The Thirty-Second AAAI Conference
on Artificial Intelligence (AAAI-18)

Multi-Facet Network Embedding: Beyond the General Solution of Detection and Representation

Liang Yang,[1,2] **Yuanfang Guo,**[2] **Xiaochun Cao**[2,*]
[1]School of Computer Science and Engineering, Hebei University of Technology
[2]State Key Laboratory of Information Security, Institute of Information Engineering, Chinese Academy of Sciences
{yangliang, guoyuanfang, caoxiaochun}@iie.ac.cn
[*]Corresponding author.

Abstract

In network analysis, community detection and network embedding are two important topics. Community detection tends to obtain the most noticeable partition, while network embedding aims at seeking node representations which contains as many diverse properties as possible. We observe that the current community detection and network embedding problems are being resolved by a general solution, i.e., *"maximizing the consistency between similar nodes while maximizing the distance between the dissimilar nodes"*. This general solution only exploits the most noticeable structure (facet) of the network, which effectively satisfies the demands of the community detection. Unfortunately, most of the specific embedding algorithms, which are developed from the general solution, cannot achieve the goal of network embedding by exploring only one facet of the network. To improve the general solution for better modeling the real network, we propose a novel network embedding method, Multi-facet Network Embedding (MNE), to capture the multiple facets of the network. MNE learns multiple embeddings simultaneously, with the Hilbert Schmidt Independence Criterion (HSIC) being the a diversity constraint. To efficiently solve the optimization problem, we propose a Binary HSIC with linear complexity and solve the MNE objective function by adopting the Augmented Lagrange Multiplier (ALM) method. The overall complexity is linear with the scale of the network. Extensive results demonstrate that MNE gives efficient performances and outperforms the state-of-the-art network embedding methods.

Introduction

In real world, many complex systems, such as the gene regulatory network and human relations, can be described by a network model. A common property of these networks is the presence of community (modular/cluster) structures (Girvan and Newman 2002), in which the nodes are densely connected within each of the communities and seldomly connected across the communities. In the past decades, numerous algorithms, which usually classify nodes into clusters (communities) directly, have been proposed for community detection (Fortunato 2010; Malliaros and Vazirgiannis 2013). Compared to the techniques in computer vision and natural language processing field, which usually perform feature extraction and then a machine learning step (such as regression, classification, clustering, etc.), community detection algorithms are usually End-to-End without explicitly seeking the node representations. Hence, it requires specially designed algorithms rather than directly adopting the existing techniques in other fields. To alleviate the abovementioned issue, network embedding, which learns the node representations regardless of the specific network analysis tasks, becomes popular (Perozzi, Al-Rfou, and Skiena 2014; Tang et al. 2015; Cao, Lu, and Xu 2015; Wang, Cui, and Zhu 2016; Ou et al. 2016).

In general, community detection and network embedding possess different objectives. Community detection usually obtains the most noticeable cluster or one specific cluster with labels and node attributes. Network embedding, on the other hand, aims to represent nodes by vectors which encodes as many diverse properties as possible, because the subsequent task is unknown and may focuses on any properties. For example, embedding methods need to exploit different properties of user in the social network, such as hobbies (political views) to recommend entertainment products (predict the result of presidential election).

However, we observe that the current community detection and network embedding problems are being resolved by a general solution. On one hand, stochastic community detection and graph cuts tend to reconstruct the (normalized) adjacency matrix with the node membership matrix. It is equivalent to maximizing the membership consistency of each pair of the directly connected nodes while maximizing the membership distance of each pair of the un-connected nodes. On the other hand, inspired by the Skip-gram model and negative sampling, network embedding maximizes the embedding similarity of neighbouring nodes in the random walk, while maximizes the embedding distance for each randomly sampled pair of nodes, which are highly unlikely to be similar. Therefore, a general solution, *"maximizing the consistency between similar nodes while maximizing the distance between the dissimilar nodes"*, can be summarized for the existing community detection and network embedding problems. This general solution, which exploits the most noticeable structure (facet) of the network, can explicitly achieves the objective of community detection.

Hence, a key issue, *"Do network embedding algorithms really achieve their goal?"*, has been seldomly considered. Similar to community detection, network embedding algo-

rithms developed from the general solution obtain the embedding which only represents one noticeable facet of the network but ignore the need for diversity.

However, the network in real world possesses multiple facets and can be partitioned based on various principles, each of which has reasonable explanations. To exploit the multiple diverse facets of the network in real situations, we propose a novel network embedding approach, Multi-facet Network Embedding (MNE). Instead of dividing the network into only one group of communities which only reveals one facet of the network, MNE partitions the network into many diverse groups of communities and represents the nodes via multiple embeddings. Specifically, MNE factorizes one network proximity matrix to obtain multiple pairs of node and context embeddings to model the multiple facets of the network. To improve the diversities of different facets, we propose a Binary Hilbert-Schmidt Independence Criterion (B-HSIC) term, which reduces the complexity of HSIC from $O(N^2)$ to $O(N)$. Further, we augment the objective function with an auxiliary variable and solve it with a linear complexity by employing the Augmented Lagrange Multiplier (ALM) method.

The contributions of this paper are summarized as follows:

- We observe that there exists similarities and dissimilarities among the existing network embedding and community detection algorithms. Based on our observation, we summarize a general solution for the community detection and network embedding problems.
- Since the general solution can only model a single facet of the network when it is applied to the network embedding problem, we propose Multi-faceted Network Embedding (MNE) to capture the multiple facets of the network.
- We propose a new regularization term, Binary Hilbert-Schmidt Independence Criterion, with linear complexity to enforce the diversities among different embeddings.
- We propose an efficient optimization approach to optimize the objective function with a linear complexity by adopting the ALM approach.

From Detection to Representation

Here, an undirected and unweighted graph $G = (V, E)$ is considered, where $V = \{v_1, v_2, ..., v_N\}$ is the set of N vertices, and $E = \{e_{ij}\}$ is the set of M edges. The adjacency matrix of G is defined as a nonnegative symmetric matrix $\mathbf{A} = [a_{ij}] \in \mathbb{R}_+^{N\times N}$, where $a_{ij} = 1$ if vertices i and j are connected or $a_{ij} = 0$ otherwise. Note that $a_{ii} = 0$ for all $1 \le i \le N$. The degree matrix of G is defined by a diagonal matrix $\mathbf{D} = diag(d_1, d_2, ..., d_N)$ where $d_i = \sum_j a_{ij}$ is the degree of node v_i. The Laplacian matrix of G is a symmetric matrix defined as $\mathbf{L} = \mathbf{D} - \mathbf{A}$.

Stochastic Community Detection

Intuitively, community detection (graph partition) is to obtain a group of K modules or communities $\{V_i\}_{i=1}^K$, which are the subgraphs (of G) whose vertices are densely connected with each other in the same subgraph. The communities may be either disjoint ($V_i \cap V_j = \emptyset$ for $i \neq j$) or overlapping ($V_i \cap V_j \neq \emptyset$).

