Multi-View Spectral Clustering for Graphs with Multiple View Structures*

Yorgos Tsitsikas† Evangelos E. Papalexakis†

Abstract

Despite the fundamental importance of clustering, to this day, much of the relevant research is still based on ambiguous foundations, leading to an unclear understanding of whether or how the various clustering methods are connected with each other. In this work, we provide an additional stepping stone towards resolving such ambiguities by presenting a general clustering framework that subsumes a series of seemingly disparate clustering methods, including various methods belonging to the widely popular spectral clustering framework. In fact, the generality of the proposed framework is additionally capable of shedding light to the largely unexplored area of multi-view graphs where each view may have differently clustered nodes. In turn, we propose GenClus: a method that is simultaneously an instance of this framework and a generalization of spectral clustering, while also being closely related to k-means as well. This results in a principled alternative to the few existing methods studying this special type of multi-view graphs. Then, we conduct in-depth experiments, which demonstrate that GenClus is more computationally efficient than existing methods, while also attaining similar or better clustering performance. Lastly, a qualitative real-world case-study further demonstrates the ability of GenClus to produce meaningful clusterings.

Keywords— Multi-graph; Multi-modal graph; Multi-view graph; Multi-layer graph; Multi-aspect graph; Multi-plex graph; Spectral clustering; Clustering; Tensor Decomposition; Tensor Factorization

1 Introduction.

Theoretical and computational developments in linear algebra and graph theory have set the foundations for the spectral clustering family of methods [20], [12], [23], [6], which has proven to be one of the most successful clustering paradigms. However, although researchers have been defining novel and intricate graph types that aim to capture more complex information compared to traditional graphs [11], extending spectral clustering to such graphs is still not very well understood. Multi-view graphs are a popular instance of such graphs, which consist of sets of simple graphs called views. Each view consists of the same set of nodes, but a potentially different set of edges. Prominent examples are time-evolving graphs [21], where different views emerge as edges appear or disappear at different points in time, and multi-relational knowledge graphs [14], which describe the relationships between entities, and where each relation type gives birth to a different view. The main rationale for developing models for multi-view graphs is that they are often able to obtain embeddings that better capture the intricacies of the graph structure as compared to considering individual views alone [16].

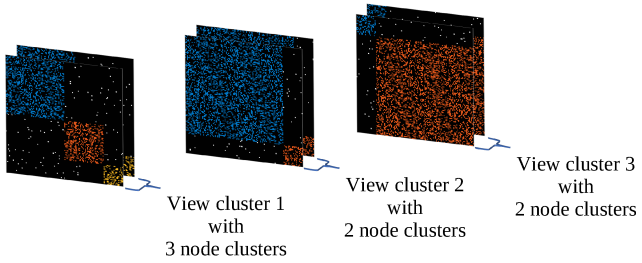


Figure 1: Example of multi-view graph with 6 views grouped into 3 view clusters, each corresponding to a different node clustering with 3, 2 and 2 node clusters, respectively. Each view is visualized via its adjacency matrix and darker colors represent lower edge weights.

Note that, existing research on multi-view graphs has primarily focused on graphs whose views are all part of the same underlying node clustering structure [22], [9]. However, recently there have also been attempts to extend such methods to multi-view graphs with multiple view structures [13], [4], [2]. Multi-view graphs of this type may contain multiple groups of views, each corresponding to a unique clustering of nodes, as illustrated in Figure 1. Also, although there have been attempts to adapt spectral clustering to multi-view graphs [22], [9], to the best of our knowledge no such extension exists for multi-view graphs with multiple view structures. Additionally, since many relevant methods have been developed independently of each other, it is not yet understood whether or how they may be related to each other. To this end, in this work we make the following contributions:

*The full version of the paper can be accessed at `https://arxiv.org/abs/2501.11422`

†University of California, Riverside. {gtsit001@ucr.edu, epapalex@cs.ucr.edu}

Symbol	Definition
x,X	Scalar
$\mathbf{x}$	Column vector
$\mathbf{X}$	Matrix
$\boldsymbol{\mathcal{X}}$	Tensor
$\mathcal{X}$	Set
$\|\|\cdot\|\|$	Frobenious norm
$\text{nnz}\,(\cdot)$	Number of non-zero elements
$\mathbf{I}$	Identity matrix
$\mathbf{0}$	All zeros column vector/matrix/tensor
$\mathbf{1}$	All ones column vector/matrix/tensor
$\text{diag}\,(\mathbf{x})$	Diagonal matrix with $\mathbf{x}$ as its diagonal
$\text{diag}\,(\mathbf{X})$	Vector consisting of the diagonal elements of $\mathbf{X}$
$\mathbf{D}_{\mathbf{X}}^{(k)}$	$\text{diag}\,(\mathbf{X}_{k:})$
$\text{Tr}\,(\mathbf{X})$	Trace of $\mathbf{X}$
$\mathbf{X}^T$	Transpose of $\mathbf{X}$
$\mathbf{x} \succeq \mathbf{y}$	$\forall i\ \mathbf{x}_i \geq \mathbf{y}_i$
$[\![\mathbf{U}, \mathbf{V}, \mathbf{W}]\!]$	PARAFAC with factor matrices $\mathbf{U}$, $\mathbf{V}$ and $\mathbf{W}$
$[\mathbf{U}, \mathbf{V}, \mathbf{W}]$	Horizontal concatenation of $\mathbf{U}$, $\mathbf{V}$ and $\mathbf{W}$
$\odot$	Column-wise Khatri–Rao product
$\times_n$	Mode-n product
$\mathbf{X}_{(n)}$	Mode-n matricization of $\boldsymbol{\mathcal{X}}$

Table 1: Table of Symbols. See [7] for further details on tensor notation and operations.

- **Unifying Clustering Framework.** We present a framework encapsulating a series of existing clustering methods able to model data as complex as multi-view graphs with multiple view structures.
- **A Principled Multi-View Graph Clustering Method.** We propose GenClus, a novel multi-view clustering method for graphs with multiple view structures, which can be seen as both a special case of the aforementioned framework and a generalization of the celebrated spectral clustering family of methods. We also show that it is very closely connected to k-means [10] as well.
- **In-Depth Experimentation**[1]**.** We design a series of experiments on both artificial and real-world data to assess the performance of GenClus both quantitatively and qualitatively. In-depth comparisons with other baselines are performed as well.

2 Proposed Unifying Framework.

Consider K adjacency matrices, $\{\mathbf{X}^{(k)}\}_{k=1}^{K}$, where each is of size $I \times I$ and corresponds to a different view of a multi-view graph. Additionally, assume that these views can be clustered into M groups, with each group corresponding to a different clustering of nodes. For convenience, we will refer to such a node clustering structure as a view structure. Given these, in this section we present an overview of existing methods designed to model and extract such clusterings. Due to space limitations, we will only discuss the embedding generation stage of these methods, but a more thorough discussion on obtaining the final clusterings is available in Appendix A.1. Then, we demonstrate that all these methods can be expressed as special cases of a unifying framework, as illustrated in Figure 2. Specifically, we show that they can be expressed as variants of PARAFAC [1], [5], where the third factor matrix is constrained to be the product of two matrices.

Please note that the goal of this unifying framework is not necessarily to precisely encapsulate the exact optimization problems of these methods. Rather, its goal is to abstract their essence. For example, we will assume that a method is conceptually aiming to calculate the same model, independent of whether it algorithmically imposes a constraint as a hard constraint or as a soft constraint. Similarly, we will assume that it conceptually remains the same, independent of whether it imposes constraints like non-negativity or sparsity.

2.1 ComClus.

ComClus [13] can be expressed as

$$\inf_{\mathbf{U},\mathbf{W},\mathbf{A},\mathbf{B}\geq 0} \sum_{k=1}^{K} \left|\left|\mathbf{X}^{(k)} - \mathbf{O}^{(k)}\mathbf{U}\mathbf{D}_{\mathbf{W}}^{(k)}(\mathbf{O}^{(k)}\mathbf{U})^T\right|\right|^2 + r(\mathbf{U},\mathbf{W},\mathbf{A},\mathbf{B}) \tag{2.1}$$

where $r(\mathbf{U},\mathbf{W},\mathbf{A},\mathbf{B}) := \beta||\mathbf{W} - \mathbf{A}\mathbf{B}||^2 + \rho(||\mathbf{U}||_1 + ||\mathbf{A}||_1 + ||\mathbf{B}||_1)$, each $\mathbf{O}^{(k)}$ is a user-defined indicator matrix, and β and ρ are user-defined penalty parameters. By making the closed-world assumption [14], the first term becomes equal to

$$\sum_{k=1}^{K} \left|\left|\mathbf{X}^{(k)} - \mathbf{U}\mathbf{D}_{\mathbf{W}}^{(k)}\mathbf{U}^T\right|\right| = ||\boldsymbol{\mathcal{X}} - [\![\mathbf{U},\mathbf{U},\mathbf{W}]\!]||$$

where $\boldsymbol{\mathcal{X}}_{::k} := \mathbf{X}^{(k)}$. Therefore, ComClus can be interpreted as aiming to approximate $\boldsymbol{\mathcal{X}}$ by $[\![\mathbf{U},\mathbf{U},\mathbf{W}]\!]$ such that $\mathbf{W} \approx \mathbf{A}\mathbf{B}$, where $\mathbf{U},\mathbf{A},\mathbf{B}$ are sparse and non-negative, and $\mathbf{W}$ is non-negative.

