Jun, and Q. Mei, "LINE: Large-scale information network embedding," in *Proc. 24th Int. Conf. World Wide Web*, Florence, Italy, May 2015, pp. 1067–1077.

[9] S. Cao, W. Lu, and Q. Xu, "GraRep: Learning graph representations with global structural information," in *Proc. 24th ACM Int. Conf. Inf. Knowl. Manage.*, Melbourne, VIC, Australia, Oct. 2015, pp. 891–900.

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

[11] D. D. Lee and H. S. Seung, "Learning the parts of objects by non-negative matrix factorization," *Nature*, vol. 401, pp. 788–791, Oct. 1999.

[12] H. Liu, Z. Wu, X. Li, D. Cai, and T. S. Huang, "Constrained nonnegative matrix factorization for image representation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 7, pp. 1299–1311, Jul. 2012.

[13] Z. Shu, X. Wu, H. Fan, P. Huang, D. Wu, C. Hu, and F. Ye, "Parameter-less auto-weighted multiple graph regularized nonnegative matrix factorization for data representation," *Knowl.-Based Syst.*, vol. 131, pp. 105–112, Sep. 2017.

[14] X. Jiang, X. Hu, and W. Xu, "Microbiome data representation by joint non-negative matrix factorization with Laplacian regularization," *IEEE/ACM Trans. Comput. Biol. Bioinf.*, vol. 14, no. 2, pp. 353–359, Mar./Apr. 2017.

[15] G. Trigeorgis, K. Bousmalis, S. Zafeiriou, and B. W. Schuller, "A deep matrix factorization method for learning attribute representations," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 3, pp. 417–429, Mar. 2017.

[16] D. Zhang, Z.-H. Zhou, and S. Chen, "Non-negative matrix factorization on kernels," in *Proc. 9th Pacific Rim Int. Conf. Artif. Intell.*, Guilin, China, Aug. 2006, pp. 404–412.

[17] D.-Q. Zhang and S.-C. Chen, "Clustering incomplete data using kernel-based fuzzy C-means algorithm," *Neural Process. Lett.*, vol. 18, pp. 155–162, Dec. 2003.

[18] H. Lee, A. Cichocki, and S. Choi, "Kernel nonnegative matrix factorization for spectral EEG feature extraction," *Neurocomputing*, vol. 72, nos. 13–15, pp. 3182–3190, Aug. 2009.

[19] W.-S. Chen, Y. Zhao, B. Pan, and B. Chen, "Supervised kernel nonnegative matrix factorization for face recognition," *Neurocomputing*, vol. 205, pp. 165–181, Sep. 2016.

[20] F. Zhu and P. Honeine, "Online kernel nonnegative matrix factorization," *Signal Process.*, vol. 131, pp. 143–153, Feb. 2017.

[21] J. Li, J. Zhu, and B. Zhang, "Discriminative deep random walk for network classification," in *Proc. 54th Annu. Meeting Assoc. Comput. Linguistics*, Berlin, Germany, Aug. 2016, pp. 1004–1013.

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

[23] X. Wang, P. Cui, J. Wang, J. Pei, W. Zhu, and S. Yang, "Community preserving network embedding," in *Proc. AAAI Conf. Artif. Intell.*, San Francisco, CA, USA, Feb. 2017, pp. 203–209.

[24] D. Zhang, J. Yin, X. Zhu, and C. Zhang, "Collective classification via discriminative matrix factorization on sparsely labeled networks," in *Proc. 25th ACM Int. Conf. Inf. Knowl. Manage.*, Indianapolis, IN, USA, Oct. 2016, pp. 1563–1572.

[25] S. Cao, W. Lu, and Q. Xu, "Deep neural networks for learning graph representations," in *Proc. 30th AAAI Conf. Artif. Intell.*, Phoenix, AZ, USA, Feb. 2016, pp. 1145–1152.

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

[27] T. Li, J. Zhang, S. Y. Philip, Y. Zhang, and Y. Yan, "Deep dynamic network embedding for link prediction," *IEEE Access*, vol. 6, pp. 29219–29230, 2018.

[28] C. Tu, W. Zhang, Z. Liu, and M. Sun, "Max-margin deepwalk: Discriminative learning of network representation," in *Proc. 25th Int. Joint Conf. Artif. Intell.*, New York, NY, USA, Jul. 2016, pp. 3889–3895.

[29] X. Huang, J. D. Li, and X. Hu, "Label informed attributed network embedding," in *Proc. 10th ACM Int. Conf. Web Search Data Mining*, Cambridge, U.K., Feb. 2017, pp. 731–739.

[30] Z. Yang, W. W. Cohen, and R. Salakhutdinov, "Revisiting semi-supervised learning with graph embeddings," in *Proc. 33rd Int. Conf. Int. Conf. Mach. Learn.*, New York, NY, USA, Jun. 2016, pp. 40–48.