From the stochastic perspective (Psorakis et al. 2011), a_{ij} can be considered as the probability of the vertices i and j being connected. This probability can be further considered to be determined by the probability that vertices i and j generate edges belonging to the same community. The latent variables $\mathbf{U} = [u_{ik}] \in \mathbb{R}_+^{N\times K}$ ($\mathbf{V} = [v_{ik}] \in \mathbb{R}_+^{N\times K}$) are introduced, where u_{ik} (v_{ik}) represents the probability that node i belongs to the in- (out-) community k. Then the probability, that vertices i and j is connected by a link belonging to the community k, is $u_{ik}v_{jk}$, and the probability that they are connected is: $\hat{a}_{ij} = \mathbf{u_i}\mathbf{v_j}^T = \sum_{k=1}^K u_{ik}v_{jk}$. Therefore, the community detection problem can be formulated as a matrix factorization. The objective function based on the square loss function is

$$\begin{aligned} &||\mathbf{A} - \mathbf{U}\mathbf{V}^T||_F^2 \\ &= \sum_{(i,j)\in E} (1 - \mathbf{u_i}\mathbf{v_j}^T)^2 + \sum_{(i,j)\notin E} (0 - \mathbf{u_i}\mathbf{v_j}^T)^2, \end{aligned} \tag{1}$$

which is often employed to quantify the reconstruction error. If the in-community and out-community are merged, i.e. $\mathbf{U} = \mathbf{V}$, (Psorakis et al. 2011) simplifies Eq. (1) to

$$\sum_{(i,j)\in E} (1 - \mathbf{u_i}\mathbf{u_j}^T)^2 + \sum_{(i,j)\notin E} (0 - \mathbf{u_i}\mathbf{u_j}^T)^2. \tag{2}$$

By defining the normalized first-order similarity matrix as $\mathbf{S} = \mathbf{D}^{-1/2}\mathbf{A}\mathbf{D}^{-1/2}$, (Yang et al. 2012) considers the t^{th}-order similarity between nodes as $(\alpha\mathbf{S})^t$, where $\alpha \in (0, 1)$ is a decay parameter controlling the similarity scale. The overall similarity matrix is obtained by summing over R high-order similarities as

$$\mathbf{B} = \sum_{t=1}^{R} (\alpha\mathbf{S})^t = \alpha\mathbf{S} + \alpha^2\mathbf{S}^2 + ... + \alpha^R\mathbf{S}^R. \tag{3}$$

By factoring the similarity matrix $\mathbf{B}$ according to Eq. (2), the community based on high-order similarities can be obtained.

Graph Cuts

Normalized Cut (NCut) (Shi and Malik 2000), one of the commonly-employed spectral clustering techniques, aims to minimize the number of links between the detected community and its complementary set as

$$\mathrm{NCut}(V_1, V_2, ..., V_K) = \sum_{k=1}^{K} \frac{link(V_i, \bar{V}_i)}{vol(V_i)},$$

where $\bar{V}_i$ is the complement of V_i ($V_i \cup \bar{V}_i = V, V_i \cap \bar{V}_i = \emptyset$), $vol(A) = \sum_{i\in A} d_i$ and $link(A, B) = \frac{1}{2}\sum_{i\in A, j\in B} a_{ij}$. Then, it is relaxed to minimize $\mathrm{Tr}(\mathbf{U}^T\mathbf{L}\mathbf{U})$ subjecting to $\mathbf{U}^T\mathbf{D}\mathbf{U} = \mathbf{I}$, whose solution yields $\mathbf{U}$ to consist the eigenvectors corresponding to the K largest eigenvalues of the normalized adjacency matrix $\mathbf{Q} = \mathbf{D}^{-1}\mathbf{A}$. The eigenvalue decomposition (EVD) of $\mathbf{Q}$ is defined as

$$\mathbf{Q} = \mathbf{D}^{-1}\mathbf{A} = \mathbf{Y}\mathbf{\Lambda}\mathbf{Y}^T, \tag{4}$$

where $\mathbf{\Lambda} = diag\{\lambda_1, \lambda_2, ..., \lambda_N\}$, $\lambda_1 \geq \lambda_2 \geq ... \geq \lambda_N$ are the N eigenvalues of $\mathbf{D}^{-1}\mathbf{A}$ and $\mathbf{Y} = [\mathbf{Y}_1, \mathbf{Y}_2, ..., \mathbf{Y}_N]$ contains the corresponding eigenvectors. Let $\mathbf{\Lambda}_K$ denote the diagonal matrix with the largest K eigenvalues in $\mathbf{\Lambda}$ and $\mathbf{Y}_K \in \mathbb{R}^{N\times K}$ be the corresponding eigenvectors. According to (Tian et al. 2014), Theorem 1 can be obtained.

Theorem 1 (Tian et al. 2014)**.** *$\mathbf{Y}_K\mathbf{\Lambda}_K\mathbf{Y}_K^T$ is the best reconstruction of matrix $\mathbf{D}^{-1}\mathbf{A}$ in terms of the Frobenuis norm among all rank-k matrices.*

It indicates that NCut can be regarded as a process of matrix reconstruction for the matrix $\mathbf{D}^{-1}\mathbf{A}$.

Network Embedding

Different from community detection which directly finds a group of modules, network embedding aims to obtain the representation for each node which can be employed in future processing steps such as clustering, classification, link prediction, recommendation, etc. In network embedding, a representation (embedding) matrix $\mathbf{H} = [h_{ip}] \in \mathbb{R}^{N\times P}$, where $\mathbf{h}_i \in \mathbb{R}^P$ is the P-dimensional embedding of vertex i, is defined. Next, DeepWalk, LINE and GreRep are selected as the examples to briefly introduce network embedding.

DeepWalk (Perozzi, Al-Rfou, and Skiena 2014) is motivated by the fact that the frequency of the vertices, appearances in the short random walks follows a power-law distribution (i.e. scale-free). This power-law is also obeyed by the word frequency in natural languages. Based on this interesting observation, DeepWalk employs Skip-gram, a word representation model (Mikolov et al. 2013), to learn the representations of the nodes. The Skip-gram model, as shown in Eq. (5), achieves the embedding by maximizing the co-occurrence probabilities among the nodes which appear within a window in the random walk.

$$\prod_{v_i\in Walk}\Big[\prod_{v_j\in Window(v_i)} p(v_j|v_i;\mathbf{H},\mathbf{C})\Big], \tag{5}$$

where $Walk$ is a random walk and $Window(v_i)$ is the content of node v_i. $\mathbf{C} = [c_{ip}] \in \mathbb{R}^{N\times P}$ is the corresponding content embedding. The condition probability $p(v_j|v_i;\mathbf{H})$ is modeled as $p(v_j|v_i;\mathbf{H},\mathbf{C}) = \exp(\mathbf{h}_i\mathbf{c}_j^T)/\sum_{j'\in C}\exp(\mathbf{h}_i\mathbf{c}_{j'}^T)$. With the logarithm of Eq. (5) and negative sampling, the final objective function can be obtained as follows.