2.2 Richcom.

Richcom [4] can be expressed as

$$\inf_{\substack{\mathbf{U}^{(m)},\mathbf{V}^{(m)},\\ \mathbf{a}^{(m)}}} \left|\left|\boldsymbol{\mathcal{X}} - \sum_{m=1}^{M}\left(\mathbf{U}^{(m)}\mathbf{V}^{(m)^T}\right)\times_3 \mathbf{a}^{(m)}\right|\right|^2 + \sum_{m=1}^{M} r\left(\mathbf{U}^{(m)},\mathbf{V}^{(m)},\mathbf{a}^{(m)}\right) \tag{2.2}$$

[1] An implementation of GenClus, along with the experiment scripts, is available at `https://github.com/gtsitsik/genclus`

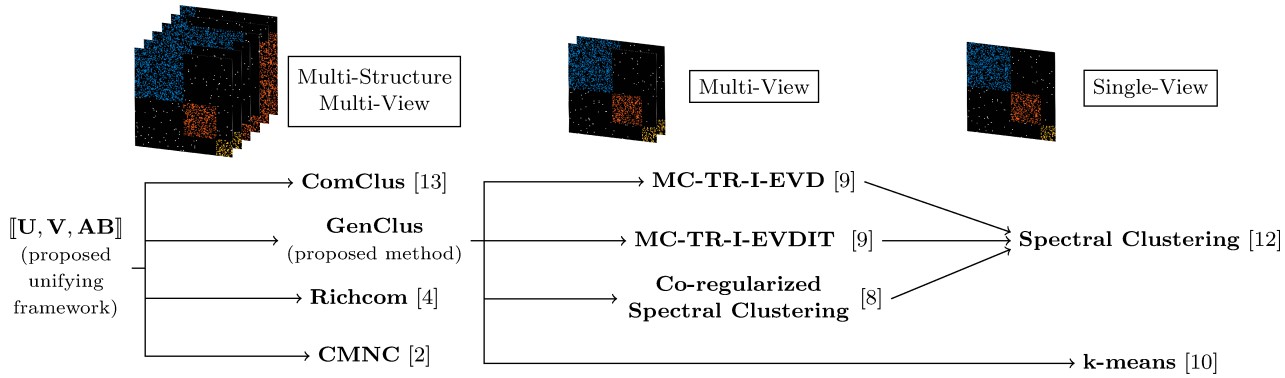


Figure 2: For each arrow the method on its right can be seen as a special case of the method on its left.

where $r\left(\mathbf{U}^{(m)}, \mathbf{V}^{(m)}, \mathbf{a}^{(m)}\right)$ encodes the sparsity and non-negativity constraints. We notice that

$$\sum_{m=1}^{M}\left(\mathbf{U}^{(m)}\mathbf{V}^{(m)^T}\right)\times_3 \mathbf{a}^{(m)} = \sum_{m=1}^{M}[\![\mathbf{U}^{(m)}, \mathbf{V}^{(m)}, \mathbf{a}^{(m)}\mathbf{1}^T]\!] = [\![\mathbf{U}, \mathbf{V}, \mathbf{AB}]\!]$$

where $\mathbf{U} := \left[\mathbf{U}^{(1)}, \cdots, \mathbf{U}^{(M)}\right]$, $\mathbf{V} := \left[\mathbf{V}^{(1)}, \cdots, \mathbf{V}^{(M)}\right]$, $\mathbf{A} := \left[\mathbf{a}^{(1)}, \cdots, \mathbf{a}^{(M)}\right]$ and $\mathbf{B}$ is an appropriate indicator matrix. The constraints of Richcom are then equivalent to requiring $\mathbf{U}$,$\mathbf{V}$, and $\mathbf{A}$ to be sparse and non-negative. Notice that, as opposed to ComClus, $\mathbf{B}$ here is a fixed matrix that needs to be defined by the user.

2.3 Centroid-based Multilayer Network Clustering (CMNC). CMNC [2] is similar to Richcom with their differences being that for CMNC $\mathbf{U}^{(m)} = \mathbf{V}^{(m)}$ for all m, $\sum_{m=1}^{M}\mathbf{a}_i^{(m)} = 1$ for all i, no sparsity is imposed and each view of the data tensor is preprocessed as $\mathbf{D}^{(k)^{-\frac{1}{2}}}\mathbf{X}^{(k)}\mathbf{D}^{(k)^{-\frac{1}{2}}}$ where $\mathbf{D}^{(k)} := \operatorname{diag}\left(\sum_{i=1}^{I}\mathbf{X}_{:i}^{(k)}\right)$.

2.4 Spectral Clustering. In spectral clustering [20], [12], [23], [6], an important quantity is the Laplacian, $\mathbf{L}$, defined as $\mathbf{D} - \mathbf{X}$, with $\mathbf{D} := \operatorname{diag}\left(\sum_{i=1}^{I}\mathbf{X}_{:i}\right)$. In turn, the symmetric normalized Laplacian, $\mathbf{L}_{sym}$, is defined as $\mathbf{D}^{-\frac{1}{2}}\mathbf{L}\mathbf{D}^{-\frac{1}{2}}$. Note that the eigenspace of $\mathbf{L}_{sym}$ corresponding to its eigenvalue $\lambda \in [0, 2]$ is also the eigenspace of $\mathbf{S} := \mathbf{I} - \mathbf{L}_{sym} = \mathbf{D}^{-\frac{1}{2}}\mathbf{X}\mathbf{D}^{-\frac{1}{2}}$ corresponding to its eigenvalue $1 - \lambda \in [-1, 1]$.

2.4.1 Single-View Spectral Clustering. Single-view spectral clustering can be formulated as $\inf_{\mathbf{U}^T\mathbf{U}=\mathbf{I}}\left|\left|-\mathbf{L}_{sym} - \mathbf{U}\mathbf{U}^T\right|\right|$, and notice that $\mathbf{U}\mathbf{U}^T = [\![\mathbf{U}, \mathbf{U}, \mathbf{1}^{\mathbf{T}}]\!] = [\![\mathbf{U}, \mathbf{U}, \mathbf{AB}]\!]$, where $\mathbf{A} := 1$ and $\mathbf{B} := \mathbf{1}^T$.

2.4.2 Multi-View Spectral Clustering. In [9] the authors proposed the MC-TR-I-EVDIT method which models multi-view spectral clustering as

$$\sup_{\substack{\mathbf{U}^T\mathbf{U}=\mathbf{I}\\ \mathbf{a}\succeq\mathbf{0}, ||\mathbf{a}||=1}}\sum_{k=1}^{K}\operatorname{Tr}\left(\mathbf{U}^T\mathbf{a}_k\mathbf{S}^{(k)}\mathbf{U}\right). \tag{2.3}$$

They also proposed the MC-TR-I method which can be seen as a variant of MC-TR-I-EVDIT where all elements of $\mathbf{a}$ are further constrained to be identical.

Then, in [8] the authors propose two forms of co-regularized spectral clustering with one utilizing a centroid-based co-regularization as

$$\sup_{\substack{\mathbf{U}^{(k)^T}\mathbf{U}^{(k)}=\mathbf{I}\\ \mathbf{U}^{*T}\mathbf{U}^{*}=\mathbf{I}}}\sum_{k=1}^{K}\operatorname{Tr}\left(\mathbf{U}^{(k)^T}\mathbf{S}^{(k)}\mathbf{U}^{(k)}\right) + \lambda_k\operatorname{Tr}\left(\mathbf{U}^{(k)}\mathbf{U}^{(k)^T}\mathbf{U}^{*}\mathbf{U}^{*T}\right) \tag{2.4}$$

and the other utilizing the pairwise co-regularization $\lambda\sum_{i\neq j}\operatorname{Tr}\left(\mathbf{U}^{(i)}\mathbf{U}^{(i)^T}\mathbf{U}^{(j)}\mathbf{U}^{(j)^T}\right)$ instead. Note that both of these formulations can be seen as relaxed versions of MC-TR-I where the embeddings from different views are forced to only be similar to each other instead of exactly equal. Specifically, the regularization terms force the columnspaces of all $\mathbf{U}^{(k)}$ to become more similar with each other as λ and λ_k take larger values. In the limit, their columnspaces become identical and the rows of $\mathbf{U}^{(k)}$ will be just orthogonally rotated versions of the rows of any other $\mathbf{U}^{(i)}$. Therefore, applying k-means, as suggested by the authors, on the rows of any $\mathbf{U}^{(k)}$ will lead to the same node clusters.

Now notice that (2.3) has the same optimal set as

$$\arg\inf_{\substack{\mathbf{U}^T\mathbf{U}=\mathbf{I}\\ \mathbf{A}\succeq\mathbf{0}, ||\mathbf{A}||=1}} ||\boldsymbol{\mathcal{S}} - [\![\mathbf{U}, \mathbf{U}, \mathbf{AB}]\!]|| \tag{2.5}$$

where $\boldsymbol{\mathcal{S}}_{::k} := \mathbf{S}^{(k)}$, $\mathbf{A} := \mathbf{a}$ and $\mathbf{B} := \mathbf{1}^T$ (see Appendix A.2.1 for proof). Similarly, we can show that MC-TR-I has the same optimal set as $\inf_{\mathbf{U}^T\mathbf{U}=\mathbf{I}} ||\boldsymbol{\mathcal{S}} - [\![\mathbf{U}, \mathbf{U}, \mathbf{AB}]\!]||$ where $\boldsymbol{\mathcal{S}}_{::k} := \mathbf{S}^{(k)}$, $\mathbf{A} := \mathbf{1}$ and $\mathbf{B} := \mathbf{1}^T$. Lastly, note that due to the close connection of (2.4) and its variant with MC-TR-I, one can argue that the same reformulation captures the essence of these problems as well.

3 Proposed Method.

In this section, we develop GenClus, a principled graph clustering method which can be viewed as an instance of this unifying framework. GenClus also generalizes spectral clustering [20] to multi-view graphs with multiple view structures, and, as we will show in Section 3.2, it is closely associated with k-means as well.

