ComClus: A Self-Grouping Framework for Multi-Network Clustering

Jingchao Ni, Wei Cheng, Wei Fan, and Xiang Zhang

Abstract—Joint clustering of multiple networks has been shown to be more accurate than performing clustering on individual networks separately. This is because multi-network clustering algorithms typically assume there is a common clustering structure shared by all networks, and different networks can provide compatible and complementary information for uncovering this underlying clustering structure. However, this assumption is too strict to hold in many emerging applications, where multiple networks usually have diverse data distributions. More popularly, the networks in consideration belong to different underlying groups. Only networks in the same underlying group share similar clustering structures. Better clustering performance can be achieved by considering such groups differently. As a result, an ideal method should be able to automatically detect network groups so that networks in the same group share a common clustering structure. To address this problem, we propose a new method, ComClus, to simultaneously group and cluster multiple networks. ComClus is novel in combining the clustering approach of non-negative matrix factorization (NMF) and the feature subspace learning approach of metric learning. Specifically, it treats node clusters as features of networks and learns proper subspaces from such features to differentiate different network groups. During the learning process, the two procedures of network grouping and clustering are coupled and mutually enhanced. Moreover, ComClus can effectively leverage prior knowledge on how to group networks such that network grouping can be conducted in a semi-supervised manner. This will enable users to guide the grouping process using domain knowledge so that network clustering accuracy can be further boosted. Extensive experimental evaluations on a variety of synthetic and real datasets demonstrate the effectiveness and scalability of the proposed method.

Index Terms—Multi-network clustering, network grouping, non-negative matrix factorization

1 Introduction

Network (or graph) clustering is a fundamental problem to discover closely related objects in a network. In many emerging applications, multiple networks are generated from different conditions or domains, such as gene co-expression networks collected from different tissues of model organisms [1], co-author networks built in different academic conferences [2], social networks generated at different time points [3], etc. These applications drive the recent research interests to joint clustering of multiple networks, which has been shown to significantly improve the clustering accuracy over single network clustering methods [4].

The key superiority of multi-network clustering methods is to leverage the shared clustering structure across all networks, since a consensus clustering structure is more robust to the incompleteness and noise in individual networks. For example, multi-view network clustering methods [4], [5], [6] work on multiple representations (views) of the same set of data objects. Different views can provide complementary information on the underlying data distribution. Multi-domain network clustering methods [7], [8] integrate networks of different sets of data objects, and uses the mappings between objects in different networks to penalize inconsistent clusterings.

To be successful, the existing multi-network clustering methods typically assume different networks share a consensus clustering structure. This very basic assumption, however, is too simplified to real-world applications. Consider an important bioinformatics problem, the gene co-expression network clustering [1], [9]. In a gene co-expression network, each node is a gene and an edge represents the functional association between two connected genes. To enhance performance, we can use multiple gene co-expression networks collected in multiple tissues. However, genes have tissue-specific roles and often form tissue-specific interactions [10], [11]. The same set of genes may form a cluster (e.g., a functional module or gene pathway) in similar tissues but not in others. Thus when we use such networks collected in different tissues, we cannot simply assume they form similar clustering structures.

In this paper, we study a novel and generalized problem where we cannot simply assume the given networks share a consensus clustering structure. Consider the six networks in Fig. 1, they may represent gene co-expression networks from different tissues. Clearly, they do not share a common clustering structure. For example, nodes $\{1, 2, 3\}$ form a cluster in network A but are in three different clusters in network C. However, by a visual inspection, we can partition

- *J. Ni and X. Zhang are with the College of Information Sciences and Technology, Pennsylvania State University, State College, PA 16802. E-mail: {jzn47, xzhang}@ist.psu.edu.*
- *W. Cheng is with NEC Laboratories America, Princeton, NJ 08540. E-mail: weicheng@nec-labs.com.*
- *W. Fan is with the Tencent Medical AI Lab, Palo Alto, CA 94301. E-mail: davidwfan@tencent.com.*

Manuscript received 31 Oct. 2016; revised 5 Aug. 2017; accepted 21 Oct. 2017. Date of publication 9 Nov. 2017; date of current version 2 Feb. 2018. (Corresponding author: Jingchao Ni.)
Recommended for acceptance by C. Domeniconi.
For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org, and reference the Digital Object Identifier below.
Digital Object Identifier no. 10.1109/TKDE.2017.2771762

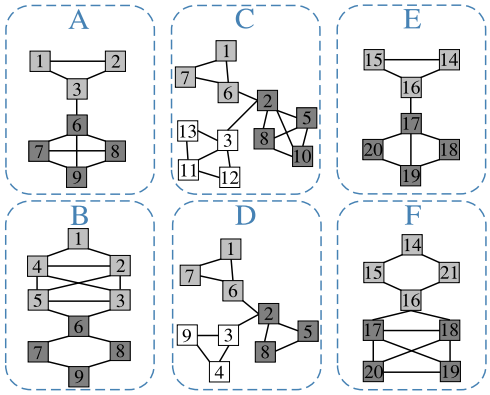


Fig. 1. An example of six networks. These networks can be grouped into {*A*, *B*}, {*C*, *D*}, and {*E*, *F*} according to their clustering structures.

these networks into three groups, i.e., {*A*, *B*}, {*C*, *D*} and {*E*, *F*}, since {*A*, *B*} share an underlying clustering structure, and so do {*C*, *D*} and {*E*, *F*}. This is practically reasonable. For example, a set of similar tissues can share many similar gene clusters so that similar functions can be supplied. As another example, consider the co-author networks of different research areas [2]. Similar areas usually attract many similar author clusters (e.g., research sub-communities). Therefore, an ideal method to cluster this collection of networks should be able to (1) automatically detect network groups such that networks in the same group share a common clustering structure, and (2) enhance clustering accuracy by leveraging group-wise consensus structures.

However, real-life networks are often diverse, noisy and incomplete. Even a group of similar networks may not have exactly the same clustering structure. Instead, they may only share a subset of their clusters and the shared clusters may only partially match. In Fig. 1, networks {*C*, *D*} only share two clusters {*1*, *7*, *6*} and {*2*, *5*, *8*}, and other nodes are irrelevant. Therefore, to effectively group together some networks, an ideal method should identify a subset of clusters that are common among these networks. This is a novel and non-trivial challenge. The existing multi-network clustering methods either assume all clusters are common [4], [5], [6] or simply enhance common clusters without identifying them [1], [7], [8] thus cannot tackle this problem.

In this paper, we propose a novel method ComClus to address these challenges. Briefly, ComClus is novel in treating node clusters as features of networks and grouping together networks sharing the same feature subspace (i.e., a common subset of clusters). This is inspired by the field of subspace clustering [12], which aim to cluster high-dimensional vector data such that each cluster is associated with a relevant feature subspace. In ComClus, network grouping and common cluster detection can mutually reinforce each other. Correctly grouping networks sharing a common clustering structure can resolve ambiguity hence refine detected common clusters. Correctly detecting common clusters can reduce the possibility that a network goes to a wrong group.

Furthermore, when prior knowledge on how to group networks is available, ComClus can effectively leverage it and further boost network clustering accuracy. This is especially desirable when users have domain knowledge that can guide the grouping procedure. One common knowledge form can be the constraints saying which networks should be grouped together and which networks should be in different groups. Another form can be the similarities between different networks indicating how similar each pair of networks are. In this paper, we consider both forms of prior knowledge and extend ComClus to a semi-supervised setting in a principled manner.

In summary, the contributions of this paper are listed as follows.

- We study a novel multi-network clustering problem, where the goal is to enhance clustering accuracy by automatically grouping networks sharing a common clustering structure. This is different from some existing community detection approaches [13], [14], which aim to identify clusters that occur consistently in multiple networks, but not to improve the clustering accuracy (See Section 2 for a detailed discussion).
- We propose a new method ComClus to address the novel challenges. ComClus is novel in employing the idea of feature subspace learning for joint multi-network grouping and clustering.
- We extend ComClus to a semi-supervised setting such that the prior knowledge on network grouping can be effectively incorporated to further enhance network clustering accuracy.
- We develop efficient optimization algorithms to both of the unsupervised and semi-supervised problems. A solid theoretical analysis on the algorithmic convergence and complexity is also provided.
- We perform extensive experiments on synthetic datasets as well as a variety of real-life datasets, including newsgroup dataset, social networks, collaboration networks and biological interaction networks. The results demonstrate the effectiveness and scalability of the proposed method.

The rest of the paper is organized as follows. Section 2 summarizes the related work. Section 3 describes the problem. Section 4 presents the ComClus algorithm and its theoretical analysis. Section 6 details how to leverage prior knowledge. Section 7 presents the experimental evaluations. Section 8 gives the concluding remarks.

2 Related Work

There are several bodies of approaches developed for multi-network clustering. Traditional multi-view clustering is among the most popular ones [4], [5], [6]. In these methods, views can be either networks or data-feature matrices about the same set of objects. Ensemble clustering [15], [16] is highly related to multi-view clustering, but it does not jointly cluster multiple data views, but aims to find an agreement of individual clustering results. More recently, methods on multi-domain network clustering [7], [8] were proposed to integrate networks about different sets of objects by cross-network object mapping relationships. All these methods are based on a simplified assumption that multiple views or networks share a single consensus clustering structure.

Multiple networks can also be represented by the tensor model. However, existing tensor decomposition methods, such as CP and Tucker decompositions [17], are good for co-clustering multiple matrices [18] but are not designed for network data where two modes of the tensor are symmetric.

Moreover, tensor decomposition also limits all networks to share a single common underlying clustering structure.

Evolutionary clustering approaches [3], [19] work only on temporally ordered multi-network. These methods assume two consecutive networks have consistent clustering structures but the clustering structures can gradually vary in a long-term. The goal of such methods is to enhance clustering accuracy but they cannot be applied on general multi-network, nor to solve our problem.

Some methods are developed to detect communities in multi-layer networks [13], [14], [20], [21], [22], where each layer is a distinct network. All layers are about the same set of objects but have different topologies. These methods aim to identify communities that are consistent in some layers, not to enhance clustering accuracy by using consensus and common clustering structure. Thus they have a different goal from us (as well as the approaches mentioned above) and cannot be applied to solve our problem.

In [1], a method NoNClus is developed to cluster multiple networks with multiple underlying clustering structures. The problem studied in [1] is markedly different from ours. In [1], the problem is to enhance clustering accuracy by using the network group information that is already known. In practice, however, such network group information may not be available beforehand. Thus NoNClus can neither identify common clusters among networks nor group networks automatically through their different clustering structures.

Note that our work is different from subspace clustering [12], which is motivated by the observation that, in high-dimensional vector data, irrelevant features may obfuscate clustering structure when using full-space clustering methods. Its goal is to discover meaningful clusters associated with feature subspace projections. It is also worth to mention the so-called multi-view subspace clustering [23]. It is highly related to subspace clustering but not the multi-view clustering mentioned above. Its goal is to find multiple clustering results of the same vector-space data such that different results are associated with different subspaces of the data features. A clustering result is a "view" of the data. All these methods neither study data represented by multiple views as mentioned above nor network data.

3 The Problem

Let $\mathcal{Y} = \{\mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(g)}\}$ be the g given *member networks*. Each network is represented by its adjacency matrix $\mathbf{A}^{(i)} \in \mathbb{R}_+^{n_i \times n_i}$, where n_i is the number of nodes in $\mathbf{A}^{(i)}$. Let $\mathcal{V}^{(i)}$ represents the set of nodes in $\mathbf{A}^{(i)}$, and $\mathcal{I}^{(ij)}$ represents the set of common nodes between $\mathbf{A}^{(i)}$ and $\mathbf{A}^{(j)}$, i.e., $\mathcal{I}^{(ij)} = \mathcal{V}^{(i)} \cap \mathcal{V}^{(j)}$. In Fig. 1, the common nodes between member networks *A* and *C* are {*1*, *2*, *3*, *6*, *7*, *8*}.

A *network group* $\mathcal{Y}^{(p)}$ is a subset of $\mathcal{Y}$ such that networks in $\mathcal{Y}^{(p)}$ share a common underlying clustering structure. In this paper, we consider each network to belong to one group. That is, if there are k network groups, then $\cup_{p=1}^{k} \mathcal{Y}^{(p)} = \mathcal{Y}$, and for any $p \neq q$, $\mathcal{Y}^{(p)} \cap \mathcal{Y}^{(q)} = \emptyset$. A more general scenario could allow each network to belong to more than one groups, in which analyzing shared common clustering structure is more complicated. As an initial work considering network grouping, we focus on the first case and leave the general case as a future study. In Fig. 1, the six networks can be grouped as {*A*, *B*}, {*C*, *D*} and {*E*, *F*}.