$$\sum_{(v_j,v_i)\in D} \log\sigma(\mathbf{h}_i\mathbf{c}_j^T) + \lambda E_{v_{j'}\sim P(V)}\log\sigma(-\mathbf{h}_i\mathbf{c}_{j'}^T), \tag{6}$$

where $\sigma(x) = 1/(1+\exp(-x))$ and D contains all the $(node.content)$ pairs extracted from the random walk. $v_{j'} \sim P(V)$ represents that the negative pairs are sampled according to the distribution of nodes V. (Yang et al. 2015) shows that Skip-gram with softmax is equivalent to factorizing a matrix $\mathbf{M} = [m_{ij}] \in \mathbb{R}^{N\times N}$ with

$$m_{ij} = \log\frac{N(v_i,v_j)}{N(v_i)} = \log\frac{[\mathbf{Q}+\mathbf{Q}^2+...+\mathbf{Q}^W]_{ij}}{W}, \tag{7}$$

where W is the size of the context window and $\mathbf{Q} = \mathbf{D}^{-1}\mathbf{A}$. Here $\mathbf{Q}^t$ can be interpreted as the t^{th} order similarity, i.e., the average probability that the vertex i walks to vertex j with t steps. Thus, m_{ij} is the logarithm of the average probability that vertex i walks to vertex j within W steps.

node2vec (Grover and Leskovec 2016) generalizes DeepWalk with the combination of BFS and DFS random walks. It possesses the same objective function Eq. (6) as DeepWalk except for the set D of $(node, content)$ pairs.

GraRep (Cao, Lu, and Xu 2015) obtains the t^{th}-order embedding $\mathbf{H}^t$ by respectively factorizing each $\log\mathbf{Q}^t, 1 \leq t \leq W$ via singular value decomposition (SVD)

$$\log\mathbf{Q}^t = (\mathbf{D}^{-1}\mathbf{A})^t = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T, \tag{8}$$

and concatenates them to construct the final embedding. This is different from DeepWalk and TADW (Yang et al. 2015) which average all the W high-order similarities Eq. (7) as a overall similarity and factorize it to obtain the embedding.

LINE (Tang et al. 2015) learns the node representations by preserving the first- and second-order proximities, and combines them to form the final representations. It defines the first- and second proximities as

$$\sum_{(v_i,v_j)\in E} a_{ij}\log\frac{1}{1+\exp(-\mathbf{h}_i\mathbf{h}_j^T)},$$
$$\sum_{(v_i,v_j)\in E} a_{ij}\log\frac{\exp(\mathbf{h}_i\mathbf{h}_j^T)}{\sum_{v_{j'}\in V}\exp(\mathbf{h}_i\mathbf{h}_{j'}^T)}. \tag{9}$$

To facilitate the optimization, LINE also adopts the negative sampling, and formulates the final objective function for the second-order proximity as

$$\sum_{(v_j,v_i)\in E} \log\sigma(\mathbf{h}_i\mathbf{h}_j^T) + \lambda E_{v_{j'}\sim P(V)}\log\sigma(-\mathbf{h}_i\mathbf{h}_{j'}^T), \tag{10}$$

where $\sigma(x) = 1/(1+\exp(-x))$. Different from DeepWalk, where node pairs are sampled within the W-sized window in the random walk, LINE samples the node pairs from the network links.

A General Solution

From the objective functions of DeepWalk in Eq. (6) (LINE in Eq. (10)) and the asymmetric matrix factorizations in Eq. (1) (symmetric matrix factorization in Eq. (2)), we can conclude that the objective functions of the community detection methods and network embedding methods are both formed by two components as

$$\sum_{(v_i,v_j)\in S} sim(\mathbf{h}_i,\mathbf{c}_j) + \sum_{(v_i,v_j)\in DS} dis(\mathbf{h}_i,\mathbf{c}_j), \tag{11}$$

where S contains pairs of similar nodes and DS contains pairs of dissimilar nodes. The first component in Eq. (11) maximizes the consistencies between the each pair of similar nodes, such as $-(1-\mathbf{u_i}\mathbf{v_j}^T)^2$ in community detection and $\log\sigma(\mathbf{h}_i\mathbf{c}_j^T)$ in network embedding. The second component maximizes the distances between each pair of the dissimilar nodes. The community detection and network embedding methods can also be considered to be similar from

Table 1: Comparisons of the community detection and network embedding algorithms.

	Objective Functions: $\sum_{(v_i,v_j)\in S} sim(\mathbf{h}_i, \mathbf{c}_j) + \sum_{(v_i,v_j)\in DS} dis(\mathbf{h}_i, \mathbf{c}_j)$				Matrix Factorization Formulation	
	$sim()$	$dis()$	S	DS	Matrix to be factorized	Factorization Formulation
Community Detection	$-(1-\mathbf{h_i c_j}^T)^2$ $-(1-\mathbf{h_i h_j}^T)^2$	$-(0-\mathbf{h_i c_j}^T)^2$ $-(0-\mathbf{h_i h_j}^T)^2$	$(v_i, v_j) \in E$	$(v_i, v_j) \notin E$	$\mathbf{A}$ or $\sum_t (\alpha \mathbf{S})^t$ †	Symmetric / Asymmetric
Graph Cuts	-	-	-	-	$\mathbf{Q}$ †	EVD
DeepWalk	$\log \sigma(\mathbf{h}_i \mathbf{c}_j^T)$	$\log \sigma(-\mathbf{h}_i \mathbf{c}_j^T)$	Random walk window D	Random sampling	$\log \frac{\mathbf{Q}+\mathbf{Q}^2+...+\mathbf{Q}^W}{W}$	Asymmetric
node2vec	$\log \sigma(\mathbf{h}_i \mathbf{c}_j^T)$	$\log \sigma(-\mathbf{h}_i \mathbf{c}_j^T)$	Random walk (BFS & DFS)	Random sampling	-	Asymmetric
LINE (2nd)	$\log \sigma(\mathbf{h}_i \mathbf{h}_j^T)$	$\log \sigma(-\mathbf{h}_i \mathbf{h}_j^T)$	$(v_i, v_j) \in E$	Random sampling	$\log \mathbf{Q}$	Symmetric
GraRep	-	-	-	-	$\log \mathbf{Q}, ... \log \mathbf{Q}^W$	SVD

† $\mathbf{S} = \mathbf{D}^{-1/2}\mathbf{A}\mathbf{D}^{-1/2}$ and $\mathbf{Q} = \mathbf{D}^{-1}\mathbf{A}$ are the normalized adjacency matrces.

the perspective of matrix factorization. As can be observed, both GraRep in Eq. (8) and NCut in Eq. (4) factorize the normalized adjacency matrix $\mathbf{Q} = \mathbf{D}^{-1}\mathbf{A}$, while most of the network embedding methods factorize a combination of $\mathbf{Q}$. We summarize our observations in Table 1.