Based on our discussion in Section 2.4, we will consider the modified data tensor $\boldsymbol{\mathcal{Y}}$, where $\boldsymbol{\mathcal{Y}}_{::k} := \mathbf{D}^{(k)}\boldsymbol{\mathcal{X}}_{::k}\mathbf{D}^{(k)}$ and $\mathbf{D}^{(k)} := \operatorname{diag}\left(1/\sqrt{\sum_{i=1}^{I}\mathbf{X}_{:ik}}\right)$. Then, we observe that in the special case of a single-view graph, where $K = M = 1$ and R is the true number of clusters, we can use properties of the best positive semi-definite approximation of a matrix (see Appendix A.2.2 for details), along with the fact that we expect $\boldsymbol{\mathcal{Y}}$ to have R eigenvalues close to 1, to show that

$$\operatorname*{arg\,inf}_{\mathbf{U}^T\mathbf{U}=\mathbf{I}} \left(\inf_{a\geq 0, \mathbf{b}\geq \mathbf{0}} \left|\left| \boldsymbol{\mathcal{Y}} - a\mathbf{U}\operatorname{diag}(\mathbf{b})\mathbf{U}^T \right|\right| \right) = \operatorname*{arg\,inf}_{\mathbf{U}^T\mathbf{U}=\mathbf{I}} \left|\left| \boldsymbol{\mathcal{Y}} - \mathbf{U}\mathbf{U}^T \right|\right|. \tag{3.6}$$

Notice that the constraints on a and $\mathbf{b}$ of the left-hand side problem are more relaxed than the constraints on the implicit a and $\mathbf{b}$ of the right-hand side problem. Therefore, this result can be particularly useful in designing solvers that are less prone to bad local optima. Also, to understand why we opted for non-negativity constraints, first note that allowing a to be negative may make (3.6) to not hold for any arbitrary $\boldsymbol{\mathcal{Y}}$ since the left-hand side problem will select eigenvectors corresponding to its largest negative eigenvalues if they are of larger magnitude than its positive eigenvalues. This is in contrast to the right-hand side problem which always retrieves eigenvectors corresponding to the maximum eigenvalues. Similarly, if we allow $\mathbf{b}$ to have negative elements, then the left-hand side problem is a direct application of the Eckart-Young theorem [3] and will, therefore, select eigenvectors corresponding to the maximum magnitude eigenvalues instead.

To design a generalized version of this problem for arbitrary values of K and M, first notice that $a\mathbf{U}\operatorname{diag}(\mathbf{b})\mathbf{U}^T = [\![\mathbf{U}, \mathbf{U}, a\mathbf{b}^T]\!] = [\![\mathbf{U}, \mathbf{U}, \mathbf{AB}]\!]$, where $\mathbf{A} := a$ and $\mathbf{B} := \mathbf{b}^T$. Therefore, in the general case where $\boldsymbol{\mathcal{Y}}$ has multiple views with multiple view structures, we propose formulating the problem as

$$\inf_{\substack{\mathbf{U}^{(m)^T}\mathbf{U}^{(m)}=\mathbf{I} \\ \mathbf{A},\mathbf{B}^T\in\mathcal{I}}} ||\boldsymbol{\mathcal{Y}} - [\![\mathbf{U}, \mathbf{U}, \mathbf{AB}]\!]||^2 \tag{3.7}$$

where $\mathcal{I}$ is the set of all matrices whose rows contain only a single non-zero element each, and the columns of $\mathbf{U}^{(m)}$ are defined to be the subset of columns of $\mathbf{U}$ that correspond to the positions of the non-zero elements of $\mathbf{B}_{m:}$. Note that the constraint $\mathbf{A}, \mathbf{B}^T \in \mathcal{I}$ is particularly important for two reasons. First, it leads to a $\left\{\mathbf{U}^{(m)}\right\}_{m=1}^{M}$ that forms a partition of the columns of $\mathbf{U}$, which in turn implies that we get a distinct model for each view structure. Second, as we will show next, it enables $\mathbf{U}$ and $\mathbf{B}$ to be calculated simultaneously, which can prove beneficial in terms of designing a solver that is both faster and less prone to bad local optima.

Lastly, note that for completeness, in Section 3.1 where we will derive optimization steps for (3.7), we will also derive optimization steps for further constrained versions of it where the non-zero elements of $\mathbf{A}$ and $\mathbf{B}$ can be either all-ones or non-negative. That said, we still recommend using the non-negativity constraint for both $\mathbf{A}$ and $\mathbf{B}$ as the default option. To understand the reasoning behind this choice, first notice that (3.7) can properly generalize the left-hand side and right-hand side of (3.6), only when the non-negativity and all-ones constraint is imposed, respectively. That is, without these additional constraints, (3.7) does not necessarily lead to generalization of spectral clustering. Additionally, notice that the non-negativity constraint may be preferable to the all-ones constraint because it leads to an optimization problem that is more relaxed, and, therefore, the resulting solver can be expected to be less prone to bad local optima.

3.1 Optimization Steps.

We propose solving (3.7) in a block coordinate descent fashion by alternatingly updating $\mathbf{A}$, and then $\mathbf{U}$ and $\mathbf{B}$ simultaneously. Note that, although in this work we do not provide arguments regarding convergence, our optimization scheme is guaranteed to monotonically improve the objective function after each update of $\mathbf{A}$, $\mathbf{B}$ and $\mathbf{U}$.

3.1.1 Steps for A.

By observing that $||\boldsymbol{\mathcal{Y}} - [\![\mathbf{U}, \mathbf{U}, \mathbf{AB}]\!]|| = \left|\left|\mathbf{Y}_{(3)} - \mathbf{AB}\left(\mathbf{U}\odot\mathbf{U}\right)^T\right|\right|$, we can see that for fixed $\mathbf{B}$ and $\mathbf{U}$, the optimal $\mathbf{A}$ is given by the solution of a constrained linear least squares problem. Specifically, below we discuss three different constraints for the non-zero elements of $\mathbf{A}$:

All-ones A. Here we observe that the k-th row of $\mathbf{A}$ is optimal when it is an indi-

cator vector with its i-th element equal to 1 where $i = \arg\inf_i \left|\left|\left[\mathbf{Y}_{(3)}\right]_{k:} - \left[\mathbf{B}\left(\mathbf{U}\odot\mathbf{U}\right)^T\right]_{i:}\right|\right|$, or equivalently by using tensor notation $i = \arg\inf_i \left|\left|\boldsymbol{\mathcal{Y}}_{::k} - \mathbf{U}\,\mathrm{diag}\left(\mathbf{B}_{i:}\right)\mathbf{U}^T\right|\right|$.

Unconstrained A. Here, the only non-zero element of $\mathbf{A}_{k:}$ will be at position m if, and only if, among all lines defined by each row of $\mathbf{B}\left(\mathbf{U}\odot\mathbf{U}\right)^T$, the one defined by the m-th row is the closest one to the k-th row of $\mathbf{Y}_{(3)}$. Thus, $m = \arg\sup_m \left|\sum_{i,j}\boldsymbol{\mathcal{Y}}_{ijk}\mathbf{Q}^{(m)}_{ij}\right| / \left|\left|\mathbf{Q}^{(m)}\right|\right|$ where $\mathbf{Q}^{(m)} := \mathbf{U}\,\mathrm{diag}\left(\mathbf{B}_{m:}\right)\mathbf{U}^T$. In turn, we have $\mathbf{A}_{km} = \arg\inf_\alpha \left|\left|\boldsymbol{\mathcal{Y}}_{::k} - \alpha\mathbf{Q}^{(m)}\right|\right|$ which can be calculated in closed form as $\sum_{i,j}\boldsymbol{\mathcal{Y}}_{ijk}\mathbf{Q}^{(m)}_{ij} / \left|\left|\mathbf{Q}^{(m)}\right|\right|^2$.

Non-negative A. Here, the only non-zero element of $\mathbf{A}_{k:}$ can be $\mathbf{A}_{km}$ if, and only if, $\left[\mathbf{Y}_{(3)}\right]_{k:}$ forms the smallest angle with the m-th row of $\mathbf{B}\left(\mathbf{U}\odot\mathbf{U}\right)^T$. Thus, we have $m = \arg\sup_m \sum_{i,j}\boldsymbol{\mathcal{Y}}_{ijk}\mathbf{Q}^{(m)}_{ij} / \left|\left|\mathbf{Q}^{(m)}\right|\right|$ where $\mathbf{Q}^{(m)} := \mathbf{U}\,\mathrm{diag}\left(\mathbf{B}_{m:}\right)\mathbf{U}^T$. In turn, $\mathbf{A}_{km}$ is calculated as in the unconstrained case, unless $\sum_{i,j}\boldsymbol{\mathcal{Y}}_{ijk}\mathbf{Q}^{(m)}_{ij}$ is negative, in which case $\mathbf{A}_{km} = 0$. Note that $\mathbf{A}_{km} = 0$ if, and only if, none of the rows of $\mathbf{B}\left(\mathbf{U}\odot\mathbf{U}\right)^T$ forms an acute angle with $\left[\mathbf{Y}_{(3)}\right]_{k:}$.

3.1.2 Steps for U and B. Here, first notice that (3.7) can be reexpressed as

(3.8)
$$\underset{\substack{\mathbf{U}^{(m)^T}\mathbf{U}^{(m)}=\mathbf{I}\\ \mathbf{A},\mathbf{B}^T\in\mathcal{I}}}{\arg\inf} \sum_{m=1}^{M}\sum_{k=1}^{K}\left|\left|\boldsymbol{\mathcal{Y}}_{::k} - \mathbf{A}_{km}\mathbf{U}^{(m)}\mathbf{D}^{(m)}_{\mathbf{B}}\mathbf{U}^{(m)^T}\right|\right|^2$$

where $\mathbf{D}^{(m)}_{\mathbf{B}}$ is defined to be a diagonal matrix containing only the non-zero elements of $\mathbf{B}_{m:}$. We also define $\mathcal{S}_m$ as the set of indices of the views assigned to view cluster m. In turn, for a fixed $\mathbf{A}$, we distinguish three types of constraints for the non-zero elements of $\mathbf{B}$:

All-ones B. In this case, we can show that for a fixed $\mathbf{A}$ (3.8) can be reexpressed as

$$\underset{\substack{\mathbf{U}^{(m)^T}\mathbf{U}^{(m)}=\mathbf{I}\\ \mathbf{B}^T\in\mathcal{I}}}{\arg\sup} \sum_{m=1}^{M}\mathrm{Tr}\left(\mathbf{U}^{(m)^T}\mathbf{Z}^{(m)}\mathbf{U}^{(m)}\right) \tag{3.9}$$

where $\mathbf{Z}^{(m)} := \sum_{k=1}^{K} 2\mathbf{A}_{km}\boldsymbol{\mathcal{Y}}_{::k} - \left|\left|\mathbf{A}_{:m}\right|\right|^2\mathbf{I}$ (see Appendix A.2.3 for detailed derivation). Therefore, we can see that, for a fixed $\mathbf{B}$ and for all m, the optimal $\mathbf{U}^{(m)}$ has columns the eigenvectors of $\mathbf{Z}^{(m)}$ corresponding to its largest eigenvalues, which in turn implies that the corresponding summand in the objective function of (3.9) will be the sum of these eigenvalues. Thus, if we define $\mathcal{E}_m$ to be a set containing the eigenvalues of $\mathbf{Z}^{(m)}$, and $\mathcal{E}_{max}$ to be a set containing the largest R elements of $\cup_{m=1}^{M}\mathcal{E}_m$, then we can see that the optimal value of the objective function in (3.9) cannot be greater than the sum of the elements of $\mathcal{E}_{max}$. In fact, we can achieve this value if for all m we set $\mathbf{U}^{(m)}$ to have columns the eigenvectors of $\mathbf{Z}^{(m)}$ corresponding to the eigenvalues in $\mathcal{E}_m \cap \mathcal{E}_{max}$. Then, the optimal pair $(\mathbf{U},\mathbf{B})$ can be given by setting $\mathbf{U} = \left[\mathbf{U}^{(1)},\mathbf{U}^{(2)},\cdots,\mathbf{U}^{(M)}\right]$ and then deriving the optimal $\mathbf{B}$ by noticing that $\mathbf{B}_{mr}$ is non-zero if, and only if, any of the columns of $\mathbf{U}^{(m)}$ was assigned as the r-th column of $\mathbf{U}$.