TABLE 1
Summary of Symbols

Symbol	Description
$\mathbf{A}^{(i)}$	the ith member network
$\mathbf{V}$	the network grouping indicator matrix
$\mathbf{U}$	the global latent factor matrix of all nodes
$\mathbf{s}^{(j)}$	the centroid vector of network group j
$\mathbf{S}$	$\mathbf{S} = [\mathbf{s}^{(1)}, \ldots, \mathbf{s}^{(k)}]$ is the common cluster indicator matrix
$\mathbf{w}^{(i)}$	the cluster-level feature vector of $\mathbf{A}^{(i)}$. $\mathbf{w}_p^{(i)}$ indicates how likely the pth latent cluster will appear in $\mathbf{A}^{(i)}$.
$\mathbf{D}_W^{(i)}$	a diagonal matrix with diagonal values being $\mathbf{w}^{(i)}$, i.e., $\mathbf{D}_W^{(i)} = \mathrm{diag}(\mathbf{w}^{(i)})$
$\mathbf{W}$	a stacked matrix $\mathbf{W} = [\mathbf{w}^{(1)}, \ldots, \mathbf{w}^{(g)}]$
$\mathbf{O}^{(i)}$	a mapping matrix between $\mathbf{A}^{(i)}$ and $\mathbf{U}$. $\mathbf{O}^{(i)}(x, y) = 1$ means the xth row of $\mathbf{A}^{(i)}$ and the yth row of $\mathbf{U}$ represent the same node; $\mathbf{O}^{(i)}(x, y) = 0$ otherwise.
g	the number of member networks
n_i	the number of nodes in $\mathbf{A}^{(i)}$
k	the number of network groups
h	the number of latent dimensions in $\mathbf{U}$
$\mathcal{Y}$	the member networks $\mathcal{Y} = \{\mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(g)}\}$
$\mathcal{Y}^{(p)}$	the pth network group $\mathcal{Y}^{(p)} \subset \mathcal{Y}$
$\mathcal{V}^{(i)}$	the set of nodes in $\mathbf{A}^{(i)}$
$\mathcal{I}^{(ij)}$	the common node set $\mathcal{I}^{(ij)} = \mathcal{V}^{(i)} \cap \mathcal{V}^{(j)}$

The member networks in the same group share a set of *common clusters*. In this paper, we consider any two clusters from two different networks as common clusters if they are about the same subset of nodes and they form dense clusters in their respective networks. These clusters are used as features to characterize each network group and distinguish one group from another. In Fig. 1, the common clusters of network group {*C*, *D*} are clusters {*1*, *7*, *6*} and {*2*, *5*, *8*}.

Our goal is to simultaneously group and cluster the given member networks $\{\mathbf{A}^{(i)}\}_{i=1}^{g}$, such that (1) the member networks are partitioned into k groups with each group sharing a common set of clusters; and (2) the common clusters in each network group are identified and their accuracies are enhanced. Note that we focus on finding non-overlapping clusters, which is also the common setting of the existing multi-view (domain) network clustering methods [1], [4], [5], [6], [7], [8]. Table 1 summarizes the important symbols used in this paper.

4 The ComClus Algorithm

In this section, we introduce ComClus, a novel method that integrates NMF and metric learning [24] for jointly grouping and clustering multiple networks.

4.1 Preliminaries

Non-negative matrix factorization (NMF) [25] is widely used for clustering. We adopt the symmetric version of NMF (SNMF) [26] as the basic approach for clustering a single network, which minimizes the following objective function

$$\mathcal{L}_G(\mathbf{H}) = \sum_{i,j=1}^{l} (\mathbf{B}_{ij} - \mathbf{z}_{i*}\mathbf{z}_{j*}^T)^2 = \|\mathbf{B} - \mathbf{Z}\mathbf{Z}^T\|_F^2, \quad (1)$$

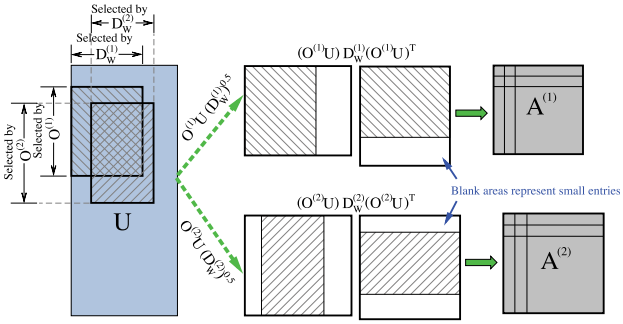


Fig. 2. An illustration of the subspace-based SNMF for two networks. $\mathbf{O}^{(i)}$ $(i = 1, 2)$ is a binary mapping matrix to align nodes between $\mathbf{A}^{(i)}$ and $\mathbf{U}$. $\mathbf{D}_W^{(i)}$ is a diagonal matrix to select important columns (i.e., clusters) for $\mathbf{A}^{(i)}$. We can consider $\mathbf{O}^{(i)}\mathbf{U}(\mathbf{D}_W^{(i)})^{\frac{1}{2}}$ as the sub-block selected for approximating $\mathbf{A}^{(i)}$. Here, shaded areas represent columns with large entries, as selected by $\mathbf{D}_W^{(i)}$.

where $\|\cdot\|_F$ is the Frobenius norm, $\mathbf{z}_{i*} \in \mathbb{R}_+^{1\times r}$ is a r-dimensional latent vector of node i, and $\mathbf{Z} = [\mathbf{z}_{1*}^T, \ldots, \mathbf{z}_{l*}^T]^T$ is the factor matrix of $\mathbf{B}$. An entry $\mathbf{Z}_{ij}$ indicates to which degree the node i belongs to the cluster j.

4.2 Clusters as Network Features

Next, we develop a *subspace SNMF* method to learn the set of clusters that can be used as features to characterize each member network in $\{\mathbf{A}^{(i)}\}_{i=1}^g$. Let $\mathcal{V} = \cup_{i=1}^g \mathcal{V}^{(i)}$ be the global set of nodes in all member networks. In the SNMF, i.e., Eq. (1), each node i is represented in an r-dimensional latent space by $\mathbf{z}_{i*}$ for a single network. Given multiple networks $\{\mathbf{A}^{(i)}\}_{i=1}^g$, we aggregate their latent spaces into a single *global h-dimensional latent space*, where h is the number of latent dimensions. Then for each node x in $\mathcal{V}$, it is represented by a *global latent vector* $\mathbf{u}_{x*} \in \mathbb{R}_+^{1\times h}$.

In Eq. (1), each entry $\mathbf{B}_{ij}$ is approximated by the inner product between $\mathbf{z}_{i*}$ and $\mathbf{z}_{j*}$, where the full spaces of the r-dimensional latent vectors $\mathbf{z}_{i*}$ and $\mathbf{z}_{j*}$ are used for approximation. Since each dimension of $\mathbf{z}_{i*}$ represents a cluster, all clusters in network $\mathbf{B}$ are used for the approximation. In our method, to learn which subset of clusters appearing in a network $\mathbf{A}^{(i)}$, when approximating an entry $\mathbf{A}_{xy}^{(i)}$, we only use a subspace (i.e., a subset of dimensions/clusters) of the global h-dimensional latent vectors $\mathbf{u}_{x*}$ and $\mathbf{u}_{y*}$.

Specifically, for each network $\mathbf{A}^{(i)}$, we define a metric vector $\mathbf{w}^{(i)} \in \mathbb{R}_+^{h\times 1}$ whose entry $\mathbf{w}_p^{(i)}$ indicates the importance of the global latent dimension p to network $\mathbf{A}^{(i)}$. Therefore, when we approximate an entry $\mathbf{A}_{xy}^{(i)}$, we use $\mathbf{u}_{x*}\mathrm{diag}(\mathbf{w}^{(i)})\mathbf{u}_{y*}^T$, where $\mathrm{diag}(\mathbf{w}^{(i)})$ is a diagonal matrix with the diagonal vector as $\mathbf{w}^{(i)}$. Let $\mathbf{D}_W^{(i)} = \mathrm{diag}(\mathbf{w}^{(i)})$, using square loss function, we can collectively approximate $\mathbf{A}^{(i)}$ by minimizing

$$\sum_{x,y=1}^{n_i} (\mathbf{A}_{xy}^{(i)} - \mathbf{u}_{x*}\mathbf{D}_W^{(i)}\mathbf{u}_{y*}^T)^2, \tag{2}$$

where $n_i = |\mathcal{V}^{(i)}|$ is the number of nodes in $\mathbf{A}^{(i)}$.

In general, different networks can have different node sets $\mathcal{V}^{(i)}$, thus having different sizes. Let $n = |\mathcal{V}|$. We define for each network $\mathbf{A}^{(i)}$ a mapping matrix $\mathbf{O}^{(i)} \in \{0, 1\}^{n_i\times n}$ such that $\mathbf{O}^{(i)}(x, y) = 1$ if node x in $\mathcal{V}^{(i)}$ and node y in $\mathcal{V}$ represent the same object. Let $\mathbf{U} = [\mathbf{u}_{1*}^T, \ldots, \mathbf{u}_{n*}^T]^T \in \mathbb{R}_+^{n\times h}$ be a *global latent factor matrix*, then we can obtain the matrix form of Eq. (2) as following:

$$\|\mathbf{A}^{(i)} - (\mathbf{O}^{(i)}\mathbf{U})\mathbf{D}_W^{(i)}(\mathbf{O}^{(i)}\mathbf{U})^T\|_F^2. \tag{3}$$

Then for all member networks, we have a loss function as

$$\mathcal{L}_A(\mathbf{U}, \{\mathbf{D}_W^{(i)}\}_{i=1}^g) = \sum_{i=1}^g \|\mathbf{A}^{(i)} - (\mathbf{O}^{(i)}\mathbf{U})\mathbf{D}_W^{(i)}(\mathbf{O}^{(i)}\mathbf{U})^T\|_F^2. \tag{4}$$

In Eq. (4), all member networks share the same latent factor matrix $\mathbf{U}$. When approximating $\mathbf{A}^{(i)}$, a sub-block of $\mathbf{U}$ is used. Fig. 2 illustrates the idea. In this process, $\mathbf{O}^{(i)}$ selects the rows of $\mathbf{U}$ for $\mathbf{A}^{(i)}$, which corresponds to the selection of node set $\mathcal{V}^{(i)}$ from $\mathcal{V}$. $\mathbf{D}_W^{(i)}$ selects the columns of $\mathbf{U}$ for $\mathbf{A}^{(i)}$, which corresponds to the selection of latent subspace (i.e., clusters). Then, we can consider $\mathbf{O}^{(i)}\mathbf{U}(\mathbf{D}_W^{(i)})^{\frac{1}{2}}$ as the sub-block selected for approximating $\mathbf{A}^{(i)}$. In Fig. 2, shaded areas of the sub-blocks represent columns with large entries, as selected by $\mathbf{D}_W^{(i)}$. Therefore, if two networks $\mathbf{A}^{(i)}$ and $\mathbf{A}^{(j)}$ share many nodes, they will have large overlap in the rows of $\mathbf{U}$. If $\mathbf{A}^{(i)}$ and $\mathbf{A}^{(j)}$ further show similar clustering structures, it is highly possible that they will share similar columns in $\mathbf{U}$, i.e., similar $\mathbf{D}_W^{(i)}$ and $\mathbf{D}_W^{(j)}$. This is because using similar sub-blocks of $\mathbf{U}$ would achieve good approximations for both $\mathbf{A}^{(i)}$ and $\mathbf{A}^{(j)}$ at this time. On the other hand, if $\mathbf{A}^{(i)}$ and $\mathbf{A}^{(j)}$ have dissimilar clustering structures, using similar subspaces of $\mathbf{U}$ (i.e., similar sub-blocks of $\mathbf{U}$) to approximate both $\mathbf{A}^{(i)}$ and $\mathbf{A}^{(j)}$ will result in large loss function value. By minimizing $\mathcal{L}_A$ in Eq. (4), $\mathbf{D}_W^{(i)}$ and $\mathbf{D}_W^{(j)}$ then tend to lie in separate subspaces of $\mathbf{U}$.