In summary, Eq. (11) can be considered as a general solution to the current problems of community detection and network embedding. This general solution exploits the most noticeable structure (facet) of the network, which best achieves the objective of Eq. (11). This character exactly meets the needs of the community detection.

Multi-faceted Network Embedding

Although Eq. (11) summarizes the current solutions to the community detection and network embedding problems, it still possesses weakness that the current general solution can only reveal a single facet of the network, i.e., it can only capture one aspect of the network. In this section, we propose a novel network embedding model, named Multi-faceted Network Embedding (MNE), to capture multiple aspects of the network. Firstly, MNE is formulated as a matrix factorization with Hilbert-Schmidt Independence Criterion (HSIC). Then the scalability of HSIC is improved by proposing an approximate Binary-HSIC. At last, we optimize the objective function via ALM and analyze its complexity.

Formulations

Following the notations in the previous section, network embedding is equivalent to factorizing the proximity matrix $\mathbf{M} = [m_{ij}] \in \mathbb{R}^{N\times N}$ into the product of embedding matrix $\mathbf{H} = [h_{ij}] \in \mathbb{R}^{N\times P}$ and context matrix $\mathbf{C} = [c_{ij}] \in \mathbb{R}^{N\times P}$ by minimizing

$$\mathcal{L}_{MF} = ||\mathbf{M} - \mathbf{H}\mathbf{C}^T||_F^2 + \lambda(||\mathbf{H}||_F^2 + ||\mathbf{C}||_F^2),$$

which only explores a single facet of the network by dividing the network into one group of communities. To exploit multiple facets of the network, the network should be divided into multiple groups of communities. Therefore, the proximity matrix is factorized to obtain multiple pairs of node and context embeddings as

$$\sum_i^Z ||\mathbf{M} - \mathbf{H}_{(i)}\mathbf{C}_{(i)}^T||_F^2 + \lambda(||\mathbf{H}_{(i)}||_F^2 + ||\mathbf{C}_{(i)}||_F^2),$$

where Z is the number of facets to be explored. $\mathbf{H}_{(i)} \in \mathbb{R}^{N\times K_{(i)}}$ and $\mathbf{C}_{(i)} \in \mathbb{R}^{N\times K_{(i)}}, i \in \{1, 2, 3..., Z\}$ denote the $K_{(i)}$-dimensional node and context embeddings which encode the information of the i^{th} facets of the network. This objective function induces a trivial solution that all $\mathbf{H}_{(i)}$s are similar, due to the lack of characterizing the relationships among different facets of the network.

To resolve this issue, the relationships among different facets can be utilized to remove the redundancies and select the most diverse facets for the embedding, due to the fact that only finite number of facets will be encoded. Intuitively, more diverse the facets are, better representation of the nodes can be obtained. Taking Facebook100 dataset as an example, each network gives the relationship of students from one university. Network can be partitioned according to students' department, major, high school and year of enrollment, etc. If the highly correlated facets, such as departments and majors, are selected together, the embedding results appear to be redundant. Therefore, the network partitions still require the selected facets to be diverse. To achieve diversity, we propose a regularization term to penalize the similarities between each two groups of the partitions as

$$\mathcal{L}_{similarity} = \sum_{i\neq j} \mathrm{Sim}(\mathbf{H}_{(i)}, \mathbf{H}_{(j)}), \tag{12}$$

where $\mathrm{Sim}(\mathbf{H}_{(i)}, \mathbf{H}_{(j)})$ measures the similarity between the different groups of communities.

Remark 1: The obtained multiple embeddings can jointly improve the performances of many classification tasks. Although the multiple embeddings correspond to the multiple

facets of the network, it is difficult to assign specific semantic labels to each embedding. Therefore, the performance of any specific classification cannot be improved by employing only one of the embeddings.

Remark 2: Although LINE and GraRep can also obtain multiple embeddings, MNE possesses two major differences compared to them. Firstly, LINE and GraRep obtain the different embeddings separately and ignore the redundancies in the embeddings, while MNE explicitly constrains the embeddings to be diverse. Secondly, GraRep obtains multiple embeddings by exploring multiple proximity matrices which induce high complexities (at least $O(N^2)$), while MNE computes with only 1^{st}-order sparse proximity matrix.

Binary-HSIC

Let the embeddings $\mathbf{H}_{(i)}$ and $\mathbf{H}_{(j)}$ contain N data points $(\mathbf{H}_{(i)p}, \mathbf{H}_{(j)p}), p = \{1, 2, 3, ..., N\}$ which drawn from variable $(\mathbf{x}, \mathbf{y}) \in \mathcal{X} \otimes \mathcal{Y}$, respectively. The diversity between $\mathbf{H}_{(i)}$ and $\mathbf{H}_{(j)}$ is measured by the independence between $\mathbf{x}$ and $\mathbf{y}$. Here, the Hilbert-Schmidt Independence Criterion (HSIC) is considered to measure the dependence as in (Gretton et al. 2005). HSIC computes the Frobenius norm of the cross-covariance operator over the domain $\mathcal{X} \times \mathcal{Y}$ in Hilbert space $\mathrm{HSIC}(\mathbf{x}, \mathbf{y}, \mathcal{F}, \mathcal{G}) = ||\mathbf{C}_{\mathbf{xy}}||_F^2$ where $\mathbf{C}_{\mathbf{xy}} = E_{\mathbf{xy}}[(\phi(\mathbf{x}) - \mu_{\mathbf{x}})) \otimes (\varphi(\mathbf{y}) - \mu_{\mathbf{y}})]$. $\phi(\mathbf{x})$ and $\varphi(\mathbf{y})$ are functions from $\mathcal{X}$ and $\mathcal{Y}$ to kernel space $\mathcal{F}$ and $\mathcal{G}$ with $k_1(\mathbf{x}_i, \mathbf{x}_j) =< \phi(\mathbf{x}_i), \phi(\mathbf{x}_j) >$ and $k_2(\mathbf{y}_i, \mathbf{y}_j) =< \phi(\mathbf{y}_i), \phi(\mathbf{y}_j) >$. $\mu_{\mathbf{x}} = E_{\mathbf{x}}[\phi(\mathbf{x})]$ and $\mu_{\mathbf{y}} = E_{\mathbf{y}}[\phi(\mathbf{y})]$. By letting $\phi(\mathbf{x}) = \mathbf{x}$ and $\varphi(\mathbf{y}) = \mathbf{y}$, HSIC can be estimated by