Unconstrained B. In this case, for a fixed $\mathbf{A}$, (3.8) can be reexpressed as

(3.10)
$$\underset{\substack{\mathbf{U}^{(m)^T}\mathbf{U}^{(m)}=\mathbf{I}\\ \mathbf{B}^T\in\mathcal{I}}}{\arg\inf} \sum_{m=1}^{M}\left|\left|\mathbf{Z}^{(m)} - \left|\left|\mathbf{A}_{:m}\right|\right|\mathbf{U}^{(m)}\mathbf{D}^{(m)}_{\mathbf{B}}\mathbf{U}^{(m)^T}\right|\right|^2$$

where $\mathbf{Z}^{(m)} := \sum_{k=1}^{K}\mathbf{A}_{km}\boldsymbol{\mathcal{Y}}_{::k} / \left|\left|\mathbf{A}_{:m}\right|\right|$ (see Appendix A.2.4 for detailed derivation). We can now see that if we leave the non-zero elements of $\mathbf{B}$ unconstrained, then, for all m, $\mathbf{U}^{(m)}$ will have a fixed number of columns which will be optimal when they consist of the eigenvectors of $\mathbf{Z}^{(m)}$ corresponding to the eigenvalues of largest magnitude. In turn, the non-zero elements of the optimal $\mathbf{B}_{m:}$ are the same eigenvalues divided by $\left|\left|\mathbf{A}_{:m}\right|\right|$. Also, notice that the corresponding summand in the objective function of (3.10) will be the sum of squares of all the remaining eigenvalues. Therefore, if we define $\mathcal{E}_m$ to be a set containing the eigenvalues of $\mathbf{Z}^{(m)}$, and $\mathcal{E}_{max}$ to be a set containing the R elements of largest magnitude of $\cup_{m=1}^{M}\mathcal{E}_m$, then we can see that the optimal value of the objective function in (3.10) cannot be lower than the sum of the squared elements of $\cup_{m=1}^{M}\mathcal{E}_m \setminus \mathcal{E}_{max}$. In fact, we can achieve this value by setting the columns of $\mathbf{U}^{(m)}$ to be the eigenvectors of $\mathbf{Z}^{(m)}$ corresponding to the eigenvalues in $\mathcal{E}_m \cap \mathcal{E}_{max}$. Then, the optimal pair $(\mathbf{U},\mathbf{B})$ can be given by setting $\mathbf{U} = \left[\mathbf{U}^{(1)},\mathbf{U}^{(2)},\cdots,\mathbf{U}^{(M)}\right]$ and by assigning the elements of $\mathcal{E}_m \cap \mathcal{E}_{max}$ divided by $\left|\left|\mathbf{A}_{:m}\right|\right|$ as the non-zero elements of $\mathbf{B}_{m:}$.

Non-negative B. Here first notice that each of the M summands in (3.10) is minimized via the best positive semi-definite approximation of the corresponding $\mathbf{Z}^{(m)}$. In other words, we can see that the same arguments as in the unconstrained case apply, with the difference that $\mathcal{E}_m$ here is instead defined to contain the largest eigenvalues of $\mathbf{Z}^{(m)}$ after its negative eigenvalues are set equal to zero.

3.2 Model Interpretation. By looking at the updates for $\mathbf{U}$, $\mathbf{A}$ and $\mathbf{B}$ we can see that there is a very natural way of interpreting GenClus. First, from (3.9) and (3.10) we can see that, for a fixed $\mathbf{A}$, GenClus computes the node clustering for each of the M view

clusters. Specifically, the m-th node clustering is calculated, roughly speaking, by performing spectral clustering based on $\mathbf{Z}^{(m)}$ which is the weighted summation of the Laplacians of all views that belong to the m-th view cluster. Also, for fixed $\mathbf{U}$ and $\mathbf{B}$, and by noticing that $[\![\mathbf{U}, \mathbf{U}, \mathbf{AB}]\!] = [\![\mathbf{U}, \mathbf{U}, \mathbf{B}]\!] \times_3 \mathbf{A}$, we can see that each row of $\mathbf{A}$ is calculated in exactly the same fashion as the cluster assignment step of k-means or a k-lines method, depending on the type of constraint. Specifically, we can think of the frontal slices of $[\![\mathbf{U}, \mathbf{U}, \mathbf{B}]\!]$ as the centroids representing the clusters, while the k-th row of $\mathbf{A}$ can be seen as an indicator vector encoding the cluster membership of the k-th data point, $\boldsymbol{\mathcal{Y}}_{::k}$.

This observation leads us to another interesting finding. That is, if we constrain the non-zero elements of $\mathbf{A}$ and $\mathbf{B}$ to be all-ones and unconstrained, respectively, and we set $R = M \cdot I$, then GenClus becomes identical to k-means. To see this, note that in this case, for a fixed $\mathbf{A}$, there will always exist $\mathbf{U}$ and $\mathbf{B}$ in (3.10) such that, for all m, $||\mathbf{A}_{:m}|| \, \mathbf{U}^{(m)} \mathbf{D}_{\mathbf{B}}^{(m)} \mathbf{U}^{(m)^T}$ will be identical to $\mathbf{Z}^{(m)}$. Therefore, the m-th frontal slice of the optimal $[\![\mathbf{U}, \mathbf{U}, \mathbf{B}]\!]$ will be $\mathbf{Z}^{(m)} / ||\mathbf{A}_{:m}||$. In turn, this can be simplified to $\sum_{k \in \mathcal{S}_m} \boldsymbol{\mathcal{Y}}_{::k} / |\mathcal{S}_m|$, with $\mathcal{S}_m$ being the set of indices of the views assigned to the m-th view cluster. In fact, this is exactly the centroid calculation step of k-means. Also, notice that if we set $R < M \cdot I$, GenClus can be seen as a version of k-means where instead of using the usual centroids, we perform calculations based on their denoised low-rank versions $\{[\![\mathbf{U}, \mathbf{U}, \mathbf{B}]\!]_{::m}\}_{m=1}^{M}$.

3.3 Space & Time Complexity. Note that our method can readily benefit from various optimizations that leverage parallelization, sparsity, partial eigendecompositions and more appropriate sorting techniques. However, for simplicity, we have omitted such optimizations from the implementation used in all our experiments in Section 4. In this case, a straightforward implementation prioritizing time minimization over memory usage can achieve a space complexity of $O\left(MI^2 + RI + K\right)$, and, for t iterations, a time complexity of $O\left(tMI^3 + t(R + KM)I^2 + tMI \log(MI)\right)$. For more details, please refer to Appendix A.3.

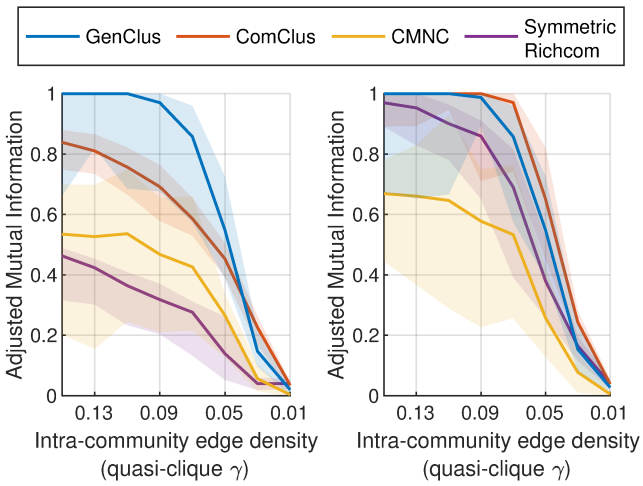
4 Experiments.

In this section, we perform an in-depth experimental exploration of the behavior of GenClus both quantitatively and qualitatively. First, in Section 4.1 we perform quantitative clustering quality comparisons with other baseline methods on artificially generated multi-view graphs with known ground truth labels. Then, in Section 4.2 we present a qualitative case study which demonstrates the ability of GenClus to generate meaningful clusterings on real-world multi-structure multi-view graphs. Lastly, in Section 4.3 we perform an experimental comparison of the time complexity of all methods. These experiments were conducted using Multi-Graph Explorer [18] whose code can be accessed at [19].

Lastly, note that in all methods discussed so far, the clustering process can be divided into four segments: data preprocessing, embedding calculation, embedding postprocessing and embedding clustering. However, embedding calculation is arguably the central novelty in both the existing methods and our proposed method. Therefore, we will consider both the clustering schemes as proposed in the original papers and enhanced versions where the best combination of the remaining segments is selected. We will call these "original methods" and "enhanced methods", respectively. For the precise details of the experiment setup, please refer to Appendix A.4.