In Eq. (1), the latent dimensions (i.e., columns) of $\mathbf{Z}$ represent clusters of nodes. Thus in our subspace SNMF, the columns of $\mathbf{U}$ represent latent clusters. Each $\mathbf{w}^{(i)}$ (recall $\mathbf{D}_W^{(i)} = \mathrm{diag}(\mathbf{w}^{(i)})$) is a *cluster-level feature vector* where an entry $\mathbf{w}_p^{(i)}$ indicates the selection the pth latent cluster for network $\mathbf{A}^{(i)}$. Therefore, $\mathbf{w}^{(i)}$ carries the clustering structure information of network $\mathbf{A}^{(i)}$ and can be used as a feature for network grouping.

Discussion. The existing NMF or spectral based multi-network/view clustering methods usually either define a latent factor matrix for each network and require all factor matrices to be consistent [1], [4], [6], [7], or use the same factor matrix for all views [5]. In our subspace SNMF, however, different networks use different sub-blocks of a global latent factor matrix $\mathbf{U}$. The sub-blocks can be (partially) shared by different networks, depending on whether the networks share common clusters. Thus our method is different from the existing ones by automatically determining whether networks share common clusters. If they do, common clusters can be automatically identified (by $\{\mathbf{D}_W^{(i)}\}_{i=1}^g$) for grouping purpose and their accuracies are then enhanced.

Also note that subspace SNMF in Eq. (4) is different from the symmetric non-negative matrix tri-factorization (SNMTF) [26]. SNMTF is used for single network clustering and its middle factor matrix has no constraints. In contrast, subspace SNMF is designed for multi-network clustering and $\{\mathbf{D}_W^{(i)}\}_{i=1}^g$ are diagonal and non-negative. Thus each $\mathbf{D}_W^{(i)}$ is positive semi-definite with diagonal entries representing weights for dimension selection, similar to a metric matrix in metric learning [24]. In this view, subspace SNMF is a

novel method combining SNMF and metric learning approaches, as desired for solving the problem of learning clusters as network features.

4.3 Regularization on Network Node Sets

In this section, we develop a regularizer to encode the similarity between node sets of different networks. The intuition is based on the following observation. Let us consider a special case when two networks $\mathbf{A}^{(1)}$ and $\mathbf{A}^{(2)}$ share few or no nodes, which makes $\mathbf{O}^{(1)}$ and $\mathbf{O}^{(2)}$ different. Their selected sub-blocks from $\mathbf{U}$ will have few overlap and be separated vertically. Using the example in Fig. 2, in this case, the lower sub-block may lie vertically below the higher one. At this time, $\mathbf{A}^{(1)}$ and $\mathbf{A}^{(2)}$ can be well approximated by the two different sub-blocks of $\mathbf{U}$ no matter $\mathbf{w}^{(1)}$ and $\mathbf{w}^{(2)}$ are similar or not. Thus it is likely that $\mathbf{w}^{(1)}$ and $\mathbf{w}^{(2)}$ are similar while $\mathbf{O}^{(1)}$ and $\mathbf{O}^{(2)}$ are different. However, this is counterintuitive since two networks having few common nodes should be considered dissimilar because they are about different relationships of different node entities. Thus we expect them to have dissimilar structural feature vectors $\mathbf{w}^{(1)}$ and $\mathbf{w}^{(2)}$. To address this issue, we employ the regularization on the network node set similarity. The details are shown in the following.

First, we measure the similarity between $\mathbf{w}^{(i)}$ and $\mathbf{w}^{(j)}$ by their inner product $(\mathbf{w}^{(i)})^T\mathbf{w}^{(j)}$. To penalize the similarity when $\mathbf{A}^{(i)}$ and $\mathbf{A}^{(j)}$ share few nodes, we propose the following penalty function

$$\mathcal{L}_\Phi(\{\mathbf{w}^{(i)}\}_{i=1}^g) = \sum_{i,j=1}^{g} \mathbf{\Phi}_{ij}(\mathbf{w}^{(i)})^T\mathbf{w}^{(j)}, \tag{5}$$

where $\mathbf{\Phi}_{i,j}$ is the penalty strength on $(\mathbf{w}^{(i)})^T\mathbf{w}^{(j)}$. A proper $\mathbf{\Phi}_{i,j}$ should have a high value when $|\mathcal{I}^{(ij)}|$ is small and a low value when $|\mathcal{I}^{(ij)}|$ is large. Thus we use a logistic function[1] as following to measure the penalty strength

$$\mathbf{\Phi}_{ij} = \begin{cases} \frac{1}{1+e^{-\lambda+2\lambda\text{Jaccard}(\mathcal{V}^{(i)},\mathcal{V}^{(j)})}} & i \neq j \\ 0 & i = j, \end{cases} \tag{6}$$

where $\text{Jaccard}(\mathcal{V}^{(i)}, \mathcal{V}^{(j)}) = \frac{|\mathcal{V}^{(i)} \cap \mathcal{V}^{(j)}|}{|\mathcal{V}^{(i)} \cup \mathcal{V}^{(j)}|}$, λ ($\lambda > 0$) is a parameter that can be used to control the range of $\mathbf{\Phi}_{ij}$. From Eq. (6), the largest possible range of $\mathbf{\Phi}_{ij}$ is $[0, 1]$. To make the value of $\mathbf{\Phi}_{ij}$ has sufficiently large varying space, we can choose λ to make the range of $\mathbf{\Phi}_{ij}$ approximate $[0, 1]$. For example, we set $\lambda = \log(999)$, then $\mathbf{\Phi}_{ij} \in [10^{-3}, 1 - 10^{-3}]$. There are other reasonable λ values, but they have minor impacts, since they do not change the most important role of $\mathbf{\Phi}_{ij}$, i.e., penalty strength, as defined in Eq. (6).

Let $\mathbf{W} = [\mathbf{w}^{(1)}, \mathbf{w}^{(2)}, \ldots, \mathbf{w}^{(g)}] \in \mathbb{R}_+^{h\times g}$ and $\mathbf{\Phi} \in \mathbb{R}_+^{g\times g}$ whose (i, j)th entry is $\mathbf{\Phi}_{ij}$. Then we can rewrite Eq. (5) by

$$\mathcal{L}_\Phi(\{\mathbf{w}^{(i)}\}_{i=1}^g) = \mathcal{L}_\Phi(\mathbf{W}) = \|\mathbf{\Phi} \circ (\mathbf{W}^T\mathbf{W})\|_1, \tag{7}$$

where $\circ$ is the entry-wise product, and $\|\cdot\|_1$ is the ℓ_1 norm.

1. Other functions can also be used. We choose logistic function because of the easy control of its range and shape.

4.4 Network Grouping

In order to assign member networks into k groups while detecting common clusters within each network group, we define k centroid vectors $\{\mathbf{s}^{(j)}\}_{j=1}^k$, where $\mathbf{s}^{(j)} \in \mathbb{R}_+^{h\times 1}$ ($1 \leq j \leq k$). The intuition is to enforce member networks in the same group to share the same centroid vector. That is, if network $\mathbf{A}^{(i)}$ belongs to group $\mathcal{Y}^{(j)}$, we want to minimize the difference $\|\mathbf{w}^{(i)} - \mathbf{s}^{(j)}\|_F^2$. Therefore, $\mathbf{s}^{(j)}$ represents the consistent cluster feature subspace of member networks in group $\mathcal{Y}^{(j)}$ and large entries in $\mathbf{s}^{(j)}$ indicate the shared latent dimensions, i.e., common clusters, in network group $\mathcal{Y}^{(j)}$.

Let $\mathbf{v}_{i*} \in \{0, 1\}^{1\times k}$ be the group membership vector of $\mathbf{A}^{(i)}$, i.e., $\mathbf{v}_{ij} = 1$ if $\mathbf{A}^{(i)} \in \mathcal{Y}^{(j)}$ and $\mathbf{v}_{ij} = 0$ otherwise. Also denote $\mathbf{S} = [\mathbf{s}^{(1)}, \ldots, \mathbf{s}^{(k)}]$. Then we can collectively minimize the difference between the cluster feature vectors and the centroid vectors by minimizing

$$\mathcal{L}_R(\mathbf{S}, \{\mathbf{v}_{i*}\}_{i=1}^g, \{\mathbf{w}^{(i)}\}_{i=1}^g) = \sum_{i=1}^{g} \|\mathbf{w}^{(i)} - \mathbf{S}\mathbf{v}_{i*}^T\|_F^2. \tag{8}$$

Equivalently, let $\mathbf{V} = [\mathbf{v}_{1*}^T, \ldots, \mathbf{v}_{g*}^T]^T$, we have

$$\mathcal{L}_R(\mathbf{S}, \mathbf{V}, \mathbf{W}) = \|\mathbf{W} - \mathbf{S}\mathbf{V}^T\|_F^2. \tag{9}$$

Eq. (9) can be explained as a co-clustering of $\mathbf{W}$ by NMF [25], [27]. Thus we can relax the $\{0, 1\}$ constraint on $\mathbf{V}$ such that $\mathbf{V} \in \mathbb{R}_+^{g\times k}$ to avoid the mixed integer programming [28], which is difficult to solve. Then $\mathbf{V}_{ij}$ indicates to which degree network $\mathbf{A}^{(i)}$ belongs to network group $\mathcal{Y}^{(j)}$.

4.5 The Unified Model

Combining the loss function of subspace SNMF in Eq. (4), the penalty function in Eq. (7) and the loss function of network grouping in Eq. (9), we obtain a unified objective function for joint multi-network grouping and clustering

$$\begin{aligned}\mathcal{L}(\mathbf{U}, \mathbf{V}, \mathbf{S}, \mathbf{W}) = \mathcal{L}_A(\mathbf{U}, \{\mathbf{D}_W^{(i)}\}_{i=1}^g) + \alpha\mathcal{L}_\Phi(\mathbf{W}) \\ + \beta\mathcal{L}_R(\mathbf{S}, \mathbf{V}, \mathbf{W}),\end{aligned} \tag{10}$$

where α and β are two parameters controlling the importances of the penalty function and network grouping, respectively. Note that $\mathbf{W}$ and $\{\mathbf{D}_W^{(i)}\}_{i=1}^g$ are two different representations of the same variables, we keep both of them in our algorithm.

Formally, we forlumate a joint optimization problem as

$$\begin{aligned}\min_{\mathbf{U},\mathbf{V},\mathbf{S},\mathbf{W}} \mathcal{L}(\mathbf{U}, \mathbf{V}, \mathbf{S}, \mathbf{W}) + \rho(\|\mathbf{V}\|_1 + \|\mathbf{U}\|_1 + \|\mathbf{S}\|_1) \\ \text{s.t. } \mathbf{U} \geq 0,\ \mathbf{V} \geq 0,\ \mathbf{S} \geq 0, \\ \mathbf{W} \geq 0,\ \mathbf{D}_W^{(i)} = \text{diag}(\mathbf{w}^{(i)}),\ \forall 1 \leq i \leq g.\end{aligned} \tag{11}$$

In Eq. (11), we add ℓ_1 norms on $\mathbf{U}$, $\mathbf{V}$ and $\mathbf{S}$ to provide the option on sparseness constraints. This is useful since each node (member network) usually belongs to a small number of clusters (network groups) [27], making $\mathbf{U}$ ($\mathbf{V}$) sparse, and the networks in each network group usually do not have many clusters in common, making $\mathbf{S}$ sparse. Here, ρ is a parameter controlling the sparseness. Typically, a larger ρ enforces more entries in $\{\mathbf{U}, \mathbf{V}, \mathbf{S}\}$ to approach zero values. Moreover, ρ provides the stability of our learning algorithm in Eqs. (12), (13), and (14), as will be seen in Section 5.1.