$$\begin{aligned} &\mathrm{HSIC}(\mathbf{H}_{(i)}, \mathbf{H}_{(j)}) = (N-1)^{-2} \operatorname{Tr}(\mathbf{K}_1 \mathbf{R} \mathbf{K}_2 \mathbf{R}) \\ = \quad & \frac{1}{(N-1)^2} \operatorname{Tr}(\mathbf{H}_{(i)}^T [\hat{\mathbf{H}}_{(j)} \hat{\mathbf{H}}_{(j)}^T] \mathbf{H}_{(i)}), \end{aligned}$$

where $\mathbf{K}_1 = \mathbf{H}_{(i)} \mathbf{H}_{(i)}^T$ and $\mathbf{K}_2 = \mathbf{H}_{(j)} \mathbf{H}_{(j)}^T$ are Gram matrices, $\mathbf{R} = \mathbf{I} - \frac{1}{N} \mathbf{e}\mathbf{e}^T$ maintains zero mean for the Gram matrix with $\mathbf{I}$ being the identity matrix and $\mathbf{e}$ is a column vector with every element being 1. $\hat{\mathbf{H}}_{(j)} = \mathbf{R}\mathbf{H}_{(j)}$ is the modified $\mathbf{H}_{(j)}$ with zero mean for each row. If we denote $\mathbf{G}_{(j)} = \hat{\mathbf{H}}_{(j)} \hat{\mathbf{H}}_{(j)}^T$, the (p, q) element of $\mathbf{G}_{(j)}$, whose value varies from -1 to 1, is the similarity between $\mathbf{H}_{(j)p}$ and $\mathbf{H}_{(j)q}$. Note that minimizing $\mathrm{HSIC}(\mathbf{H}_{(i)}, \mathbf{H}_{(j)})$ reduces the similarity between $\mathbf{K}_1 = \mathbf{H}_{(i)} \mathbf{H}_{(i)}^T$ and $\mathbf{G}_{(j)}$, i.e., HSIC constrains the similarity matrices from different facets to be diverse for better robustness to the scale and rotation transformations, rather than directly constrains embedding diverse. Unfortunately,, computing $\mathbf{G}_{(j)}$ demands $O(N^2 K)$ floating point operations, where N is the number of nodes and K is the number of embedding dimensions.

To reduce the complexity of computing $\mathbf{G}_{(j)}$, we introduce a Binary Hilbert-Schmidt Independence Criterion (B-HSIC). Instead of directly computing the embedding similarity matrix $\mathbf{G}_{(j)}$, we compute the binary similarity matrix $\hat{\mathbf{G}}_{(j)}$ according to a fast clustering result

$$\hat{\mathbf{G}}_{(j)pq} = \begin{cases} 1 & \text{if } p, q \text{ belong to the same cluster} \\ 0 & \text{otherwise} \end{cases}. \quad (13)$$

Algorithm 1: Multifaceted Network Embedding

Input: Network proximity matrix $\mathbf{M}$, the number of facet Z, embedding dimension $K_i, i \in \{1, 2..., Z\}$ in i^{th} facet and positive real parameters λ and α

Initialization: Random matrix $\mathbf{H}_{(i)}, \mathbf{C}_{(i)}, \mathbf{X}_{(i)}, \mathbf{Q}_{(i)} \in \mathbb{R}^{N \times k_i}$ and $\rho = 1.1$

while *not converged* **do**
  for $i \in \{1, 2..., Z\}$ **do**
    Compute $\mathbf{T}_i \mathbf{X}_i$ and $\mathbf{T}_i \mathbf{H}_i$ via Eq. (25);
    Update $\mathbf{C}_{(i)}$ via Eq. (21);
    Update $\mathbf{H}_{(i)}$ via Eq. (22);
    Update $\mathbf{X}_{(i)}$ via Eq. (23);
    Update Multiplier $\mathbf{Q}_{(i)}$ and ν via Eq. (24);
  end
end

Output: Multifaceted Network Embedding $\mathbf{H}_{(i)}, i \in \{1, 2..., Z\}$

This clustering result can be obtained via any efficient clustering algorithms. Here, K-means is selected as it only requires $O(NK)$ operations. Then, we define B-HSIC as

$$\text{B-HSIC}(\mathbf{H}_{(i)} | \mathbf{H}_{(j)}) = \frac{1}{(N-1)^2} \operatorname{Tr}(\mathbf{H}_{(i)}^T \hat{\mathbf{G}}_{(j)} \mathbf{H}_{(i)}),$$

where $\mathbf{H}_{(j)}$ is fixed and the similarity between $\mathbf{H}_{(i)}$ and $\mathbf{H}_{(j)}$ is minimized by varying $\mathbf{H}_{(i)}$. By letting

$$\begin{aligned} &\mathrm{Sim}(\mathbf{H}_{(i)}, \mathbf{H}_{(j)}) \\ = \quad & \text{B-HSIC}(\mathbf{H}_{(i)} | \mathbf{H}_{(j)}) + \text{B-HSIC}(\mathbf{H}_{(j)} | \mathbf{H}_{(i)}), \end{aligned}$$

we can rewrite Eq. (12) as

$$\begin{aligned} &\mathcal{L}_{similarity} = \sum_i \sum_{j \neq i} \text{B-HSIC}(\mathbf{H}_{(i)} | \mathbf{H}_{(j)}) \\ = &\sum_i \sum_{j \neq i} \operatorname{Tr}(\mathbf{H}_{(i)}^T \hat{\mathbf{G}}_{(j)} \mathbf{H}_{(i)}) = \sum_i \operatorname{Tr}(\mathbf{H}_{(i)}^T \mathbf{T}_{(i)} \mathbf{H}_{(i)}), \end{aligned}$$

where $\mathbf{T}_{(i)} = \sum_{j=1, j \neq i}^{Z} \hat{\mathbf{G}}_{(j)}$. Then, the overall objective function is

$$\begin{aligned} &\mathcal{L} = \mathcal{L}_{MMF} + \alpha \mathcal{L}_{similarity} \\ = \quad & \sum_i^Z \Big[||\mathbf{M} - \mathbf{H}_{(i)} \mathbf{C}_{(i)}^T||_F^2 + \lambda (||\mathbf{H}_{(i)}||_F^2 + ||\mathbf{C}_{(i)}||_F^2) \\ + \quad & \alpha \operatorname{Tr}(\mathbf{H}_{(i)}^T \mathbf{T}_{(i)} \mathbf{H}_{(i)}) \Big], \end{aligned} \quad (14)$$

where α is the parameter which controls the tradeoff between $\mathcal{L}_{MMF}$ and $\mathcal{L}_{similarity}$.