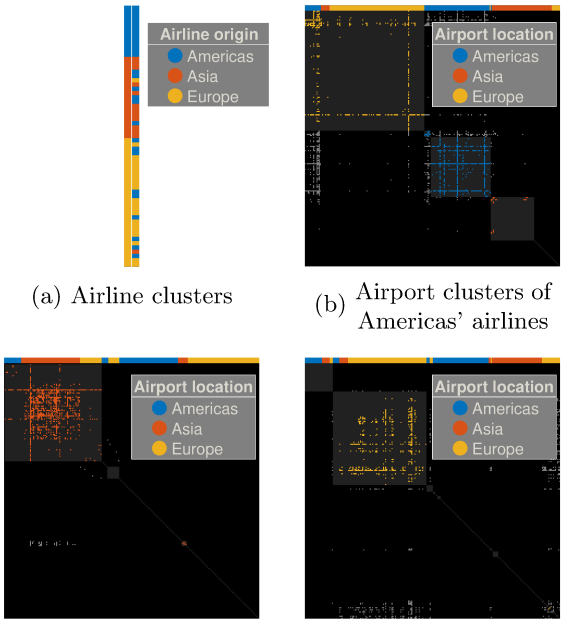
4.1 Clustering Quality on Artificial Data. We generate a directed unweighted multi-view graph with 120 nodes and 9 views, which leads to a tensor of size $120 \times 120 \times 9$. The views form 3 clusters with 3 views each, and each view cluster corresponds to 3, 2 and 2 node clusters, respectively. Specifically, the node clusters corresponding to the first view cluster contain 60, 40 and 20 nodes, respectively, while in the second view cluster they contain 100 and 20 nodes, respectively, and in the third view cluster they contain 20 and 100 nodes, respectively. All node clusters are γ-quasi-cliques [17] whose intra-community edge density, γ, takes values in the set of the 8 equally spaced values from 15% to 1%, inclusive. For a specific generated graph, all quasi-cliques have identical value of γ. Then, we randomly select 1% of all pairs of nodes, and, for each pair, we remove their connecting edge if they are connected, or introduce a new edge between them if they are not connected. For more details, please refer to Appendix A.4.3.

4.1.1 Results Analysis. First, we make comparisons of the original methods as shown in Figure 3a. Here we see that GenClus offers superior performance compared to all baselines, and, in fact, it is the only method that manages, in the median, to perfectly reconstruct the ground truth communities even with an intra-community edge density, γ, as low as 11%. ComClus is the next best method, and it also significantly outperforms CMNC and Symmetric Richcom. Also, although ComClus slightly outperforms GenClus for very low values of γ, it performs significantly worse than GenClus for higher values of γ. CMNC outperforms Symmetric Richcom in the median, but due to its high variance we cannot confidently declare it as the clear winner.



(a) Original methods (b) Enhanced methods

Figure 3: Clustering performance comparisons. Lines represent medians, while shaded areas represent 25-th and 75-th percentiles. Higher values signify better clustering quality and the maximum possible value is 1.



(a) Airline clusters (b) Airport clusters of Americas' airlines

(c) Airport clusters of Asian airlines (d) Airport clusters of European airlines

Figure 4: Clustering of airlines and airports by GenClus for the flights dataset. All colored bars are split into small pieces representing individual airports or airlines, and the colors represent labels. The left-hand bar in (a) depicts the airlines colored based on their continent of origin, while the right-hand bar shows them colored based on the view clustering produced by GenClus. (b)-(d) show the adjacency matrices of representative views from each of the airline clusters produced by GenClus, and the horizontal colored bars on top indicate the actual location of each airport. The airport clusters generated by GenClus are depicted as grey squares, which is achieved by appropriately permuting each adjacency matrix individually.

Now, we make comparisons of the enhanced methods as shown in Figure 3b. In this case, we observe that the enhanced versions of ComClus and Symmetric Richcom present a significant performance uplift. In fact, ComClus now observably tends to perform better than GenClus. At the same time, Symmetric Richcom not only became significantly better than CMNC, but its performance is now closer to the performance of GenClus. On the other hand, the enhanced versions of GenClus and CMNC barely show a performance improvement. However, note that, at least for GenClus, this is a positive outcome, since it experimentally validates our theoretical arguments from Section 3 in favor of pairing GenClus with normalized Laplacians and with non-negativity constraints for $\mathbf{A}$ and $\mathbf{B}$.

4.2 Real-World Case Study. In this case study we are using a dataset of flight routes from 2012 [15] containing flights from a large number of airlines and airports around the world. Note that, to simplify the wording of our observations, we will refer to the Americas as if they were a single continent. For additional details, please refer to Appendix A.4.4.

4.2.1 Results Analysis. Figure 4 shows the results of our proposed method on this dataset. As we can see, the clustering of views (airlines) generated by GenClus separates the majority of them based on their continent of origin. This aligns with our intuition that the flight patterns of airlines originating in the same continent are similar to each other and different from the flight patterns of airlines from different continents. Additionally, Figure 4b indicates that airlines from the Americas tend to have substantial presence on all three continents, although these flights tend to not be intercontinental. On the other hand, Figure 4c and Figure 4d indicate that airlines originating in Asia and Europe tend to fly almost exclusively within their continent of origin.

4.3 Execution Time on Artificial Data. Here we experimentally compare the time complexity of all original methods for increasing graph sizes and embedding sizes. Note that since in this experiment our interest is in evaluating the time complexity of the

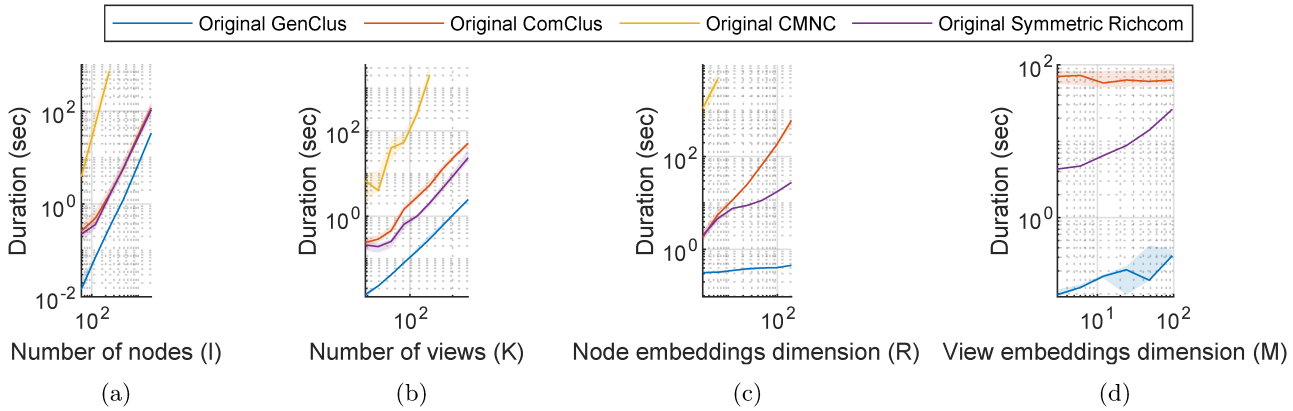


Figure 5: Execution time comparisons of original methods for varying graph sizes and embedding sizes. Log-log plots are used for all experiments, with each plot having powers of ten in its vertical axis scaled equally to powers of ten in its horizontal axis. Lines represent medians, while shaded areas represent 25-th and 75-th percentiles.

embedding generation of each method, we will omit the normalization and the clustering of the embeddings from our measurements. For the complete details of the experiment setup, please refer to Appendix A.4.5.

4.3.1 Results Analysis. All results are presented in Figure 5. However, note that, due to out-of-memory errors, only partial results are presented for CMNC.

In Figure 5a, we see that all methods present an approximately polynomial time complexity with regards to the number of nodes, and that ComClus performs very similarly to Symmetric Richcom. Also, while GenClus exhibits a growth rate similar to ComClus and Symmetric Richcom, it is about 3 to 4 times faster than both of them. CMNC performs the worst among all methods while also exhibiting a higher growth rate.

Based on Figure 5b, we reach similar conclusions with regards to the number of views as well. The main difference here is that GenClus exhibits even better performance, by being about 7 to 9 times faster than Symmetric Richcom, and about 18 to 21 times faster than ComClus. Also, Symmetric Richcom is now about 2.5 to 3 times faster than ComClus.

In Figure 5c, we notice that although all methods exhibit an approximately polynomial time complexity with respect to the dimension of node embeddings, they seem to have very different growth rates from each other. CMNC seems to again be having the worse performance, but due to out-of-memory errors, it is hard to make strong claims here. Also, ComClus exhibits a much worse growth rate compared to Symmetric Richcom. GenClus presents once again the best performance, and in fact it exhibits a near-linear time complexity. That said, we suspect that this may be misleading, as our implementation of GenClus often unnecessarily computes full eigendecompositions, even though typically only a portion of the eigenpairs is required.

Lastly, in Figure 5d, we see that GenClus again significantly outperforms the baselines, while CMNC produced out-of-memory errors for the entirety of this experiment. An advantage of ComClus is that it exhibits a linear time complexity in this case, while GenClus and Symmetric Richcom seem to still have an approximately polynomial time complexity.

5 Conclusions.

In this work, we devised a unifying framework for multiple graph clustering methods, which is capable of modeling data as complex as multi-view graphs with multiple view structures. Then, we proposed GenClus, a novel instance of this framework, which also aims to have principled foundations by virtue of being a generalization of the highly successful spectral clustering. Additionally, we conducted in-depth experiments on artificial data, in which we controlled for every aspect of the clustering workflow. These experiments demonstrated that GenClus can have similar or better clustering performance than the baselines, while also being more computationally efficient. Lastly, we evaluated GenClus on a real-world multi-view graph with multiple view structures, which showed that it can effectively model such complex datasets and uncover meaningful insights.

Acknowledgements

This research was supported by the National Science Foundation under CAREER grant no. IIS 2046086, grant no. IIS 1901379, and CREST Center for Multidisciplinary Research Excellence in Cyber-Physical Infrastructure Systems (MECIS) grant no. 2112650.

References

[1] J. Douglas Carroll and Jih-Jie Chang, *Analysis of individual differences in multidimensional scaling via an N-way generalization of "Eckart-Young" decomposition*, in Psychometrika 35.3 (1970), pp. 283–319.