Example. Consider the example in Fig. 1. By minimizing Eq. (11), we learn $\mathbf{W}$ and $\mathbf{S}$ as the following:

$$\mathbf{W} = \begin{bmatrix} 0 & 0 & \mathbf{.04} & \mathbf{.05} & .01 & 0 \\ 0 & 0 & \mathbf{.02} & \mathbf{.02} & 0 & .02 \\ 0 & 0 & \mathbf{.06} & \mathbf{.06} & 0 & 0 \\ 0 & 0 & 0 & 0 & \mathbf{.14} & \mathbf{.09} \\ \mathbf{.27} & \mathbf{.24} & 0 & 0 & 0 & 0 \\ \mathbf{.21} & \mathbf{.12} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \mathbf{.18} & \mathbf{.11} \end{bmatrix}, \quad \mathbf{S} = \begin{bmatrix} 0 & \mathbf{.17} & .01 \\ 0 & \mathbf{.06} & .02 \\ 0 & \mathbf{.21} & 0 \\ 0 & 0 & \mathbf{.33} \\ \mathbf{.76} & 0 & 0 \\ \mathbf{.51} & 0 & 0 \\ 0 & 0 & \mathbf{.41} \end{bmatrix}.$$

In $\mathbf{W}$, each column is a h-dimensional cluster-level feature of a network (ordered from *A* to *F*), where $h = 7$. For {*A*, *B*}, $\mathbf{w}_A$ and $\mathbf{w}_B$ share two latent dimensions representing the two shared clusters. Similarly, {*C*, *D*} and {*E*, *F*} share three and two clusters, respectively. Moreover, networks in different groups have almost orthogonal features since they have different underlying clustering structures. Each column of $\mathbf{S}$ is a centroid vector of a network group. As discussed in Section 4.4, large entries in $\mathbf{S}$ indicate common clusters. For example, in the results, the largest two entries in the second column of $\mathbf{S}$, i.e., 0.17 and 0.21, correspond to the clusters {*1*, *7*, *6*} and {*2*, *5*, *8*}, which are common clusters in {*C*, *D*}. This demonstrates the capability of our method to identify common clusters.

In Fig. 1, networks {*E*, *F*} have different nodes from those in {*A*, *B*, *C*, *D*}. This is the special case discussed in Section 4.3. The $\mathbf{W}$ shown above is learned using $\mathcal{L}_\Phi$ (i.e., Eq. (7)). The learned $\mathbf{W}$ without using $\mathcal{L}_\Phi$ is shown below

$$\mathbf{W} = \begin{bmatrix} 0 & .09 & 0 & 0 & .17 & 0 \\ 0 & 0 & .20 & .24 & 0 & .18 \\ .32 & .17 & 0 & 0 & .21 & .04 \\ 0 & 0 & .23 & .25 & 0 & .19 \\ 0 & 0 & .25 & .28 & 0 & .20 \\ 0 & 0 & .13 & .14 & 0 & .11 \\ .50 & .30 & 0 & 0 & .41 & .06 \end{bmatrix}.$$

Here $\mathbf{w}_E$ and $\mathbf{w}_F$ can share the same subspace with other networks which makes the features less discriminative and not effective to group networks. This shows the importance to consider node set information (i.e., $\mathcal{L}_\Phi$) in our model.

5 Learning Algorithm

Since the objective function in Eq. (11) is not jointly convex, we optimize it by an alternating minimization approach, i.e., the objective function is alternately minimized w.r.t. one variable while fixing others. This process repeats until a stationary point is achieved. Next, we provide the solutions to the subproblems w.r.t. $\mathbf{U}$, $\mathbf{V}$, $\mathbf{S}$ and $\mathbf{W}$, respectively.

5.1 Solutions to $\{\mathbf{U}, \mathbf{V}, \mathbf{S}\}$

Considering the non-negative constraints, we use the auxiliary function approach [25] to derive iterative updating rules for $\mathbf{U}$, $\mathbf{V}$ and $\mathbf{S}$, which are summarized in Theorem 1. A theoretical analysis will be provided in Section 5.3.

Theorem 1 (Auxiliary Function of $\mathcal{L}_U(\mathbf{U})$). *Fixing other variables, alternately updating* $\mathbf{U}$, $\mathbf{V}$ *and* $\mathbf{S}$ *according to Eqs. (12), (13), and (14) monotonically decreases the value of the objective function in Eq. (11) until convergence*

$$\mathbf{U} \leftarrow \mathbf{U} \circ \left(\frac{4\sum_{i=1}^{g} \mathbf{Q}^{(i)}\mathbf{U}\mathbf{D}_W^{(i)}}{4\sum_{i=1}^{g} \mathbf{R}^{(i)} + \rho} \right)^{\frac{1}{4}} \tag{12}$$

$$\mathbf{V} \leftarrow \mathbf{V} \circ \left(\frac{2\beta\mathbf{W}^T\mathbf{S}}{2\beta\mathbf{V}\mathbf{S}^T\mathbf{S} + \rho} \right)^{\frac{1}{2}} \tag{13}$$

$$\mathbf{S} \leftarrow \mathbf{S} \circ \left(\frac{2\beta\mathbf{W}\mathbf{V}}{2\beta\mathbf{S}\mathbf{V}^T\mathbf{V} + \rho} \right)^{\frac{1}{2}}, \tag{14}$$

where $\mathbf{R}^{(i)} = (\mathbf{O}^{(i)})^T\mathbf{O}^{(i)}\mathbf{U}\mathbf{D}_W^{(i)}\mathbf{U}^T(\mathbf{O}^{(i)})^T\mathbf{O}^{(i)}\mathbf{U}\mathbf{D}_W^{(i)}$ *and* $\mathbf{Q}^{(i)} = (\mathbf{O}^{(i)})^T\mathbf{A}^{(i)}\mathbf{O}^{(i)}$.

In Eqs. (12), (13) and (14), $\circ$, $\frac{[\cdot]}{[\cdot]}$, $(\cdot)^{\frac{1}{2}}$ and $(\cdot)^{\frac{1}{4}}$ are entry-wise operators.

5.2 Solution to $\mathbf{W}$

In our objective function in Eq. (11), we have two representations $\mathbf{W}$ and $\{\mathbf{D}_W^{(i)}\}_{i=1}^g$ for the same variables, both of which will be kept in our algorithm. Considering the diagonal constraints on $\{\mathbf{D}_W^{(i)}\}_{i=1}^g$, we optimize $\{\mathbf{D}_W^{(i)}\}_{i=1}^g$ (i.e., $\mathbf{W}$) using the efficient coordinate descent method [27], which gives the following updating formula that solves each subproblem optimally

$$\begin{aligned} (\mathbf{D}_W^{(i)})_{pp} &\leftarrow \max\left((\mathbf{D}_W^{(i)})_{pp} - \frac{z_1}{z_2}, 0 \right) \\ \mathbf{W}_{pi} &\leftarrow (\mathbf{D}_W^{(i)})_{pp}, \end{aligned} \tag{15}$$

where $(\cdot)_{pp}$ ($(\cdot)_{pi}$) is the ppth (pith) entry of a matrix, and

$$\begin{aligned} z_2 &= (\hat{\mathbf{U}}^T\hat{\mathbf{U}})_{pp}^2 + \beta \\ z_1 &= (\hat{\mathbf{U}}^T\hat{\mathbf{U}}\mathbf{D}_W^{(i)}\hat{\mathbf{U}}^T\hat{\mathbf{U}})_{pp} - (\hat{\mathbf{U}}^T\mathbf{A}^{(i)}\hat{\mathbf{U}})_{pp} + \alpha\mathbf{w}_{p*}\boldsymbol{\phi}_{*i} \\ &\quad + \beta(\mathbf{D}_W^{(i)})_{pp} - \beta\mathbf{s}_{p*}\mathbf{v}_{i*}^T. \end{aligned} \tag{16}$$

Here, $\hat{\mathbf{U}} = \mathbf{O}^{(i)}\mathbf{U}$, $\mathbf{w}_{p*}$ is the pth row of $\mathbf{W}$, $\boldsymbol{\phi}_{*i}$ is the ith column of $\mathbf{\Phi}$, $\mathbf{s}_{p*}$ is the pth row of $\mathbf{S}$, $\mathbf{v}_{i*}$ is the ith row of $\mathbf{V}$.

Algorithm 1 summarizes our alternating minimization algorithm according to the solutions of $\mathbf{U}$, $\mathbf{V}$, $\mathbf{S}$ and $\mathbf{W}$.

Algorithm 1. ComClus

Input: member networks $\{\mathbf{A}^{(i)}\}_{i=1}^g$; mapping matrices $\{\mathbf{O}^{(i)}\}_{i=1}^g$; number of network groups k; latent dimension h; parameters α, β and ρ
Output: $\mathbf{U}$, $\mathbf{V}$, $\mathbf{S}$ and $\mathbf{W}$

```
Normalize {A^(i)}_{i=1}^g by Frobenius norm;
Initialize U, V, S, W with random values within (0, 1];
Construct Φ according to Eq. (6);
repeat
    repeat
        Update U by Eq. (12);
        Update V by Eq. (13);
        Update S by Eq. (14);
    until Convergence
    for i ← 1 to g do
        for p ← 1 to h do
            Update (D_W^(i))_pp by Eq. (15);
            Update W_pi ← (D_W^(i))_pp;
        end
    end
until Convergence
return U, V, S and W.
```

5.3 Algorithm Analysis

5.3.1 Convergence Analysis

In the next, we first provide the convergence analysis of the updating rule of $\mathbf{U}$ in Eq. (12) using the auxiliary function approach [25]. Then we analyze the convergence of Algorithm 1. The proofs for the convergences of $\mathbf{V}$ in Eq. (13) and $\mathbf{S}$ in Eq. (14) are similar and hence are omitted for brevity.

Definition 1. *[25] A function $Z(u, \tilde{u})$ is an* auxiliary function *for a given function $\mathcal{L}(u)$ if the conditions $Z(u, \tilde{u}) \geq \mathcal{L}(u)$ and $Z(u, u) = \mathcal{L}(u)$ are satisfied.*

Lemma 1. *[25] If Z is an auxiliary function for $\mathcal{L}$, then $\mathcal{L}$ is non-increasing under the update $u^{(t+1)} = \arg\min_u Z(u, u^{(t)})$.*

The following theorem gives the auxiliary function for the objective function in Eq. (11) w.r.t. $\mathbf{U}$.

Theorem 2. *Let $\mathcal{L}_U(\mathbf{U})$ denote the sum of all terms in the objective function in Eq. (11) that contains $\mathbf{U}$, then the following function*

$$Z(\mathbf{U}, \tilde{\mathbf{U}}) = -2 \sum_{i=1} \sum_{prtq} \mathbf{Q}^{(i)}_{pr} \tilde{\mathbf{U}}_{rt} (\mathbf{D}^{(i)}_W)_{tq} \tilde{\mathbf{U}}_{pq} \left(1 + \log \frac{\mathbf{U}_{rt}\mathbf{U}_{pq}}{\tilde{\mathbf{U}}_{rt}\tilde{\mathbf{U}}_{pq}}\right) + \sum_{i=1} \sum_{pq} \tilde{\mathbf{R}}^{(i)}_{pq} \frac{\mathbf{U}^4_{pq}}{\tilde{\mathbf{U}}^3_{pq}} + \frac{\rho}{4} \sum_{pq} \frac{\mathbf{U}^4_{pq} + 3\tilde{\mathbf{U}}^4_{pq}}{\tilde{\mathbf{U}}^3_{pq}}, \tag{17}$$

where

$$\mathbf{Q}^{(i)} = (\mathbf{O}^{(i)})^T \mathbf{A}^{(i)} \mathbf{O}^{(i)}$$
$$\tilde{\mathbf{R}}^{(i)} = (\mathbf{O}^{(i)})^T \mathbf{O}^{(i)} \tilde{\mathbf{U}} \mathbf{D}^{(i)}_W (\tilde{\mathbf{U}})^T (\mathbf{O}^{(i)})^T \mathbf{O}^{(i)} \tilde{\mathbf{U}} \mathbf{D}^{(i)}_W,$$

is an auxiliary function for $\mathcal{L}_U(\mathbf{U})$. It is also a convex function in $\mathbf{U}$ and its global minimum is

$$\mathbf{U} = \tilde{\mathbf{U}} \circ \left(\frac{4 \sum_{i=1}^{g} \mathbf{Q}^{(i)} \tilde{\mathbf{U}} \mathbf{D}^{(i)}_W}{4 \sum_{i=1}^{g} \tilde{\mathbf{R}}^{(i)} + \rho} \right)^{\frac{1}{4}}. \tag{18}$$

Proof. The formal proof can be found in the Supplementary Material, which can be found on the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/TKDE.2017.2771762. □

Next we show the convergence of updating $\mathbf{U}$ by Eq. (12) in Theorem 3.