Optimization

The classic strategy to solve Eq. (14) is to directly employ the alternating direction method which iteratively minimizes the objective function by considering one variable at a time while fixing the others. Specifically, minimizing Eq. (14)

with respect to $\mathbf{C}_{(i)}$ and $\mathbf{H}_{(i)}$ is equivalent to minimizing

$$||\mathbf{M} - \mathbf{H}_{(i)}\mathbf{C}_{(i)}^T||_F^2 + \lambda||\mathbf{C}_{(i)}||_F^2 \quad , \tag{15}$$

$$||\mathbf{M} - \mathbf{H}_{(i)}\mathbf{C}_{(i)}^T||_F^2 + \lambda||\mathbf{H}_{(i)}||_F^2 + \alpha \operatorname{Tr}(\mathbf{H}_{(i)}^T\mathbf{T}_{(i)}\mathbf{H}_{(i)}), \tag{16}$$

which are both convex. By differentiating the two objective functions with respect to $\mathbf{C}_{(i)}$ and $\mathbf{H}_{(i)}$ respectively and setting them to zero, the optimal solution $\mathbf{C}_{(i)}^*$ and $\mathbf{H}_{(i)}^*$ satisfy

$$\mathbf{C}_{(i)}^*(\lambda\mathbf{I} + \mathbf{H}_{(i)}^T\mathbf{H}_{(i)}) = \mathbf{M}^T\mathbf{H}_{(i)}, \tag{17}$$

$$(\lambda\mathbf{I} + \alpha\mathbf{T}_{(i)})\mathbf{H}_{(i)}^* + \mathbf{H}_{(i)}^*(\mathbf{C}_{(i)}^T\mathbf{C}_{(i)}) = \mathbf{M}\mathbf{C}_{(i)}. \tag{18}$$

Eq. (18) is a standard Sylvester equation (Bartels and Stewart 1972). Although it possesses a unique solution, this classic optimization approach requires $O(N^3)$ operations, which induces an inability to process large-scale networks. Here, we propose an efficient algorithm with Augmented Lagrangian method (ALM) (Bertsekas 2014). For simplicity, we omit the subscript $\cdot_{(i)}$ in the following paragraphs.

By introducing an auxiliary variable $\mathbf{X}$, the unconstrained optimization problem Eq. (16) can be transformed to be the following constrained optimization problem

$$\begin{aligned} &\underset{\mathbf{H},\mathbf{C}}{\operatorname{argmin}} ||\mathbf{M} - \mathbf{H}\mathbf{C}^T||_F^2 + \lambda||\mathbf{H}||_F^2 + \alpha \operatorname{Tr}(\mathbf{H}^T\mathbf{T}\mathbf{X}) \\ &s.t.\ \mathbf{H} = \mathbf{X}. \end{aligned} \tag{19}$$

The Lagrangian function of Eq. (19) can be written in the following form:

$$\begin{aligned} \mathcal{L}(\mathbf{H},\mathbf{C},\mathbf{X}) = ||\mathbf{M} - \mathbf{H}\mathbf{C}^T||_F^2 + \lambda||\mathbf{H}||_F^2 \\ +\alpha \operatorname{Tr}(\mathbf{H}^T\mathbf{T}\mathbf{X}) + < \mathbf{Q}, \mathbf{H} - \mathbf{X} > + \frac{\nu}{2}||\mathbf{H} - \mathbf{X}||_F^2, \end{aligned} \tag{20}$$

where $< \cdot, \cdot >$ represents the matrix inner product, ν is a positive penalty scalar and $\mathbf{Q}$ is the Lagrangian multiplier. The proposed solver also iteratively updates one variable at a time by fixing the others. The closed-form solutions to sub-problems are shown as follows.

Sub-problem C: By differentiating the objective function Eq. (15) with respect to $\mathbf{C}$, the closed-form of $\mathbf{C}$ is

$$\mathbf{C} = (\mathbf{M}^T\mathbf{H})(\lambda\mathbf{I} + \mathbf{H}^T\mathbf{H})^{-1}. \tag{21}$$

Sub-problem H: After the terms which are independent to $\mathbf{H}$ are ignored, the sub-problem $\mathbf{H}$ can be obtained,

$$\begin{aligned} \underset{\mathbf{H}}{\operatorname{argmin}} ||\mathbf{M} - \mathbf{H}\mathbf{C}^T||_F^2 + \lambda||\mathbf{H}||_F^2 + \alpha \operatorname{Tr}(\mathbf{H}^T\mathbf{T}\mathbf{X}) \\ + \operatorname{Tr}(\mathbf{Q}^T\mathbf{H}) + \frac{\nu}{2}||\mathbf{X} - \mathbf{H}||_F^2. \end{aligned}$$

Then the optimal $\mathbf{H}$ can be calculated via

$$(2\mathbf{M}\mathbf{C} + \nu\mathbf{X} - \alpha\mathbf{T}\mathbf{X} - \mathbf{Q})(2\mathbf{C}^T\mathbf{C} + (2\lambda + \nu)\mathbf{I})^{-1} \tag{22}$$

Sub-problem X: Similar to the sub-problem $\mathbf{H}$, $\mathbf{X}$ can also be obtained by selecting the terms related to $\mathbf{X}$:

$$\underset{\mathbf{X}}{\operatorname{argmin}} \operatorname{Tr}(\alpha\mathbf{H}^T\mathbf{T}\mathbf{X} + \mathbf{Q}^T(\mathbf{H} - \mathbf{X})) + \frac{\nu}{2}||\mathbf{X} - \mathbf{H}||_F^2.$$

Its closed-form solution can be obtained via

$$\mathbf{X} = [(\nu - \alpha\mathbf{T})\mathbf{H} + \mathbf{Q}]/\nu. \tag{23}$$

Multipliers Q and ν: The multiplier $\mathbf{Q}$ and ν are updated according to

$$\begin{aligned} \mathbf{Q} = \mathbf{Q} + \nu(\mathbf{H} - \mathbf{X}), \\ \nu = \rho\nu. \end{aligned} \tag{24}$$

Although the multiplications between $\mathbf{T}_{(i)} = \sum_{j=1,j\neq i}^{Z} \hat{\mathbf{G}}_{(j)}$ and $\mathbf{H}$ (or $\mathbf{X}$) still require $O(N^2K)$ operations since $\mathbf{T}_{(i)} \in \mathbb{R}^{N\times N}$ and $\mathbf{H}_i \in \mathbb{R}^{N\times K}$, we can significantly reduce the complexity by analyzing the structure of $\mathbf{T}_{(i)}$. In Eq. (13) , $\hat{\mathbf{G}}_{(j)}$ is sparse and $\hat{\mathbf{G}}_{(j)pq} = 1$ if and only if nodes v_p and v_q are in the same cluster. Then, the elements in the p^{th} row of $\hat{\mathbf{G}}_{(j)}$ are all zero except for those in the same cluster as v_p. Therefore, $\mathbf{Y}_{(j)} = \hat{\mathbf{G}}_{(j)}\mathbf{H}$ is equivalent to setting the p^{th} row of the result matrix $\mathbf{Y}_{(j)}$ as

$$\mathbf{Y}_{(j)p\cdot} = \hat{\mathbf{G}}_{(j)p\cdot}\mathbf{H} = \sum_{(p,q)\ in\ the\ same\ cluster} \mathbf{H}_{q\cdot}, \tag{25}$$

which only needs $O(NK)$ operations. In total, multiplications between $\mathbf{T}_{(i)} = \sum_{j=1,j\neq i}^{Z} \hat{\mathbf{G}}_{(j)}$ and $\mathbf{H}$ (or $\mathbf{X}$) requires $O(NKZ)$ operations, where Z is the number of facets. The procedure of MNE is outlined in Algorithm 1.