[2] Zitai Chen et al., *Tensor decomposition for multilayer networks clustering*, in Proceedings of the AAAI Conference on Artificial Intelligence, vol. 33, 01, 2019, pp. 3371–3378.

[3] Carl Eckart and Gale Young, *The approximation of one matrix by another of lower rank*, in Psychometrika 1.3 (1936), pp. 211–218.

[4] Ekta Gujral, Ravdeep Pasricha, and Evangelos Papalexakis, *Beyond rank-1: Discovering rich community structure in multi-aspect graphs*, in Proceedings of The Web Conference 2020, 2020, pp. 452–462.

[5] Richard A. Harshman, *Foundations of the PARAFAC procedure: Models and conditions for an "explanatory" multimodal factor analysis*, tech. rep. 16, UCLA Working Papers in Phonetics, 1970, pp. 1–84.

[6] Stefan Klus and Natasa Djurdjevac Conrad, *Koopman-based spectral clustering of directed and time-evolving graphs*, in arXiv preprint arXiv:2204.02951 (2022).

[7] Tamara Gibson Kolda, *Multilinear operators for higher-order decompositions.* Tech. rep., Sandia National Laboratories (SNL), Albuquerque, NM, and Livermore, CA (United States), 2006.

[8] Abhishek Kumar, Piyush Rai, and Hal Daume, *Co-regularized multi-view spectral clustering*, in Advances in Neural Information Processing Systems, ed. by J. Shawe-Taylor et al., vol. 24, Curran Associates, Inc., 2011.

[9] Xinhai Liu et al., *Multiview partitioning via tensor methods*, in IEEE Transactions on Knowledge and Data Engineering 25.5 (2012), pp. 1056–1069.

[10] Stuart Lloyd, *Least squares quantization in PCM*, in IEEE transactions on information theory 28.2 (1982), pp. 129–137.

[11] Matteo Magnani et al., *Community detection in multiplex networks*, in ACM Computing Surveys (CSUR) 54.3 (2021), pp. 1–35.

[12] Andrew Ng, Michael Jordan, and Yair Weiss, *On spectral clustering: Analysis and an algorithm*, in Advances in neural information processing systems 14 (2001).

[13] Jingchao Ni et al., *ComClus: A self-grouping framework for multi-network clustering*, in IEEE transactions on knowledge and data engineering 30.3 (2017), pp. 435–448.

[14] Maximilian Nickel et al., *A Review of Relational Machine Learning for Knowledge Graphs*, in Proceedings of the IEEE 104.1 (2016), pp. 11–33.

[15] OpenFlights, *OpenFlights Data*, Accessed: May 16, 2024, URL: `https://openflights.org/data`.

[16] Evangelos E Papalexakis, Leman Akoglu, and Dino Ienco, *Do more views of a graph help? community detection and clustering in multi-graphs*, in Proceedings of the 16th International Conference on Information Fusion, IEEE, 2013, pp. 899–905.

[17] Jeffrey Pattillo et al., *On the maximum quasi-clique problem*, in Discrete Applied Mathematics 161.1-2 (2013), pp. 244–257.

[18] Yorgos Tsitsikas and Evangelos E. Papalexakis, *Multi-Graph Explorer: A framework for Advanced Multi-Graph Analysis and Method Development*, in Proceedings of the 33rd ACM International Conference on Information and Knowledge Management, 2024.

[19] Yorgos Tsitsikas and Evangelos E. Papalexakis, *Multi-Graph Explorer: A framework for Advanced Multi-Graph Analysis and Method Development*, `https://github.com/gtsitsik/multi-graph-explorer`, GitHub repository, 2024.

[20] Ulrike Von Luxburg, *A tutorial on spectral clustering*, in Statistics and computing 17.4 (2007), pp. 395–416.

[21] Weiqiong Zhong et al., *The evolution of communities in the international oil trade network*, in Physica A: Statistical Mechanics and its Applications 413 (2014), pp. 42–52.

[22] Dengyong Zhou and Christopher JC Burges, *Spectral clustering and transductive learning with multiple views*, in Proceedings of the 24th international conference on Machine learning, 2007, pp. 1159–1166.

[23] Dengyong Zhou, Jiayuan Huang, and Bernhard Schölkopf, *Learning from labeled and unlabeled data on a directed graph*, in Proceedings of the 22nd international conference on Machine learning, 2005, pp. 1036–1043.

A Appendix.

A.1 Related Work.

A.1.1 ComClus. ComClus [12] operates on tensors whose frontal slices are symmetric adjacency matrices with non-negative elements. Specifically, it models the k-th view by approximating $\mathbf{X}^{(k)}$ as $\mathbf{O}^{(k)}\mathbf{U}\mathbf{D}_{\mathbf{W}}^{(k)}(\mathbf{O}^{(k)}\mathbf{U})^T$, where $\mathbf{U}$ and $\mathbf{W}$ are factor matrices of size $I \times R$ and $K \times R$, respectively, and $\mathbf{D}_{\mathbf{W}}^{(k)}$ is defined as $\operatorname{diag}(\mathbf{W}_{k:})$. Also, each $\mathbf{O}^{(k)}$ is a predefined indicator matrix that accounts for the fact that a node that is shared between different views may be represented by different rows and columns in the corresponding adjacency matrices. ComClus additionally defines factor matrices $\mathbf{A}$ and $\mathbf{B}$ of sizes $K \times M$ and $M \times R$, respectively, which are then used to model $\mathbf{W}$ as $\mathbf{AB}$, where $\mathbf{A}$ is constrained to be an indicator matrix, and the rows of $\mathbf{B}$ are the latent representations of the views clusters. Also, for computational tractability reasons the authors relax the constraint of $\mathbf{A}$ and impose sparsity and non-negativity on $\mathbf{U}$, $\mathbf{A}$ and $\mathbf{B}$, while $\mathbf{W}$ is constrained to be non-negative. Then, the authors calculate this model via the following optimization problem:

(A.1)
$$\inf_{\mathbf{U},\mathbf{W},\mathbf{A},\mathbf{B}\geq 0} \sum_{k=1}^{K} \left|\left|\mathbf{X}^{(k)} - \mathbf{O}^{(k)}\mathbf{U}\mathbf{D}_{\mathbf{W}}^{(k)}(\mathbf{O}^{(k)}\mathbf{U})^T\right|\right|^2 + r(\mathbf{U},\mathbf{W},\mathbf{A},\mathbf{B})$$

where $r(\mathbf{U},\mathbf{W},\mathbf{A},\mathbf{B}) := \beta||\mathbf{W} - \mathbf{AB}||^2 + \rho(||\mathbf{U}||_1 + ||\mathbf{A}||_1 + ||\mathbf{B}||_1)$, and β and ρ are penalty parameters that need to be defined by the user. Each factor matrix is updated individually via multiplicative updates in a block coordinate descent fashion [19], until the value of the objective function stops improving. Lastly, the authors assign the m-th view to the n-th cluster when the n-th element of the m-th row of $\mathbf{A}$ has the largest magnitude among all elements of that row. Then, they consider the embeddings of the nodes of the n-th calculated views cluster to be the rows of $\mathbf{U}\operatorname{diag}(\mathbf{B}_{n:})$ and cluster the nodes in the same way.

Note that the original formulation of ComClus includes two additional terms. Specifically, one term forces the latent representations of two views to be more orthogonal to each other as the number of their mutual nodes decreases, while the other term enables the user to perform semi-supervised learning when there is additional available information about how the various views relate to each other. While in this work we omit these terms, note that the effect of the term that imposes orthogonality can be achieved implicitly in a different way. That is, instead of considering an edge between two unshared nodes as unknown, we can consider it as known with weight zero. In this way, the larger the number of unshared nodes between two views is, the larger the number of elements which are non-zero in only one of the corresponding adjacency matrices of the networks will be. This implies that these adjacency matrices will tend to be orthogonal to each other, a property which will tend to hold for their latent representations as well. In fact, the original ComClus formulation can be interpreted as operating under the open world assumption [13], while our modification can be seen as operating under the closed world assumption.

A.1.2 Methods Based on Block Term Decomposition.

Richcom. Richcom [6] applies a rank-$(L_m, L_m, 1)$ terms decomposition [3] on the data tensor and operates on tensors whose frontal slices can be arbitrary adjacency matrices with non-negative elements. Specifically, it considers a tensor $\boldsymbol{\mathcal{X}}$ of size $I \times J \times K$ and models the graph by approximating $\boldsymbol{\mathcal{X}}$ as $\sum_{m=1}^{M}\left(\mathbf{U}^{(m)}\mathbf{V}^{(m)^T}\right) \times_3 \mathbf{a}^{(m)}$, where $\mathbf{U}^{(m)}$, $\mathbf{V}^{(m)}$ and $\mathbf{a}^{(m)}$ are factors of sizes $I \times R_m$, $J \times R_m$ and K, respectively. The authors implicitly assume that the ordering of the nodes is identical for all views, and, therefore, no permutations similar to these of ComClus are required. Also, note that all adjacency matrices need to be of size $I \times J$, which can be interepreted as either that Richcom only works when all nodes exist in all views, or that it makes the closed world assumption. Note that all factor matrices are constrained to be sparse and non-negative. Then, the authors calculate their model via the following optimization problem:

(A.2)
$$\inf_{\substack{\mathbf{U}^{(m)},\mathbf{V}^{(m)},\\ \mathbf{a}^{(m)}}} \left|\left|\boldsymbol{\mathcal{X}} - \sum_{m=1}^{M}\left(\mathbf{U}^{(m)}\mathbf{V}^{(m)^T}\right) \times_3 \mathbf{a}^{(m)}\right|\right|^2 + \sum_{m=1}^{M} r\left(\mathbf{U}^{(m)},\mathbf{V}^{(m)},\mathbf{a}^{(m)}\right)$$

where $r\left(\mathbf{U}^{(m)},\mathbf{V}^{(m)},\mathbf{a}^{(m)}\right)$ encodes the sparsity and non-negativity constraints. They solve (A.2) using the AO-ADMM framework [7] which solves for all $\mathbf{U}^{(m)}$, $\mathbf{V}^{(m)}$ and $\mathbf{a}^{(m)}$ individually in an alternating fashion via the alternating direction method of multipliers [1]. Lastly, the authors assign the k-th view to the m-th views cluster if $\mathbf{a}_k^m \geq \mathbf{a}_k^n$ for all n, and then consider $\mathbf{U}^{(m)}\mathbf{V}^{(m)^T}$ as an adjacency matrix representing the m-th views structure and obtain the corresponding nodes clustering by extracting its weakly connected

components after proper thresholding.