Theorem 3. *Convergence of Eq. (12). When other variables are fixed, updating $\mathbf{U}$ according to Eq. (12) monotonically decreases the value of the objective function in Eq. (11) until convergence.*

Proof. From Theorem 2, the global minimum in Eq. (18) of the auxiliary function $Z(\mathbf{U}, \tilde{\mathbf{U}})$ is consistent with the updating rule of $\mathbf{U}$ in Eq. (12). Therefore, each time $\mathbf{U}$ is updated using the global minimum of the auxiliary function $Z(\mathbf{U}, \tilde{\mathbf{U}})$. According to Definition 1 and Lemma 1, at any iteration $\kappa \geq 0$ during updating $\mathbf{U}$, we have

$$\mathcal{L}_U(\mathbf{U}^{(\kappa)}) = Z(\mathbf{U}^{(\kappa)}, \mathbf{U}^{(\kappa)}) \geq Z(\mathbf{U}^{(\kappa+1)}, \mathbf{U}^{(\kappa)}) \geq \mathcal{L}_U(\mathbf{U}^{(\kappa+1)}),$$

where $\mathbf{U}^{(\kappa)}$ denotes the updated $\mathbf{U}$ at κth iteration. Thus $\mathcal{L}_U(\mathbf{U})$ monotonically decreases. Note that $\mathcal{L}_U(\mathbf{U})$ is the objective function in Eq. (11) w.r.t. $\mathbf{U}$. Since the objective function in Eq. (11) is bounded below by 0, the updating of $\mathbf{U}$ will converge. □

Similarly, the convergence of Eqs. (13) and (14) can be proved. Therefore, in Algorithm 1, alternately updating $\mathbf{U}$, $\mathbf{V}$ and $\mathbf{S}$ by Eqs. (12), (13) and (14) monotonically decreases the value of the objective function in Eq. (11). The coordinate descent method for updating $\mathbf{W}$ also monotonically decreases the value of the objective function in Eq. (11). Since the objective function in Eq. (11) is bounded below by 0, Algorithm 1 is guaranteed to converge.

5.3.2 Cluster Membership Inference

After obtaining $\mathbf{U}$, $\mathbf{V}$, $\mathbf{S}$, and $\mathbf{W}$, we can infer the network group of $\mathbf{A}^{(i)}$ by $j^* = \arg\max_j \mathbf{V}_{ij}$. We can infer the cluster membership of node x in network $\mathbf{A}^{(i)}$ by $p^* = \arg\max_p (\mathbf{O}^{(i)}\mathbf{U}\mathbf{D}^{(i)}_W)_{xp}$. Also, for a node x, we can infer its membership to a common cluster shared in network group j by $p^* = \arg\max_p (\mathbf{U}\mathrm{diag}(\mathbf{s}^{(j)}))_{xp}$, where $\mathbf{s}^{(j)}$ is the centroid vector of network group j. More uniquely, we can sort the values in $\mathbf{s}^{(j)}$ in descending order to identify the most common clusters in network group j.

5.3.3 Complexity Analysis

Let N and M be the maximal number of nodes and edges in any member network. Based on Eq. (12), with proper order of multiplication, updating $\mathbf{U}$ requires $O(g(Mh + Nh^2))$. Based on Eqs. (13) and (14), updating $\mathbf{V}$ and $\mathbf{S}$ both require $O(ghk)$. Based on Eq. (15), updating $\{\mathbf{D}^{(i)}_W\}_{i=1}^g$ requires $O(g(Mh + Nh^2))$. Thus the overall time complexity of our algorithm is $O(Ig(Mh + Nh^2 + hk))$ where I is the total number of iterations before convergence. In practice, k is much smaller than N and M, h is usually a small constant. Therefore, the actual time complexity can be denoted as $O(Ig(M + N))$. In real applications, $\{\mathbf{A}^{(i)}\}_{i=1}^g$ are often sparse, where M is almost linear w.r.t. N, hence our method is efficient. The experimental results show that our algorithm is almost linear w.r.t. g and N, respectively.

6 Leveraging Prior Knowledge

In real-life applications, we may have prior knowledge on the member networks that can help enhance performance. Here we consider two common types of prior knowledge that can be easily incorporated into our model.

6.1 Semi-Supervised Network Grouping

Semi-supervised clustering methods [29], [30] can dramatically improve clustering quality by exploring both labelled and unlabelled data. Here we discuss how to extend ComClus to a novel semi-supervised problem setting, where the supervision is enforced on network grouping, in contrast to that on clustering in the above mentioned methods.

Following the convention, suppose we have a set of pairwise *must-link* constraints $\mathcal{M}$, where $(i, j) \in \mathcal{M}$ implies that networks i and j should be in the same group, and a set of *cannot-link* constraints $\mathcal{C}$, where $(i, j) \in \mathcal{C}$ implies that networks i and j should be in different groups. Intuitively, for any pair $(i, j) \in \mathcal{M}$ ($(i, j) \in \mathcal{C}$), the group assignments of networks i and j should be similar (dissimilar). We measure the network group assignment similarity by inner product, i.e., $\mathbf{v}_{i*}\mathbf{v}^T_{j*}$, and penalize the following square loss function

$$\mathcal{L}_{\mathbf{\Theta}}(\{\mathbf{v}_{i*}\}_{i=1}^{g}) = \sum_{i,j\in\mathcal{M}} (\theta_{ij} - \mathbf{v}_{i*}\mathbf{v}_{j*}^T)^2 + \sum_{i,j\in\mathcal{C}} (\mathbf{v}_{i*}\mathbf{v}_{j*}^T)^2,$$

where θ_{ij} is a large value (e.g., $\theta_{ij} = 1$) to enforce the similarity between $\mathbf{v}_{i*}$ and $\mathbf{v}_{j*}$.

Let $\mathcal{S} = \mathcal{M} \cup \mathcal{C}$. We define $\mathbf{\Omega} \in \{0,1\}^{g\times g}$ such that $\mathbf{\Omega}_{ij} = 1$ if $(i,j) \in \mathcal{S}$, and $\mathbf{\Theta} \in \{0,\theta_{ij}\}^{g\times g}$ such that $\mathbf{\Theta}_{ij} = \theta_{ij}$ if $(i,j) \in \mathcal{M}$. Then the above loss function becomes

$$\mathcal{L}_{\mathbf{\Theta}}(\mathbf{V}) = \|\mathbf{\Omega} \circ (\mathbf{\Theta} - \mathbf{V}\mathbf{V}^T)\|_F^2.$$

$\mathcal{L}_{\mathbf{\Theta}}$ can be incorporated into our model by replacing $\mathcal{L}$ in Eq. (11) by

$$\mathcal{L}_{Semi}(\mathbf{U},\mathbf{V},\mathbf{S},\mathbf{W}) = \mathcal{L}(\mathbf{U},\mathbf{V},\mathbf{S},\mathbf{W}) + \mathcal{L}_{\mathbf{\Theta}}(\mathbf{V}). \quad (19)$$

6.2 Leveraging Similarity Between Networks

When we have prior knowledge about the pairwise similarity between member networks, we can form a similarity matrix $\mathbf{G} \in \mathbb{R}^{g\times g}$ with $\mathbf{G}_{ij}$ indicating the similarity between the member networks $\mathbf{A}^{(i)}$ and $\mathbf{A}^{(j)}$. Then we can utilize such information to partition the member networks. In our model, we only need to replace $\mathcal{L}$ in Eq. (11) by

$$\mathcal{L}_{Sim}(\mathbf{U},\mathbf{V},\mathbf{S},\mathbf{W}) = \mathcal{L}(\mathbf{U},\mathbf{V},\mathbf{S},\mathbf{W}) + \mathcal{L}_{G}(\mathbf{V}), \quad (20)$$

where $\mathcal{L}_G(\mathbf{V})$ is the same as the loss function in Eq. (1) except that $\mathbf{B}$ is replaced by $\mathbf{G}$ and $\mathbf{Z}$ is replaced by $\mathbf{V}$.

The objective functions using Eqs. (19) and (20) can be optimized similarly as before in Algorithm 1, except when updating $\mathbf{V}$, instead of using Eq. (13), we use

$$\mathbf{V} \leftarrow \mathbf{V} \circ \left(\frac{2\beta\mathbf{W}^T\mathbf{S} + 4\mathbf{\Pi}}{2\beta\mathbf{V}\mathbf{S}^T\mathbf{S} + 4\mathbf{\Lambda} + \rho} \right)^{\frac{1}{4}}, \quad (21)$$

where $\mathbf{\Pi} = (\mathbf{\Omega} \circ \mathbf{\Theta})\mathbf{V}$, $\mathbf{\Lambda} = (\mathbf{\Omega} \circ (\mathbf{V}\mathbf{V}^T))\mathbf{V}$ for $\mathcal{L}_{Semi}$, and $\mathbf{\Pi} = \mathbf{G}\mathbf{V}$, $\mathbf{\Lambda} = \mathbf{V}\mathbf{V}^T\mathbf{V}$ for $\mathcal{L}_{Sim}$.

Discussion. With the priori knowledge, users can guide ComClus to find common clusters that best fit the desired network groups. Note since Eq. (9) can be explained as a co-clustering of $\mathbf{W}$ by NMF, $\{\mathbf{s}^{(j)}\}_{j=1}^{k}$ can be considered as k different row entity clusters of $\mathbf{W}$. Thus Eq. (9) encourages $\{\mathbf{s}^{(j)}\}_{j=1}^{k}$ to be dissimilar to each other. Hence ComClus tends to find clusters that are common within a network group but are rare in other groups, like features that characterize each network group.

7 Experimental Results

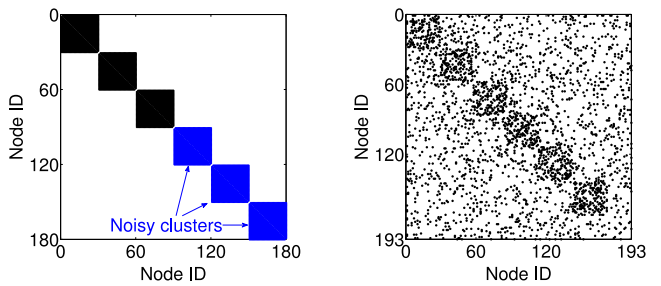
In this section, we evaluate the performance of ComClus on a variety of synthetic and real-world datasets.

7.1 Effectiveness Evaluation

We first evaluate the effectiveness of ComClus. We consider multiple networks with different clustering structures and apply ComClus to see whether network grouping and clustering can enhance each other.

7.1.1 Simulation Study

We first evaluate the clustering accuracy of ComClus using synthetic datasets. The member networks and network groups are generated as follows. Suppose we have k network groups. For each network group, we first generate an underlying clustering structure with d_c clusters (30 nodes per cluster). The k underlying clustering structures have the same set of nodes but different node cluster memberships. Then member networks are generated from each underlying clustering structure. Based on an underlying clustering structure, each member network is appended with d_n irrelevant ("noisy") clusters (30 nodes per cluster). The noisy clusters of different networks have different nodes. Thus they represent clusters that are not common in any groups. Then, we can evaluate whether the proposed method can effectively identify common clusters from noisy clusters.



(a) An example of the clustering structure of a network

(b) An example of simulated member network

Fig. 3. An illustration for synthetic dataset generation, as shown by network adjacency matrices.

Fig. 3a shows an example clustering structure using $d_c = 3$ and $d_n = 3$, where non-zero entries are set to 1. To embed noises when generating a member network, we randomly flip ω_0 fraction of "1" entries in the matrix of its underlying clustering structure to "0", and ω_1 fraction of "0" entries to "1". To make networks of different sizes, we randomly remove or add ε fraction of nodes in the matrix from the previous step. ε follows normal distribution with mean μ and standard deviation σ and its value is set between 0 and 1. Thus different networks will have different proportions of nodes to be added or removed, depending on a specific value of ε, whose mean value is controlled by μ.

In general, we want to generate synthetic networks that are not too noisy (i.e., ω_1 is not too large) nor having too dense clusters (i.e., ω_0 is not too small). Note ω_1 is usually much smaller than ω_0 because there are significantly more "0" entries than "1" entries in the matrix. A reasonable μ is a relatively small value so that not too many nodes are removed or added. An example member network with a reasonable parameter setting, $\omega_0 = 80\%$, $\omega_1 = 5\%$, $\mu = 0.1$, $\sigma = 0.05$, is shown in Fig. 3b, which contains 193 nodes.

Using this generation process, we generate two types of synthetic datasets, both have $k = 5$ network groups where each group has 10 networks (thus 50 networks in total). In the first dataset, $d_c = 6$ and $d_n = 0$. All member networks have the same set of 180 nodes. ω_0 and ω_1 are set to 80 and 5 percent respectively to simulate noise. We refer to this dataset as `SynView` dataset. In the second dataset, $d_c = 3$ and $d_n = 3$. To simulate noise, we set $\omega_0 = 80\%$, $\omega_1 = 5\%$, $\mu = 0.1$, $\sigma = 0.05$. Thus different networks have different node sets and sizes. We refer to this dataset as `SynNet` dataset. Moreover, we generate 50 networks forming one network group, whose parameter setting is the same as `SynView` dataset except for $k = 1$. We refer to it as `OneGroup` dataset, which is used to understand the effectiveness of ComClus in the conventional multi-network setting.