[31] S. Pan, J. Wu, X. Zhu, C. Zhang, and Y. Wang, "Tri-party deep network representation," in *Proc. 25th Int. Joint Conf. Artif. Intell.*, New York, NY, USA, Jul. 2016, pp. 1895–1901.

[32] C. Yang, Z. Liu, D. Zhao, M. Sun, and E. Y. Chang, "Network representation learning with rich text information," in *Proc. 24th Int. Joint Conf. Artif. Intell.*, Buenos Aires, Argentina, Jul. 2015, pp. 2111–2117.

[33] C. Ding, T. Li, and M. I. Jordan, "Convex and semi-nonnegative matrix factorizations," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 32, no. 1, pp. 45–55, Jan. 2010.

[34] D. D. Lee and H. S. Seung, "Algorithms for non-negative matrix factorization," in *Proc. 13th Int. Conf. Neural Inf. Process. Syst.*, Denver, CO, USA, Dec. 2000, pp. 535–541.

[35] L. A. Adamic and N. Glance, "The political blogosphere and the 2004 U.S. election: Divided they blog," in *Proc. 3rd Int. Workshop Link Discovery*, Chicago, IL, USA, Aug. 2005, pp. 36–43.

[36] H. Yin, A. R. Benson, J. Leskovec, and D. F. Gleich, "Local higher-order graph clustering," in *Proc. 23rd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Halifax, NS, Canada, Aug. 2017, pp. 555–564.

[37] J. Yang and J. Leskovec, "Defining and evaluating network communities based on ground-truth," in *Proc. 12th IEEE Int. Conf. Data Mining*, Brussels, Belgium, Dec. 2012, pp. 745–754.

[38] Z. Zhang, H. Yang, J. Bu, S. Zhou, P. Yu, J. Zhang, M. Ester, and C. Wang, "ANRL: Attributed network representation learning via deep neural networks," in *Proc. 27th Int. Joint Conf. Artif. Intell.*, Melbourne, VIC, Australia, Aug. 2017, pp. 3155–3161.

[39] H. W. Kuhn, "The Hungarian method for the assignment problem," *Naval Res. Logistics*, vol. 52, pp. 7–21, Feb. 2005.

[40] L. van der Maaten and G. Hinton, "Visualizing data using t-SNE," *J. Mach. Learn. Res.*, vol. 9, pp. 2579–2605, Nov. 2008.

[41] C. He, X. Fei, H. Li, Y. Tang, H. Liu, and S. Liu, "Improving NMF-based community discovery using distributed robust nonnegative matrix factorization with SimRank similarity measure," *J. Supercomput.*, vol. 74, no. 10, pp. 5601–5624, Oct. 2018.

[42] C. He, H. Li, X. Fei, A. Yang, Y. Tang, and J. Zhu, "A topic community-based method for friend recommendation in large-scale online social networks," *Concurrency Comput., Pract. Exper.*, vol. 29, no. 6, p. e3924, Mar. 2017.

CHAOBO HE received the B.S., M.S., and Ph.D. degrees from South China Normal University, China, in 2004, 2007, and 2014, respectively. He is currently an Associate Professor with the Zhongkai University of Agriculture and Engineering, China, and is also a Visiting Scholar with the School of Data and Computer Science, Sun Yat-sen University, China. His research interests include data mining and social computing. He has published more than 20 papers on international journals and conferences.

QIONG ZHANG received the Ph.D. degree from Laval University, QC, Canada, in 2016. He was a Visiting Researcher with McGill University, QC, Canada, from 2015 to 2016. He was a Research Scientist with the School of Data and Computer Science, Sun Yat-sen University, China, from 2017 to 2019. His research interests include computer vision, image processing, pattern recognition, artificial intelligence, and data mining.

YONG TANG received the B.S. and M.Sc. degrees from Wuhan University, in 1985 and 1990, respectively, and the Ph.D. degree from the University of Science and Technology of China, in 2001, all in computer science. He serves as the Director of the Services Somputing Engineering Research Center of Guangdong Province. He was the Vice Dean of the School of Information Science and Technology, Sun Yat-sen University. He joined SCNU in 2009. He has published more than 200 papers and books. He is currently a Professor and the Dean of the School of Computer Science, South China Normal University (SCNU). As a Supervisor, he has more than 40 Ph.D. students and Postdoctoral Researchers since 2003 and more than 100 master's students, since 1996. He is a Distinguished Member and the Vice Director of the Technical Committee on Collaborative Computing of China Computer Federation. He has also served as a General or a Program Committee Co-Chair of more than 10 conferences.

SHUANGYIN LIU received the Ph.D. degree from the College of Information and Electrical Engineering, China Agricultural University, in 2014. He is currently a Professor with the School of Information Science and Technology, Zhongkai University of Agriculture and Engineering. His current research interests include intelligent information system of agriculture, artificial intelligence, software engineering, and computational intelligence.

HAI LIU received the Ph.D. degree from the School of Data and Computer Science, Sun Yat-sen University, China, in 2010. He is currently an Associate Professor with the School of Computer Science, South China Normal University. His current research interests include machine learning, data mining, and big data.