Centroid-based Multilayer Network Clustering (CMNC). CMNC [2] also applies a rank-$(L_m, L_m, 1)$ terms decomposition and its model is very similar to that of Richcom with their only differences being that for CMNC $\mathbf{U}^{(m)} = \mathbf{V}^{(m)}$ for all m, $\sum_{m=1}^{M} \mathbf{a}_i^{(m)} = 1$ for all i, and no sparsity is imposed. It is also different in the way the optimization problem was solved which was reformulated into an unconstrained one by introducing two differentiable operators, and then a trust region optimization method was applied [14]. Another difference is that CMNC preprocesses each view of the data tensor as $\mathbf{D}^{(k)^{-\frac{1}{2}}}\mathbf{X}^{(k)}\mathbf{D}^{(k)^{-\frac{1}{2}}}$ where $\mathbf{D}^{(k)} := \text{diag}\left(\sum_{i=1}^{I} \mathbf{X}_{:i}^{(k)}\right)$ inspired by the spectral clustering paradigm [18]. Lastly, another difference is that the nodes clusters of the m-th view structure are formed by assigning the i-th node to the cluster corresponding to the largest element of the i-th row of $\mathbf{U}^{(m)}$.

A.1.3 Spectral Clustering. Spectral clustering [11], [18], [21], [8] has seen great success and developments in the past decades thanks to its strong theoretical foundations and its ability to discover clusters of arbitrary shape. In fact, these developments have led to generalized versions for multi-view graphs [10], [20].

Consider an adjacency matrix $\mathbf{X}$ of an arbitrary undirected graph with non-negative weights. If we define $\mathbf{D} := \text{diag}\left(\sum_{i=1}^{I} \mathbf{X}_{:i}\right)$ then $\mathbf{L} := \mathbf{D} - \mathbf{X}$ is called the Laplacian of $\mathbf{X}$. It can be shown [18] that $\mathbf{L}$ is positive semi-definite and that the number of connected components of the graph is equal to the multiplicity of the smallest eigenvalue of $\mathbf{L}$, which is always 0. It can also be shown [11] that if $\mathbf{U}$ is a matrix whose columns form a basis for the eigenspace corresponding to the smallest eigenvalue of $\mathbf{L}$, then two rows of $\mathbf{U}$ are collinear if the corresponding nodes belong to the same connected component, and orthogonal to each other if the corresponding nodes belong to different components. Spectral clustering algorithms then use these properties to cluster the rows of $\mathbf{U}$ and identify the communities of a graph in a principled manner. Additionally, $\mathbf{L}_{sym} := \mathbf{D}^{-\frac{1}{2}}\mathbf{L}\mathbf{D}^{-\frac{1}{2}}$ is called the normalized Laplacian and inherits all the aforementioned nice properties of $\mathbf{L}$. The usefulness of $\mathbf{L}_{sym}$ is usually justified in the literature by associating it to a relaxation of the n-cut problem [18]. Lastly, note that the eigenspace of $\mathbf{L}_{sym}$ corresponding to an eigenvalue of 0 is identical to the eigenspace of $\mathbf{S} := \mathbf{I} - \mathbf{L}_{sym} = \mathbf{D}^{-\frac{1}{2}}\mathbf{X}\mathbf{D}^{-\frac{1}{2}}$ corresponding to its maximum eigenvalue which is 1.

Note there is a more direct and intuitive justification for choosing $\mathbf{L}_{sym}$ over $\mathbf{L}$. Specifically, notice that since $\mathbf{L}_{sym}$ is positive semi-definite, the eigenvalues of $\mathbf{S} := \mathbf{I} - \mathbf{L}_{sym} = \mathbf{D}^{-\frac{1}{2}}\mathbf{X}\mathbf{D}^{-\frac{1}{2}}$ will all be less than or equal to 1, and since all elements of $\mathbf{S}$ are non-negative, it can in turn be shown that its smallest eigenvalue will also be greater than or equal to -1 . Therefore, the eigenvalues of $\mathbf{L}_{sym}$ will be bounded between 0 and 2. This property can be especially useful when the different communities of the graph are not completely disconnected from each other, in which case we evaluate the number of communities to be equal to the number of eigenvalues of $\mathbf{L}_{sym}$ that are only approximately 0, or the number of eigenvalues of $\mathbf{S}$ that are only approximately 1. Another reason is that when communities are completely disconnected from each other, then $\mathbf{L}_{sym}$ can be expressed as a block-diagonal matrix where each block corresponds to a different community. This implies that the set of the eigenvalues of $\mathbf{L}_{sym}$ is the union of the eigenvalues of its blocks, and, therefore, the maximum eigenvalue for all communities will be 2. In turn, this implies that when communities are not entirely disconnected from each other, the decision of whether an eigenvalue is close enough to 0 does not have to involve the size of the corresponding community.

A.1.4 Multi-View Spectral Clustering. When a graph has multiple views such that all views are assumed to share a common underlying structure, i.e. $M = 1$, then one of the multi-view spectral clustering models proposed in [10], [20] can be applied. In our work, we will be particularly interested in some of the models proposed in [10]. Specifically, we will study the MC-TR-I model,

$$\sup_{\mathbf{U}^T\mathbf{U}=\mathbf{I}} \sum_{k=1}^{K} \text{Tr}\left(\mathbf{U}^T\mathbf{S}^{(k)}\mathbf{U}\right), \tag{A.3}$$

which aims to perform spectral clustering jointly on all views in a way that the embeddings for all views are identical to each other. We will also consider its weighted variant,

$$\sup_{\substack{\mathbf{U}^T\mathbf{U}=\mathbf{I} \\ \mathbf{a}\succeq\mathbf{0}, ||\mathbf{a}||=1}} \sum_{k=1}^{K} \text{Tr}\left(\mathbf{U}^T\mathbf{a}_k\mathbf{S}^{(k)}\mathbf{U}\right), \tag{A.4}$$

which assigns a different weight to each view. To calculate these models, the authors proposed the MC-TR-I-EVD and MC-TR-I-EVDIT algorithms, respectively.

Then, in [9] the authors propose two forms of co-regularized spectral clustering with the first form consisting of a pairwise co-regularization as

$$\sup_{\mathbf{U}^{(k)^T}\mathbf{U}^{(k)}=\mathbf{I}} \sum_{k=1}^{K} \operatorname{Tr}\left(\mathbf{U}^{(k)^T}\mathbf{S}^{(k)}\mathbf{U}^{(k)}\right) + \lambda \sum_{i\neq j} \operatorname{Tr}\left(\mathbf{U}^{(i)}\mathbf{U}^{(i)^T}\mathbf{U}^{(j)}\mathbf{U}^{(j)^T}\right) \tag{A.5}$$

and the second one consisting of a centroid-based co-regularization as

$$\sup_{\substack{\mathbf{U}^{(k)^T}\mathbf{U}^{(k)}=\mathbf{I} \\ \mathbf{U}^{*T}\mathbf{U}^{*}=\mathbf{I}}} \sum_{k=1}^{K} \operatorname{Tr}\left(\mathbf{U}^{(k)^T}\mathbf{S}^{(k)}\mathbf{U}^{(k)}\right) + \lambda_k \operatorname{Tr}\left(\mathbf{U}^{(k)}\mathbf{U}^{(k)^T}\mathbf{U}^{*}\mathbf{U}^{*T}\right). \tag{A.6}$$

Note that both of these formulations can be seen as relaxed versions of (A.3) where instead the embeddings from different views are forced to only be similar to each other instead of exactly equal. Specifically, the regularization terms force the columnspaces of all $\mathbf{U}^{(k)}$ to become more similar with each other as λ and λ_k take larger values. In the limit, their columnspaces become identical and the rows of $\mathbf{U}^{(k)}$ will be just orthogonally rotated versions of the rows of any other $\mathbf{U}^{(i)}$. Therefore, applying k-means, which is the clustering method suggested by the authors, on the rows of any $\mathbf{U}^{(k)}$ will lead to the same node clusters.

Now notice that we can show that (A.4) has the same optimal set as

$$\operatorname*{arg\,inf}_{\substack{\mathbf{U}^T\mathbf{U}=\mathbf{I} \\ \mathbf{A}\succeq\mathbf{0},||\mathbf{A}||=1}} ||\boldsymbol{\mathcal{S}} - [\![\mathbf{U},\mathbf{U},\mathbf{AB}]\!]|| \tag{A.7}$$

where $\boldsymbol{\mathcal{S}}_{::k} := \mathbf{S}^{(k)}$, $\mathbf{A} := \mathbf{a}$ and $\mathbf{B} := \mathbf{1}^T$ (see Appendix A.2.1 for proof). Similarly, we can show that (A.3) has the same optimal set as $\inf_{\mathbf{U}^T\mathbf{U}=\mathbf{I}} ||\boldsymbol{\mathcal{S}} - [\![\mathbf{U},\mathbf{U},\mathbf{AB}]\!]||$ where $\boldsymbol{\mathcal{S}}_{::k} := \mathbf{S}^{(k)}$, $\mathbf{A} := \mathbf{1}$ and $\mathbf{B} := \mathbf{1}^T$. Lastly, note that due to the close connection of (A.5) and (A.6) with (A.3) one can argue that the same reformulation captures the essence of these problems as well, despite the fact that they cannot be reformulated exactly like that from a strictly mathematical point of view.