TABLE 2
Clustering Accuracy on Three Types of Synthetic Datasets

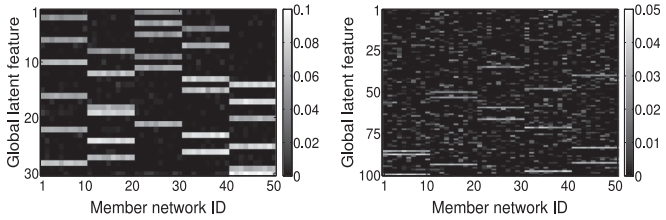
Method	OneGroup		SynView		SynNet	
	NMI	ACC	NMI	ACC	NMI	ACC
SNMF	0.4683	0.6769	0.4853	0.6900	0.3391	0.7659
Spectral	0.4519	0.6484	0.4723	0.6698	0.3145	0.7408
PairCRSC	0.9948	0.9933	0.3280	0.5437	–	–
CentCRSC	0.9441	0.9659	0.6541	0.8155	–	–
CTSC	0.9914	0.9897	0.4604	0.6587	–	–
TF	**1.0000**	**1.0000**	0.4803	0.5431	–	–
CGC	0.9729	0.9672	0.1291	0.3322	0.4498	0.7625
Naive	0.9897	0.9867	0.6834	0.8233	0.4692	0.7584
NoNClus	0.9879	0.9943	0.5572	0.7320	0.3824	0.7454
ComClus	**1.0000**	**1.0000**	**0.9764**	**0.9766**	**0.8605**	**0.9896**

We compare ComClus with the state-of-the-art methods, including (1) SNMF [26]; (2) Spectral clustering (Spectral) [31]; (3) Multi-view pair-wise co-regularized spectral clustering (PairCRSC) [4]; (4) Multi-view centroid-based co-regularized spectral clustering (CentCRSC) [4]; (5) Multi-view co-training spectral clustering (CTSC) [6]; (6) Tensor factorization (TF) [17]; (7) multi-domain co-regularized graph clustering (CGC) [7]; (8) NoNClus [1]; and (9) Naive: a naive approach that first groups member networks by clustering them using the similarities between them and then applies one of the above multi-view/domain clustering methods in each network group that gives the best accuracy.

SNMF and spectral clustering methods can only be applied on single networks. PairCRSC, CentCRSC and CTSC are multi-view graph clustering methods and can only be applied on `OneGroup` and `SynView` datasets. For TF, we use both CP and Tucker decomposition and report the best performance [17]. TF is similar to multi-view methods thus can only be applied on `OneGroup` and `SynView` datasets. CGC is a recent multi-domain graph clustering method that can be applied on `SynNet` dataset. NoNClus and Naive can be applied on all datasets given that the similarity between networks is available. To apply them, we generate a similarity matrix for the 50 member networks using the same method described above by setting $\omega_0 = 80\%$, $\omega_1 = 5\%$, $\mu = 0$, $\sigma = 0$. The generated similarity matrix allows to partition member networks into 5 groups.

The accuracies of common clusters are evaluated using both normalized mutual information (NMI) and purity accuracy (ACC), which are standard evaluation metrics. Table 2 shows the averaged results of different methods over 100 runs. The NMIs and ACCs are averaged over all member networks. The parameters are tuned for optimal performance of all methods.

From Table 2, we observe that ComClus achieves significantly better performance than other methods on `SynView` and `SynNet` datasets. The multi-view/domain clustering methods, PairCRSC, CentCRSC, CTSC, TF and CGC, assume all networks share the same underlying clustering structure thus perform well on `OneGroup` dataset but are not able to handle the other two datasets. NoNClus and Naive differentiate network clustering structures completely based on the similarity between member networks, which makes them sensitive to the noise in the similarity matrix. In contrast, ComClus is able to automatically group networks based on their shared clusters and use the grouping information to improve the clustering of individual networks.



(a) $\mathbf{W}$ on `SynView` dataset (b) $\mathbf{W}$ on `SynNet` dataset

Fig. 4. The learned cluster-level feature matrices.

Fig. 4 shows the learned $\mathbf{W}$ matrices of ComClus. Recall each column $\mathbf{w}^{(i)}$ is a cluster-level feature vector of network $\mathbf{A}^{(i)}$ and a large entry $\mathbf{w}_p^{(i)}$ indicates the selection of the latent cluster p for $\mathbf{A}^{(i)}$. From the figure, we can observe ComClus clearly detects 5 network groups (i.e., column groups) in both datasets. There are 6 shared latent dimensions, i.e., common clusters, in each network group in the `SynView` dataset and 3 shared latent dimensions in the `SynNet` dataset. These results demonstrate that ComClus can effectively group networks and correctly find common clusters in each network group.

7.1.2 20Newsgroup Dataset

Next, we evaluate the clustering accuracy of ComClus using the `20Newsgroup` dataset.[2] Here, the networks of documents are constructed as following.

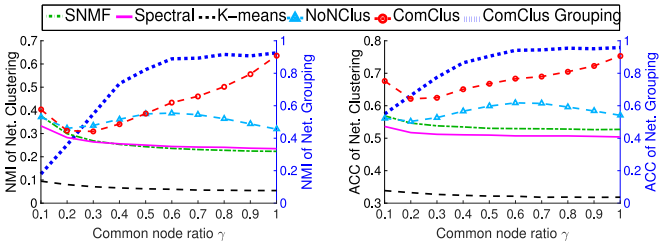
We use 12 news groups of 3 categories, Comp, Rec and Talk,[3] corresponding to 3 underlying clustering structures, each with 4 clusters (news groups). In this study, we generate 10 member networks from each category. Thus there are 30 member networks forming 3 groups corresponding to the 3 categories. Each member network contains randomly sampled 200 documents from the 4 news groups (50 documents from each news group) in a category. The adjacency matrix of documents is computed based on cosine similarity between the word frequencies of documents.

The common nodes in different member networks are generated as follows. For any two networks from the same category, a document in one network is randomly mapped to a document with the same cluster label (e.g., comp. graphics) in another network. For any two networks from different categories, the documents are randomly mapped with one-to-one relationship. We vary the ratio of common nodes, γ, from 0 to 1 to evaluate its effects.

For comparison, the single network clustering methods SNMF and Spectral clustering are performed on individual member networks. The widely used k-means clustering [32] is also selected as a baseline method, it is applied on the original document-word matrix instead of the network data. Note that multi-view clustering methods PairCRSC, CentCRSC, CTSC, and TF cannot be applied here since they require full mapping of nodes between networks. We omit CGC since it is very slow on tens of networks (see Section 7.3). To apply

2. http://qwone.com/%7Ejason/20Newsgroups/

3. *Comp*: comp.graphics, comp.os.ms-windows.misc, comp.sys. ibm. pc.hardware, comp.sys.mac.pc.hardware; *Rec*: rec.autos, rec. motorcycles, rec.sport.baseball, rec.sport.hockey; *Talk*: talk.politics.guns, talk. politics.mideast, talk.politics.misc, talk.religion.misc.



(a) NMI comparison (b) ACC comparison

Fig. 5. Performance on 20Newsgroup dataset with various common node ratio. The blue dotted curve shows the network grouping performance of ComClus.

NoNClus, we calculate the cosine similarity between the overall word frequencies of member networks.

Fig. 5 shows the averaged NMI and ACC of common clusters detected by different methods over 100 runs. In Fig. 5, there are slight initial decreases of NMI and ACC of different methods. This is because when common node ratio γ is very small, the sizes of the evaluated common clusters are small. The small sizes may affect the evaluated accuracy. However, when γ is reasonably large, e.g., $\gamma \geq 0.3$, the results are less affected by cluster sizes, which will demonstrate the general trends of different methods w.r.t. γ. In general, ComClus achieves better performance than other methods. Note ComClus is better than NoNClus although NoNClus uses the high quality similarity information between member networks. This shows the importance to group and cluster multiple networks simultaneously. The blue dotted curves in Fig. 5 shows that ComClus achieves increased NMI and ACC of network grouping as more common nodes are added. This confirms that better common cluster detection enhances network grouping.

In this study, we also test a multi-layer network community detection method MLCS [21]. It can run on 20Newsgroup dataset only when $\gamma = 1$. However, it runs out of memory on a 16 GB machine. Thus we reduce the size of the dataset such that it contains 6 networks forming 2 groups and each network has 80 nodes. Then MLCS detects 179 clusters of sizes between 5 and 13. However, each ground truth cluster has a size of 20. Since its clusters cannot match the ground truth well, its averaged NMI is 8.98 percent. In comparison, the NMI of ComClus, SNMF and spectral clustering are 17.32, 11.87 and 10.52 percent, respectively. These results validate the discussion in Section 2, i.e., MLCS aims to detect consistent clusters, but not to improve clustering accuracy. In contrast, our method is effective to improve clustering accuracy using multiple networks.

7.1.3 Reality Mining Dataset

In this section, we evaluate ComClus on the MIT reality mining proximity networks [33]. From the original dataset, we obtain 371 proximity networks about 91 subjects (e.g., faculties, staffs, students). Each network is constructed in one day between July 2004 and July 2005. In a network, any pair of subjects are linked if their phones detect each other (within certain distance) at least once in that day.

As analyzed in [33], subjects have different roles during work and out of campus, which reflects in different subject clusters (e.g., working groups or social communities) in on- and off-campus. Some clusters that occur in campus may disappear or split after work while new clusters may appear. As suggested by [33], we separate each of the 371 proximity networks by time 8 p.m. to obtain two groups of networks for on- and off-campus, respectively.

TABLE 3
Performance on RM-month Dataset

Measure	SNMF	CTSC	CGC	TF	ComClus
NMI	0.7278	0.8705	0.9083	0.9066	**1.0000**
Density	0.2019	0.1822	0.1702	0.1852	**0.2253**

Since many proximity networks are very sparse without obvious structures, we take two steps to process them. First, we extract networks from September to December 2004, which generally has more data collected than other periods. Then we aggregate the networks by month. Finally we have dataset RM-month: 8 proximity networks, 4 of them are on-campus and 4 of them are off-campus.

Next we evaluate ComClus to see if it can (1) automatically group on (off)-campus networks together; and (2) enhance the qualities of common subject clusters.

First, in our results, we observe ComClus correctly groups on-campus and off-campus networks. To evaluate the subject clusters in on-campus networks, we use the ground truth from the dataset, which indicates the subjects' affiliations, i.e., MIT media lab or business school. The averaged NMIs (over all on-campus networks) of different methods are shown in Table 3. Here NoNClus is omitted because network similarity is not available in this dataset. For spectral based methods, we report the best results, which is given by CTSC. As can be seen, ComClus exactly discovers the subject clusters in on-campus networks, while none of the baseline methods can achieve this accuracy. This shows the importance to group networks and enhance clustering by group-wise consensus. For off-campus networks, since there is no ground truth, we use internal density [34] as the cluster quality measure. As shown in Table 3, ComClus achieves the best averaged density, which indicates its capability to discover meaningful clusters in off-campus networks. These results imply that subjects may have different communities during and after work.

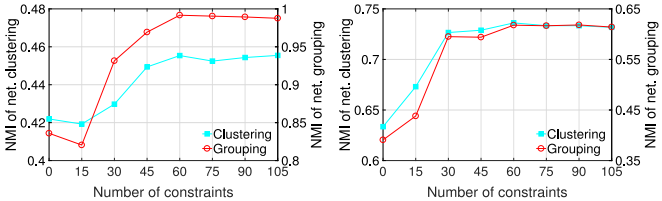
7.2 Leveraging Prior Knowledge

In the next, we evaluate how well ComClus can incorporate prior knowledge as discussed in Section 6.

7.2.1 Semi-Supervised Network Grouping

First, we show our model in Eq. (19) can effectively incorporate user-provided network group labels using two datasets. The first is the 20Newsgroup dataset, for which we fix the common node ratio $\gamma = 0.3$ and enlarge the size of each network to be 400 (100 nodes per cluster). Setting a relatively small γ can ensure there is enough room for network grouping performance to be improved by providing labels, while increasing the network size can make the networks more informative so that we can observe clear performance gain given labels. The second dataset is RM-week. In Section 7.1.3, we aggregate 371 proximity networks by month and obtain RM-month dataset. Here, to involve more networks, we aggregate the proximity networks by week and obtain 30 networks (15 on-campus and 15 off-campus).