Complexity analysis Here, we give a comprehensive complexity analysis of MNE. For simplicity, we assume $K_i = K$ for $i \in \{1, 2, 3..., Z\}$. From Eqs. (21), (22), (23) and (24), we conclude that the majority of the computations consists of four parts in each iteration. 1) The multiplications between $\mathbf{M}$ and $\mathbf{H}$ (or $\mathbf{C}$). Since $\mathbf{M}$ is sparse, $O(MK)$ operations are demanded, where M and K are the number of nonzero elements in $\mathbf{M}$ and the number of embedding dimensions, respectively. 2) The computation of the inverse of $K \times K$ matrix requires $O(K^3)$ operations. 3) The additions and subtractions between two $N \times K$ matrices will cost $O(NK)$ operations. 4) The multiplications between $\mathbf{T}_i$ and $\mathbf{H}$ (or $\mathbf{X}$) need $O(NKZ)$ operations as previously analyzed. In summary, MNE requires a linear complexity of $O(MK + K^3 + NKZ)$ in each iteration.

Experiments

In this section, we empirically evaluate the performances of MNE via vertex classification. Besides, the parameters sensitivity is analyzed.

Datasets. The experiments are conducted on Facebook100 dataset (Traud, Mucha, and Porter 2012). Each network is an abstraction of the facebook social network for one university in U.S.A. in Sept. 2005. In addition to the network structure, the user metadata, which includes student/faculty status, gender, major, dormitory, year of enrollment and high school, is available. For conveniences, three properties, gender, year of enrollment and dormitory, are selected as the ground-truth to evaluate the proposed embedding algorithm. Although other metadata is also available, they are not considered due to the following reasons. 1)

Table 2: Classification results on four networks.

Datasets	UChicago									Temple								
Tasks	Gender			Grade			Dormitory			Gender			Grade			Dormitory		
Label %	1%	5%	9%	1%	5%	9%	1%	5%	9%	1%	5%	9%	1%	5%	9%	1%	5%	9%
DeepWalk	50.1	52.3	55.9	55.6	59.1	63.8	20.2	35.7	47.4	50.1	55.5	58.2	51.1	55.7	60.3	21.4	31.8	36.1
LINE	52.1	54.1	56.9	61.0	61.9	65.2	21.1	43.5	50.1	52.9	57.9	58.5	56.3	66.9	69.6	25.4	32.7	38.2
GraRep	47.7	48.5	50.1	50.5	55.3	59.9	18.6	30.3	40.0	45.6	49.0	55.0	50.3	57.2	65.1	21.7	29.6	31.5
node2vec	51.3	53.5	55.2	**60.2**	61.2	64.1	22.1	39.8	49.7	51.0	54.8	57.9	52.8	55.3	64.2	20.2	29.8	38.1
MNE	**54.5**	**57.7**	**59.7**	58.1	**65.9**	**67.7**	**24.8**	**48.2**	**54.4**	**55.9**	**61.4**	**62.9**	**61.5**	**69.9**	**72.7**	**30.1**	**36.1**	**41.9**
Datasets	Haverford									Mississippi								
Tasks	Gender			Grade			Dormitory			Gender			Grade			Dormitory		
Label %	1%	5%	9%	1%	5%	9%	1%	5%	9%	1%	5%	9%	1%	5%	9%	1%	5%	9%
DeepWalk	50.6	53.5	57.3	61.4	76.7	81.1	29.0	37.4	43.9	53.1	60.4	60.9	46.5	55.3	61.6	32.5	44.1	48.3
LINE	50.1	51.6	52.9	59.1	76.1	80.5	27.9	36.6	41.5	55.3	62.7	64.7	48.6	58.9	63.2	34.2	48.9	53.4
GraRep	48.8	51.1	51.9	57.4	72.1	77.5	29.0	39.8	42.9	44.6	48.0	52.9	42.7	48.3	49.2	32.5	45.9	52.1
node2vec	51.3	57.1	57.1	57.6	75.6	79.1	29.2	41.4	43.8	52.6	59.8	59.8	47.2	56.8	60.1	31.3	39.5	44.1
MNE	**54.2**	**59.6**	**62.0**	**66.9**	**81.3**	**84.4**	**33.0**	**45.7**	**47.6**	**58.9**	**65.9**	**68.0**	**53.3**	**59.4**	**63.8**	**38.7**	**53.7**	**56.7**

The data of the status flag is unbalanced, because most of the users are students, 2) High school and major are too scattered. For example, there are 30,147 users, which are from 3,909 different high schools, in Michigan University. Among these networks, Haverford (1,446 nodes and 59,589 edges), UChicago (6,591 nodes and 208,103 edges), Mississippi (10,521 nodes and 610,911 edges) and Temple (13,686 nodes and 360,795 edges) are chosen to demonstrate the quantitative results.

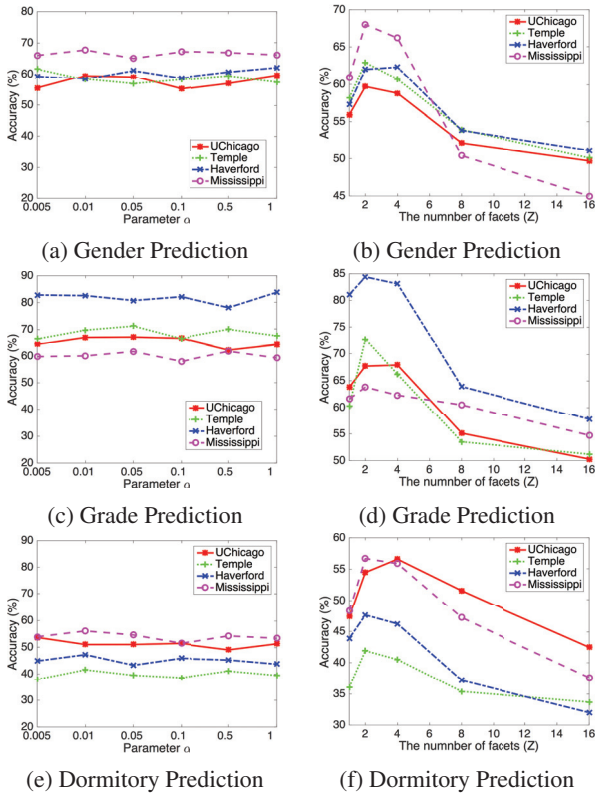
Baseline Methods. In the experiments, MNE is compared to four baseline methods, DeepWalk (Perozzi, Al-Rfou, and Skiena 2014), LINE (Tang et al. 2015), node2vec (Grover and Leskovec 2016) and GraRep (Cao, Lu, and Xu 2015). Each of them emploies their default settings in their original paper. For fair comparisons, the embedding dimension P is set to 128 for each of them.