Following [29], [30], we take a 2-fold cross validation scheme, that is, each time we use 50 percent of the member



(a) 20Newsgroup dataset (b) RM-week dataset

Fig. 6. Performance of semi-supervised network grouping of ComClus.

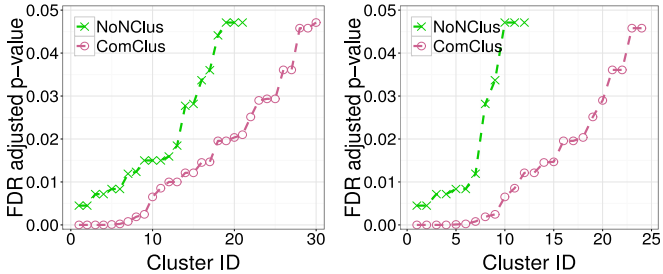
networks as training data and the remaining as test data. The must-links and cannot-links are randomly sampled from the training set. ComClus then runs on all networks, but the performance is evaluated only on the test set. The number of constraints varies from 0 to the maximal number of pairs in the training set. The results are averaged over 100 runs.

Fig. 6 shows the averaged NMIs of network grouping and clustering on both datasets. As can be seen, with more constraints, ComClus is able to effectively boost network grouping quality, which in turn improves the common cluster detection accuracy. In Fig. 6a, we observe a slight decrease of NMI when initial constraints are provided. This behavior is also observed in some existing semi-supervised clustering methods [29], [30]. One reason may be that a few constraints will distort the clustering results but are not informative enough to refine the results. Generally, ComClus can improve network grouping and clustering quality with a small number of constraints. These results also confirm that better network grouping enhances common cluster detection. Also, considering the results shown in Fig. 5, i.e., better network clustering enhances network grouping, we confirm the mutual enhancement of the two procedures of multi-network grouping and common cluster detection.

7.2.2 Leveraging Similarity Between Tissue-Specific Gene Co-Expression Networks

In this section, we apply ComClus on a real-world dataset, the tissue-specific gene co-expression networks, for case study. In a gene co-expression network, each node is a gene and an edge represents the functional association between two connected genes. By detecting gene clusters in such network, we can uncover functional modules containing genes that are functionally similar, which are useful in biomedical analysis [16], [35]. As demonstrated by [1], [7], jointly clustering multiple gene networks can enhance clustering accuracy. Thus, we construct multiple gene co-expression networks from different human tissues, which may form several tissue groups (i.e., network groups). In this study, our goal is to evaluate the capability of ComClus to incorporate the pairwise similarities between gene networks.

We select 8 tissues, Blood, Lymph node, Tonsil, Thymus, Brain, Caudate nucleus, Hypothalamus, and Cerebellum, from a recently published global map of human gene expression dataset [9]. For each tissue, we extract genes that are expressed in it, and use the expression data of these genes to construct its tissue-specific gene co-expression network. Then we have 8 different networks for 8 tissues. The edges in a network are weighted by the Pearson's correlation coefficient (normalized between $[0, 1]$) between two connected genes. There are in total 5,455 nodes and 21,097 edges in these networks. A tissue-tissue genetic similarity



(a) $\tau = 3$ (b) $\tau = 4$

Fig. 7. Evaluation on tissue-specific gene co-expression networks (Lower curve is better).

network is also constructed as the prior knowledge. The similarity is calculated using the pairwise correlation of the expression data of genes in each tissue.

We compare the supervised ComClus algorithm in Eq. (20) with NoNClus, which is also applicable when the network similarities are known [1]. In [1], NoNClus has been shown to be better than SNMF, spectral clustering and CGC. Here, multi-view clustering methods cannot be applied since different networks are about different nodes. To evaluate performance, we use the standard Gene Set Enrichment Analysis (GSEA) [16]. Specifically, for each identified gene clusters with at least 5 genes, the most significant Gene Ontology (GO) term in the biological process category [36] is assigned to it. The significance is assessed by Hypergeometric distribution [16]. Raw p-values are adjusted for multiple testing by False Discovery Rate (FDR) [37].

First, we observe both NoNClus and ComClus identify two tissue network groups: {Blood, Lymph node, Tonsil, Thymus} and {Brain, Caudate nucleus, Hypothalamus, Cerebellum}. The first group includes gland tissues and the second group includes tissues related to the brain. To understand of importance of leveraging prior knowledge in this particular application, we also apply the unsupervised ComClus algorithm (Eq. (11)) on this dataset and observe it identifies two network groups {Blood, Lymph node, Tonsil, Hypothalamus} and {Brain, Caudate nucleus, Cerebellum, Thymus}. Here it incorrectly groups the networks of Hypothalamus and Thymus. This is because the networks in this application are too noisy to provide sufficiently clear clustering structures that are effective for grouping them automatically. Hence, in the following, we focus on supervised ComClus and compare its clustering with NoNClus.

To compare the common gene clusters detected by NoNClus and ComClus, we take the following strategy. For each method, we have a list of enriched GO terms for each network, each term corresponding to a detected gene cluster. If a GO term is significantly enriched in at least τ ($\tau = 3$ or $\tau = 4$) tissues in a group and is not enriched in any tissue in another group, we consider it as a discriminative GO term. We collect all gene clusters corresponding to all discriminative GO terms from all networks, sort these clusters in ascending order of their p-values.

Fig. 7 shows the p-values of the discriminative gene clusters of both methods, using significance threshold 0.05. In both cases, ComClus detects more such clusters than NoNClus. Moreover, the discriminative gene clusters detected by ComClus are more significant (i.e., with smaller p-values) than those identified by NoNClus. This is because ComClus tends

to detect cluster-level features that are distinct to each network group, as discussed in Section 6.2, while NoNClus does not consider to use clusters to enhance network groups. Thus the clusters detected by ComClus are more meaningful than NoNClus w.r.t. each network group. For example, in the results of ComClus, GO:0050771 is enriched in all networks in the tissue group related to brain but none in another group. It is ranked 2 in $\mathbf{S}$ by ComClus (recall $\mathbf{s}_p^{(j)}$ indicates the commonalities of cluster p to group j). The annotation of this GO term is "negative regulation of axonogenesis", which is known to relate to the activities of axon, neuron and brain [38].

7.2.3 A Case Study on Co-Author Networks

Next, we perform a case study on co-author networks of different research conferences. Here, node clusters represent author communities, network groups represent research areas. Since there is no ground truth labels of author clusters in this dataset, some clusters detected by ComClus will be discussed as case studies.

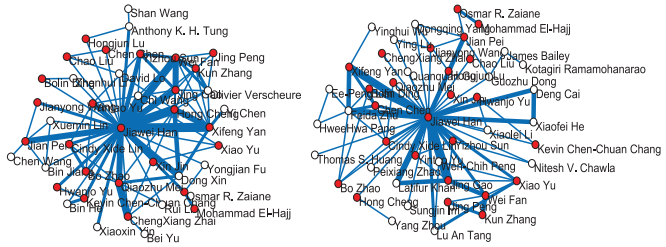
We extract 12 top conferences[4] from the DBLP bibliographic dataset [39]. These conferences form four areas, data mining (DM), database (DB), machine learning (ML) and theory (TH). We first extract productive authors that have at least 5 papers. Then for each conference, we construct its co-author network using authors that have papers in it. An edge in a co-author network is weighted by the number of papers the two connected authors cooperated in one conference. Moreover, for each network, we use its largest component to ensure the network is connected. Then the number of nodes in each network varies from 440 to 1,256, and the number of edges ranges from 918 to 4,585. There are in total 10,486 nodes and 27,239 edges in these networks. A conference-conference topic similarity network is also constructed. The topic similarities are calculated using the pairwise cosine similarity between the bag-of-words vectors of conferences, where the key words are obtained from all the paper titles in each conference.

As analyzed in Sectoin 5.3.2, the entries in each column of $\mathbf{S}$ can be sorted to identify the most common clusters to a network group. Our goal of this study is to evaluate this unique property of ComClus. Next, we apply ComClus to see what author clusters are most common (thus important) to each research area (network group). To our best knowledge, there is no existing methods to address such common cluster ranking task.

First, the detected conference groups are {KDD, ICDM}, {VLDB, SIGMOD, ICDE}, {NIPS, ICML, IJCAI, AAAI}, {FOCS, STOC, SODA}. This matches our intuition. Similar to Section 7.2.2, we also apply the unsupervised ComClus on this dataset. It detects conference groups {KDD}, {ICDM, SODA}, {VLDB, SIGMOD, ICDE} and {NIPS, ICML, IJCAI, AAAI, FOCS, STOC}. As can be seen, it obtains partially correct groups, demonstrating the importance of the prior knowledge in this application.

The top 1 ranked author clusters by the supervised ComClus in DM area are shown in Fig. 8. In Fig. 8, we observe the clusters are formed mostly by Dr. Jiawei Han's collaborators, which is a well known group in DM area. The red nodes represent common nodes. Here we also show some

4. KDD, ICDM, VLDB, SIGMOD, ICDE, NIPS, ICML, IJCAI, AAAI, FOCS, STOC, and SODA.



(a) The cluster in KDD (b) The cluster in ICDM

Fig. 8. The top 1 ranked clusters by ComClus in two DM conferences. The edge width is proportional to the number of co-author papers. The red nodes are common nodes. The clusters induced by common nodes represent common clusters in DM area.

uncommon nodes in each conference that are also densely connected in the clusters. Note only the clusters induced by common nodes represent common clusters. The internal densities of these two clusters are 0.1890 and 0.1179, respectively. As a reference, the averaged densities (over 100 runs) of random node sets of the same sizes to these two clusters in their respective networks are 0.0095 and 0.0142, respectively. Thus the detected clusters are meaningful considering the sparsity of the networks.

Moreover, in the top 10 detected clusters in the DM area, we observe Dr. Philip S. Yu's group, Dr. Christos Faloutsos's group, Dr. Wei Wang's group and so on. All are well known and relevant. Similar results are also found in the other three areas. These results further validate the capability of ComClus to identify common clusters in network groups.

7.3 Performance Evaluation

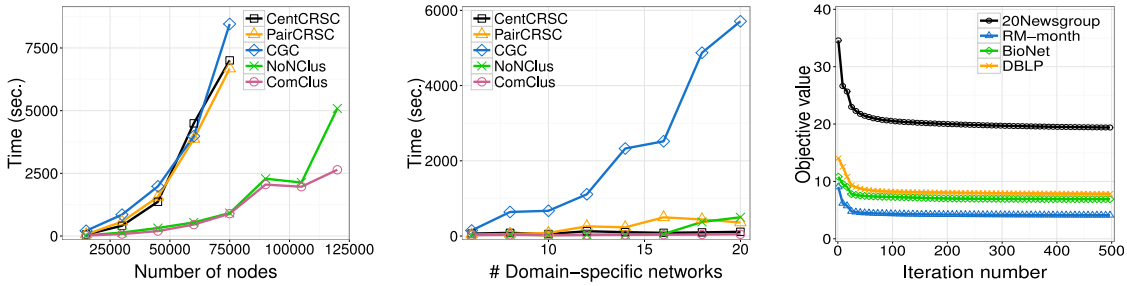
7.3.1 Running Time Evaluation

In this section, we evaluate the efficiency of ComClus w.r.t. the size and number of member networks. For this purpose, we use `SynView` dataset such that other multi-view/domain network clustering methods can be applied for comparison (CTSC is omitted since it does not guarantee convergence). In the experiment, each method is running 10 times and the averaged results are reported.

Fig. 9a shows the running time w.r.t. the size of the member networks. There are 6 networks, the network size is measured by the total number of nodes in all networks. We omit some results of PairCRSC, CentCRSC and CGC because of their high memory and running time costs. Fig. 9b shows the running time w.r.t. the number of networks, where we fix the number of nodes of each network to be 2500 and increase the number of networks. From the results, we observe ComClus has almost linear running time w.r.t. the size and number of networks, respectively. This is consistent with the complexity analysis in Section 5.3.3. ComClus is faster than other multi-view/domain network clustering methods since PairCRSC and CGC perform time-consuming pairwise regularization between networks, and the eigendecomposition of PairCRSC and CentCRSC on non-sparse matrices are both time and space consuming.

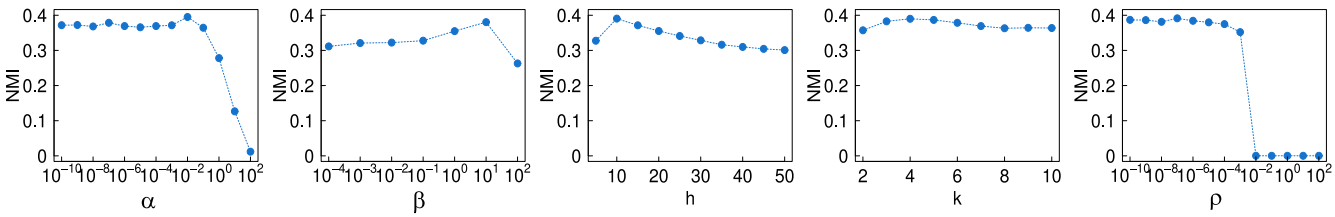
7.3.2 Convergence Evaluation

Next, we evaluate the algorithmic convergence of ComClus. Fig. 9c shows the value of the objective function in Eq. (11) w.r.t. the number of iterations on different datasets. As can be seen, the objective values monotonically decrease, which



(a) Running time w.r.t. network size (b) Running time w.r.t. # of networks (c) Alg. convergence of ComClus

Fig. 9. Performance evaluation results. In (c), the BioNet dataset refers to the gene co-expression networks used in Section 7.2.2.



(a) Varying α (b) Varying β (c) Varying h (d) Varying k (e) Varying ρ

Fig. 10. Parameter study results. The shown NMI trends are obtained by varying one parameter while fixing others.

is consistent with our theoretical analysis in Section 5.3.1. Typically, $100 \sim 200$ iterations are sufficient for ComClus to converge. This suggests its efficiency in terms of iteration.

7.3.3 Parameter Study

There are four major parameters, α, β, h and k in the proposed model. α controls the regularization on network node sets (see Section 4.3). β balances the contribution of network grouping (see Section 4.4). h is the global latent dimension. k is the number of network groups. There is an optional parameter ρ which can be used to enforce the sparseness of the learned variables or simply be set as a small value to provide stabilities of the updating rules in Eqs. (12), (13) and (14). Next, we use the `20Newsgroup` dataset to study the impact of each parameter on the clustering accuracy.

Fig. 10 shows the trends of NMI by varying each parameter in turn while fixing others. Generally, ComClus performs stably in a relatively wide range of each parameter. Specifically, the best α is around 10^{-2}, indicating the importance to consider node set regularization. The best β lies between 1 to 10, verifying that better network grouping helps better network clustering. For h and k, their best values are around 10 and 3 (or 4), respectively, which are exactly the number of all clusters (i.e., 12) and the number of groups (i.e., 3) in this dataset. Finally, a small ρ works well on this dataset, suggesting a moderate sparseness. When ρ becomes too large, all variables tend to be zero, resulting in the sharp accuracy drop in the figure. In our experiments, the configurations for α, β and ρ as discussed above apply to other datasets. k and h are usually set according to the number and size of member networks in different applications. Generally, we can determine them as following. Suppose there are in total n nodes in all member networks, we can assume the average cluster size to be z, and test h by $\frac{n}{z}$ for several z values, such as $\frac{n}{10}$, $\frac{n}{20}$, $\frac{n}{30}$, and use the one with the best performance (in terms of accuracy or conductance). Usually, k can be tested using several values smaller than 10. This is because in many applications, the number of network groups is small.

8 Conclusion

In this paper, we study multi-network clustering by considering network groups. The existing approaches assume all networks agree on a common underlying clustering structure. However, real-world applications suggest shared clustering structures in underlying network groups. To enhance clustering accuracy, we propose ComClus that can detect networks sharing similar clustering structures and group them together. ComClus treats node clusters as features of networks, and jointly group networks and infer shared cluster-level feature subspaces in network groups. Moreover, we extend ComClus to a semi-supervised setting that can leverage prior knowledge on network grouping to further boost clustering accuracy. Extensive experimental results demonstrate the effectiveness of the proposed method.

Acknowledgments

This work was partially supported by US National Science Foundation grant IIS-1664629, US National Science Foundation CAREER, and NIH grant R01 GM115833.

References

[1] J. Ni, H. Tong, W. Fan, and X. Zhang, "Flexible and robust multi-network clustering," in *Proc. ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2015, pp. 835–844.

[2] J. Ni, H. Tong, W. Fan, and X. Zhang, "Inside the atoms: Ranking on a network of networks," in *Proc. ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2014, pp. 1356–1365.

[3] Y. Chi, X. Song, D. Zhou, K. Hino, and B. L. Tseng, "Evolutionary spectral clustering by incorporating temporal smoothness," in *Proc. ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2007, pp. 153–162.

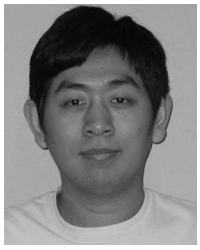
[4] A. Kumar, P. Rai, and H. Daume, "Co-regularized multi-view spectral clustering," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2011, pp. 1413–1421.

[5] D. Zhou and C. J. Burges, "Spectral clustering and transductive learning with multiple views," in *Proc. Int. Conf. Mach. Learn.*, 2007, pp. 1159–1166.

[6] A. Kumar and H. Daumé, "A co-training approach for multi-view spectral clustering," in *Proc. Int. Conf. Mach. Learn.*, 2011, pp. 393–400.

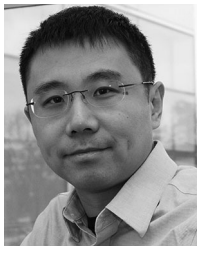
[7] W. Cheng, X. Zhang, Z. Guo, Y. Wu, P. F. Sullivan, and W. Wang, "Flexible and robust co-regularized multi-domain graph clustering," in *Proc. ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2013, pp. 320–328.
[8] R. Liu, W. Cheng, H. Tong, W. Wang, and X. Zhang, "Robust multi-network clustering via joint cross-domain cluster alignment," in *Proc. IEEE Int. Conf. Data Mining*, 2015, pp. 291–300.
[9] M. Lukk, et al., "A global map of human gene expression," *Nat. Biotechnol.*, vol. 28, no. 4, pp. 322–324, 2010.
[10] A. Bossi and B. Lehner, "Tissue specificity and the human protein interaction network," *Mol. Syst. Biol.*, vol. 5, no. 1, 2009, Art. no. 260.
[11] J. Ni, M. Koyuturk, H. Tong, J. Haines, R. Xu, and X. Zhang, "Disease gene prioritization by integrating tissue-specific molecular networks using a robust multi-network model," *BMC Bioinf.*, vol. 17, no. 1, 2016, Art. no. 453.
[12] H.-P. Kriegel, P. Kröger, and A. Zimek, "Clustering high-dimensional data: A survey on subspace clustering, pattern-based clustering, and correlation clustering," *Trans. Knowl. Discovery Data*, vol. 3, no. 1, 2009, Art. no. 1.
[13] P. J. Mucha, T. Richardson, K. Macon, M. A. Porter, and J.-P. Onnela, "Community structure in time-dependent, multiscale, and multiplex networks," *Sci.*, vol. 328, no. 5980, pp. 876–878, 2010.
[14] J. Kim and J.-G. Lee, "Community detection in multi-layer graphs: A survey," *ACM SIGMOD Rec.*, vol. 44, no. 3, pp. 37–48, 2015.
[15] A. Strehl and J. Ghosh, "Cluster ensembles—A knowledge reuse framework for combining partitionings," in *Proc. Nat. Conf. Artif. Intell.*, 2002, pp. 93–99.
[16] S. Asur, D. Ucar, and S. Parthasarathy, "An ensemble framework for clustering protein–protein interaction networks," *Bioinf.*, vol. 23, no. 13, pp. i29–i40, 2007.
[17] T. G. Kolda and B. W. Bader, "Tensor decompositions and applications," *SIAM Rev.*, vol. 51, no. 3, pp. 455–500, 2009.
[18] J. C. Ho, J. Ghosh, and J. Sun, "Marble: High-throughput phenotyping from electronic health records via sparse nonnegative tensor factorization," in *Proc. ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2014, pp. 115–124.
[19] L. Tang, H. Liu, J. Zhang, and Z. Nazeri, "Community evolution in dynamic multi-mode networks," in *Proc. ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2008, pp. 677–685.
[20] Z. Zeng, J. Wang, L. Zhou, and G. Karypis, "Coherent closed quasi-clique discovery from large dense graph databases," in *Proc. ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2006, pp. 797–802.
[21] B. Boden, S. Günnemann, H. Hoffmann, and T. Seidl, "Mining coherent subgraphs in multi-layer graphs with edge labels," in *Proc. ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2012, pp. 1258–1266.
[22] A. Silva, W. Meira Jr, and M. J. Zaki, "Mining attribute-structure correlated patterns in large attributed graphs," *Proc. VLDB Endowment*, vol. 5, no. 5, pp. 466–477, 2012.
[23] S. Günnemann, I. Färber, and T. Seidl, "Multi-view clustering using mixture models in subspace projections," in *Proc. ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2012, pp. 132–140.
[24] E. P. Xing, M. I. Jordan, S. Russell, and A. Y. Ng, "Distance metric learning with application to clustering with side-information," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2003, pp. 505–512.
[25] D. D. Lee and H. S. Seung, "Algorithms for non-negative matrix factorization," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2001, pp. 556–562.
[26] C. H. Ding, X. He, and H. D. Simon, "On the equivalence of nonnegative matrix factorization and spectral clustering," in *Proc. SIAM Int. Conf. Data Mining*, 2005, pp. 606–610.
[27] C.-J. Hsieh and I. S. Dhillon, "Fast coordinate descent methods with variable selection for non-negative matrix factorization," in *Proc. ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2011, pp. 1064–1072.
[28] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge, U.K.: Cambridge Univ. Press, 2004.
[29] S. Basu, M. Bilenko, and R. J. Mooney, "A probabilistic framework for semi-supervised clustering," in *Proc. ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2004, pp. 59–68.
[30] M. Bilenko, S. Basu, and R. J. Mooney, "Integrating constraints and metric learning in semi-supervised clustering," in *Proc. Int. Conf. Mach. Learn.*, 2004, Art. no. 11.
[31] J. Shi and J. Malik, "Normalized cuts and image segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, no. 8, pp. 888–905, Aug. 2000.
[32] J. MacQueen, "Some methods for classification and analysis of multivariate observations," in *Proc. 5th Berkeley Symp. Math. Statist. Probability*, 1967, pp. 281–297.
[33] N. Eagle, A. S. Pentland, and D. Lazer, "Inferring friendship network structure by using mobile phone data," *Proc. Nat. Academy Sci. USA*, vol. 106, no. 36, pp. 15 274–15 278, 2009.
[34] J. Leskovec, K. J. Lang, and M. Mahoney, "Empirical comparison of algorithms for network community detection," in *Proc. Int. Conf. World Wide Web*, 2010, pp. 631–640.
[35] Y. Chen, L. Li, G.-Q. Zhang, and R. Xu, "Phenome-driven disease genetics prediction toward drug discovery," *Bioinf.*, vol. 31, no. 12, pp. i276–i283, 2015.
[36] M. Ashburner, et al., "Gene ontology: Tool for the unification of biology," *Nature Genetics*, vol. 25, no. 1, pp. 25–29, 2000.
[37] Y. Benjamini and D. Yekutieli, "The control of the false discovery rate in multiple testing under dependency," *Ann. Statist.*, vol. 29, pp. 1165–1188, 2001.
[38] J. Ule, et al., "Nova regulates brain-specific splicing to shape the synapse," *Nature Genetics*, vol. 37, no. 8, pp. 844–852, 2005.
[39] J. Tang, J. Zhang, L. Yao, J. Li, L. Zhang, and Z. Su, "ArnetMiner: Extraction and mining of academic social networks," in *Proc. ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, 2008, pp. 990–998.

Jingchao Ni received the bachelor's degree in electronic and information engineering from Sichuan University, in 2012. He is currently working toward the PhD degree in the College of Information Sciences and Technology, Pennsylvania State University. His research interests include data mining, machine learning, and bioinformatics.



Wei Cheng received the PhD degree from UNC at Chapel Hill, in 2015. He is a research staff member in the Data Science Department, NEC Laboratories America. His research interests include data mining, machine learning, and bioinformatics.

Wei Fan received the PhD degree from Columbia University, in 2000. He is the director of the Tencent Medical AI Lab. His research interests include data mining and machine learning.



Xiang Zhang received the PhD degree from UNC at Chapel Hill, in 2011. He is an associate professor in the College of Information Sciences and Technology, Pennsylvania State University. His research interests include data mining, bioinformatics, and databases.