MNE Settings. The hyperparameters of MNE are set as follows: balancing parameters $\lambda = 0.3, \alpha = 0.001$, number of facets $Z = 2$, number of embedding dimensions for each facets $K_i = 64$ for $i = 1, 2$ (the total dimension $K = Z * K_i = 128$) and proximity matrix $\mathbf{M} = \mathbf{Q}$.

Node Classification

To demonstrate the performances of MNE on multiple classification tasks (gender, year of enrollment and dormitory predictions), a portion of the users are randomly selected to be the training data, while the rest of users are employed as the testing data. The embeddings of users are normalized according to the L2-norm and fed into the SVM classifier which is implemented by Liblinear (Fan et al. 2008). This process is repeated 10 times, and the average results are reported in Table 2 with the highest accuracies hightlighted in bold. The results indicate that MNE outperforms the baselines by about 4%, which demonstrates the effectiveness of MNE for the multiple classification tasks.

Parameter Sensitivity. The effects of the hyperparameters are investigated here. There are three hyperparameters in MNE, the number of facets Z, the balancing parameters λ and α. By fixing the training rate to be 9%, the classification accuracies with respect to α are shown in Figures 1a, 1c and 1e, and the impacts of the Z are shown in Figure 1b, 1d and 1f. According to the experimental results, we can observe that MNE maintains a stable accuracy when the balancing parameter α varies within a certain range. Besides, the best performance of MNE can be achieved when 2 or 4 facets are employed.



(a) Gender Prediction (b) Gender Prediction

(c) Grade Prediction (d) Grade Prediction

(e) Dormitory Prediction (f) Dormitory Prediction

Figure 1: The impacts of α and Z on prediction accuracies.

Conclusions

In this paper, we analyze the different goals of community detection and network embedding, and thus observe that there exists a general solution, *"maximizing the consistency between similar nodes while maximizing the distance between the dissimilar nodes"*, for the current community detection and network embedding problems. Network embedding algorithms developed from the general solution usually perform the embeddings representing only one noticeable facet of the network and ignore necessity of diverse properties. The proposed Multi-facet Network Embedding captures the multiple facets of the network via multiple representation learning. MNE is regularized by an efficient Binary Hilbert Schmidt Independence Criterion to ensure the diversities between different facets. Extensive results demonstrate the necessity of multiple diverse embeddings and the superiority of the proposed method compared to the state-of-the-art ones.

Acknowledgments

This work was supported by National Key Research and Development Plan (No.2016YFC0801004), National Natural Science Foundation of China (No. 61503281, U1636214, 61332012, 61672514), Fundamental Theory and Cutting Edge Technology Research Program of Institute of Information Engineering, CAS (Grant No. Y7Z0381102) and Foundation for the Young Scholars by Tianjin University of Commerce (No. 150113),.

References

Bartels, R. H., and Stewart, G. W. 1972. Solution of the matrix equation ax+ xb= c [f4]. *Communications of the ACM* 15(9):820–826.

Bertsekas, D. P. 2014. *Constrained optimization and Lagrange multiplier methods*. Academic Press.

Cao, S.; Lu, W.; and Xu, Q. 2015. Grarep: Learning graph representations with global structural information. In *Proceedings of the 24th ACM International Conference on Information and Knowledge Management*, 891–900. Melbourne, VIC, Australia: ACM.

Fan, R.-E.; Chang, K.-W.; Hsieh, C.-J.; Wang, X.-R.; and Lin, C.-J. 2008. Liblinear: A library for large linear classification. *Journal of Machine Learning Research* 9(Aug):1871–1874.

Fortunato, S. 2010. Community detection in graphs. *Physics Reports* 486(3):75–174.

Girvan, M., and Newman, M. E. 2002. Community structure in social and biological networks. *Proceedings of the National Academy of Sciences* 99(12):7821–7826.

Gretton, A.; Bousquet, O.; Smola, A.; and Scholkopf, B. 2005. Measuring statistical dependence with hilbert-schmidt norms. In *Proceeding of 16th International Conference on Algorithmic Learning Theory*, volume 16, 63–78. Singapore: Springer.

Grover, A., and Leskovec, J. 2016. node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*, 855–864. San Francisco, CA, USA.: ACM.

Malliaros, F. D., and Vazirgiannis, M. 2013. Clustering and community detection in directed networks: A survey. *Physics Reports* 533(4):95–142.

Mikolov, T.; Sutskever, I.; Chen, K.; Corrado, G. S.; and Dean, J. 2013. Distributed representations of words and phrases and their compositionality. In *Proceeding of Advances in Neural Information Processing Systems*, 3111–3119.

Ou, M.; Cui, P.; Pei, J.; Zhang, Z.; and Zhu, W. 2016. Asymmetric transitivity preserving graph embedding. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*, 1105–1114. San Francisco, CA, USA.: ACM.

Perozzi, B.; Al-Rfou, R.; and Skiena, S. 2014. Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 701–710. New York, NY, USA.: ACM.

Psorakis, I.; Roberts, S.; Ebden, M.; and Sheldon, B. 2011. Overlapping community detection using bayesian non-negative matrix factorization. *Physical Review E* 83(6):066114.

Shi, J., and Malik, J. 2000. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 22(8):888–905.

Tang, J.; Qu, M.; Wang, M.; Zhang, M.; Yan, J.; and Mei, Q. 2015. Line: Large-scale information network embedding. In *Proceedings of the 24th International Conference on World Wide Web*, 1067–1077. Florence, Italy: ACM.

Tian, F.; Gao, B.; Cui, Q.; Chen, E.; and Liu, T.-Y. 2014. Learning deep representations for graph clustering. In *Proceedings of the Twenty-Eighth AAAI Conference on Artificial Intelligence*, 1293–1299. Québec City, Québec, Canada: AAAI Press.

Traud, A. L.; Mucha, P. J.; and Porter, M. A. 2012. Social structure of facebook networks. *Physica A: Statistical Mechanics and its Applications* 391(16):4165–4180.

Wang, D.; Cui, P.; and Zhu, W. 2016. Structural deep network embedding. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*, 1225–1234. San Francisco, CA, USA.: ACM.

Yang, Z.; Hao, T.; Dikmen, O.; Chen, X.; and Oja, E. 2012. Clustering by nonnegative matrix factorization using graph random walk. In *Proceeding of Advances in Neural Information Processing Systems*, 1079–1087.

Yang, C.; Liu, Z.; Zhao, D.; Sun, M.; and Chang, E. Y. 2015. Network representation learning with rich text information. In *Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence*, 2111–2117. Buenos Aires, Argentina: AAAI Press.