A.2 Proofs

A.2.1 Derivation of (2.5).

$$\operatorname*{arg\,sup}_{\substack{\mathbf{U}^T\mathbf{U}=\mathbf{I} \\ \mathbf{a}\succeq\mathbf{0},||\mathbf{a}||=1}} \operatorname{Tr}\left(\mathbf{U}^T\sum_{k=1}^{K}\mathbf{a}_k\mathbf{S}^{(k)}\mathbf{U}\right) =$$

$$\operatorname*{arg\,inf}_{\substack{\mathbf{U}^T\mathbf{U}=\mathbf{I} \\ \mathbf{a}\succeq\mathbf{0},||\mathbf{a}||=1}} -2\operatorname{Tr}\left(\mathbf{U}^T\sum_{k=1}^{K}\mathbf{a}_k\mathbf{S}^{(k)}\mathbf{U}\right) + R =$$

$$\operatorname*{arg\,inf}_{\substack{\mathbf{U}^T\mathbf{U}=\mathbf{I} \\ \mathbf{a}\succeq\mathbf{0},||\mathbf{a}||=1}} \sum_{k=1}^{K} -2\operatorname{Tr}\left(\mathbf{U}^T\mathbf{a}_k\mathbf{S}^{(k)}\mathbf{U}\right) + {\mathbf{a}_k}^2 R =$$

$$\operatorname*{arg\,inf}_{\substack{\mathbf{U}^T\mathbf{U}=\mathbf{I} \\ \mathbf{a}\succeq\mathbf{0},||\mathbf{a}||=1}} \sum_{k=1}^{K} \operatorname{Tr}\left(-2\mathbf{a}_k\mathbf{U}\mathbf{U}^T\mathbf{S}^{(k)} + \left(\mathbf{a}_k\mathbf{U}\mathbf{U}^T\right)^2\right) =$$

$$\operatorname*{arg\,inf}_{\substack{\mathbf{U}^T\mathbf{U}=\mathbf{I} \\ \mathbf{a}\succeq\mathbf{0},||\mathbf{a}||=1}} \sum_{k=1}^{K} \left|\left|\mathbf{S}^{(k)} - \mathbf{a}_k\mathbf{U}\mathbf{U}^T\right|\right|^2 =$$

$$\operatorname*{arg\,inf}_{\substack{\mathbf{U}^T\mathbf{U}=\mathbf{I} \\ \mathbf{A}\succeq\mathbf{0},||\mathbf{A}||=1}} ||\boldsymbol{\mathcal{S}} - [\![\mathbf{U},\mathbf{U},\mathbf{AB}]\!]||$$

where $\boldsymbol{\mathcal{S}}_{::k} := \mathbf{S}^{(k)}$, $\mathbf{A} := \mathbf{a}$ and $\mathbf{B} := \mathbf{1}^T$.

A.2.2 Best Positive Semi-Definite Approximation.

THEOREM A.1. *If $\mathbf{Y} \in \mathbb{R}^{I\times I}$ is a symmetric matrix, then a positive semi-definite matrix, $\mathbf{S} \in \mathbb{R}^{I\times I}$, that minimizes $||\mathbf{Y}-\mathbf{S}||$ such that $\operatorname{rank}(\mathbf{S}) \leq R$ has an eigenvalue decomposition consisting of the largest R non-negative eigenvalues of $\mathbf{Y}$ and their corresponding eigenvectors.*

Proof. If we denote the eigenvalue decompositions of $\mathbf{Y}$ and $\mathbf{S}$ as $\mathbf{U}\boldsymbol{\Lambda}\mathbf{U}^T$ and $\mathbf{V}\boldsymbol{\Sigma}\mathbf{V}^T$, respectively, such that the eigenvalues are sorted in a descending order, then we have

$$\operatorname*{arg\,inf}_{\substack{\mathbf{S}\succeq\mathbf{0} \\ \operatorname{rank}(\mathbf{S})\leq R}} ||\mathbf{Y}-\mathbf{S}|| = \operatorname*{arg\,inf}_{\substack{\mathbf{S}\succeq\mathbf{0} \\ \operatorname{rank}(\mathbf{S})\leq R}} \frac{1}{2}\operatorname{Tr}\left(\mathbf{S}^2\right) - \operatorname{Tr}\left(\mathbf{Y}^T\mathbf{S}\right)$$

or equivalently

$$\operatorname*{arg\,inf}_{\substack{\mathbf{V}^T\mathbf{V}=\mathbf{I} \\ \operatorname{rank}(\boldsymbol{\Sigma})\leq R \\ \boldsymbol{\Sigma}_{ii}\geq 0 \\ \forall i\neq j, \boldsymbol{\Sigma}_{ij}=0}} \frac{1}{2}\sum_{i=1}^{I}\boldsymbol{\Sigma}_{ii}^2 - \operatorname{Tr}\left(\mathbf{U}\boldsymbol{\Lambda}\mathbf{U}^T\mathbf{V}\boldsymbol{\Sigma}\mathbf{V}^T\right).$$

From [16] we know that $\operatorname{Tr}\left(\mathbf{U}\boldsymbol{\Lambda}\mathbf{U}^T\mathbf{V}\boldsymbol{\Sigma}\mathbf{V}^T\right) \leq \sum_{i=1}^{I}\boldsymbol{\Lambda}_{ii}\boldsymbol{\Sigma}_{ii}$, and notice that the equality can be achieved if we set $\mathbf{V}=\mathbf{U}$. Then, we can find an op-

Model Name	Model Definition
Spectral Clustering [11]	$\sup_{\mathbf{U}^T\mathbf{U}=\mathbf{I}} \operatorname{Tr}\left(\mathbf{U}^T\left(\mathbf{D}^{-\frac{1}{2}}\mathbf{X}\mathbf{D}^{-\frac{1}{2}}\right)\mathbf{U}\right)$
MC-TR-I-EVD [10]	$\sup_{\mathbf{U}^T\mathbf{U}=\mathbf{I}} \sum_{k=1}^{K} \operatorname{Tr}\left(\mathbf{U}^T\left(\mathbf{D}^{(k)^{-\frac{1}{2}}}\boldsymbol{\mathcal{X}}_{::k}\mathbf{D}^{(k)^{-\frac{1}{2}}}\right)\mathbf{U}\right)$
MC-TR-I-EVDIT [10]	$\sup_{\substack{\mathbf{U}^T\mathbf{U}=\mathbf{I}\\ \mathbf{a}\succeq\mathbf{0},\lvert\lvert\mathbf{a}\rvert\rvert=1}} \sum_{k=1}^{K} \operatorname{Tr}\left(\mathbf{U}^T\left(\mathbf{a}_k\mathbf{D}^{(k)^{-\frac{1}{2}}}\boldsymbol{\mathcal{X}}_{::k}\mathbf{D}^{(k)^{-\frac{1}{2}}}\right)\mathbf{U}\right)$
Co-regularized Spectral Clustering (pairwise regularization) [9]	$\sup_{\mathbf{U}^{(k)^T}\mathbf{U}^{(k)}=\mathbf{I}} \sum_{k=1}^{K} \operatorname{Tr}\left(\mathbf{U}^{(k)^T}\left(\mathbf{D}^{(k)^{-\frac{1}{2}}}\boldsymbol{\mathcal{X}}_{::k}\mathbf{D}^{(k)^{-\frac{1}{2}}}\right)\mathbf{U}^{(k)}\right) + \lambda\sum_{i\neq j}\operatorname{Tr}\left(\mathbf{U}^{(i)}\mathbf{U}^{(i)^T}\mathbf{U}^{(j)}\mathbf{U}^{(j)^T}\right)$
Co-regularized Spectral Clustering (centroid-based regularlization) [9]	$\sup_{\substack{\mathbf{U}^{(k)^T}\mathbf{U}^{(k)}=\mathbf{I}\\ \mathbf{U}^{*T}\mathbf{U}^{*}=\mathbf{I}}} \sum_{k=1}^{K} \operatorname{Tr}\left(\mathbf{U}^{(k)^T}\left(\mathbf{D}^{(k)^{-\frac{1}{2}}}\boldsymbol{\mathcal{X}}_{::k}\mathbf{D}^{(k)^{-\frac{1}{2}}}\right)\mathbf{U}^{(k)}\right) + \lambda_k \operatorname{Tr}\left(\mathbf{U}^{(k)}\mathbf{U}^{(k)^T}\mathbf{U}^{*}\mathbf{U}^{*T}\right)$
ComClus [12]	$\inf_{\mathbf{U},\mathbf{W},\mathbf{A},\mathbf{B}\geq 0} \sum_{k=1}^{K} \left\lvert\left\lvert\mathbf{X}_{::k} - \mathbf{U}\operatorname{diag}\left(\mathbf{W}_{k:}\right)\mathbf{U}^T\right\rvert\right\rvert^2 + \beta\lvert\lvert\mathbf{W}-\mathbf{A}\mathbf{B}\rvert\rvert^2 + \rho(\lvert\lvert\mathbf{U}\rvert\rvert_1 + \lvert\lvert\mathbf{A}\rvert\rvert_1 + \lvert\lvert\mathbf{B}\rvert\rvert_1)$
CMNC [2]	$\inf_{\substack{\mathbf{U}^{(m)},\mathbf{A}\geq 0\\ \mathbf{A}\mathbf{1}=\mathbf{1}}} \left\lvert\left\lvert\boldsymbol{\mathcal{Y}} - \sum_{m=1}^{M}\left(\mathbf{U}^{(m)}\mathbf{U}^{(m)^T}\right)\times_3 \mathbf{A}_{:m}\right\rvert\right\rvert^2$ where $\boldsymbol{\mathcal{Y}}_{::k} := \mathbf{D}^{(k)^{-\frac{1}{2}}}\boldsymbol{\mathcal{X}}_{::k}\mathbf{D}^{(k)^{-\frac{1}{2}}}$
Richcom [6]	$\inf_{\substack{\mathbf{U}^{(m)},\mathbf{V}^{(m)},\\ \mathbf{a}^{(m)}}} \left\lvert\left\lvert\boldsymbol{\mathcal{X}} - \sum_{m=1}^{M}\left(\mathbf{U}^{(m)}\mathbf{V}^{(m)^T}\right)\times_3 \mathbf{a}^{(m)}\right\rvert\right\rvert^2 + \sum_{m=1}^{M} r\left(\mathbf{U}^{(m)},\mathbf{V}^{(m)},\mathbf{a}^{(m)}\right)$ where $r\left(\mathbf{U}^{(m)},\mathbf{V}^{(m)},\mathbf{a}^{(m)}\right)$ encodes sparsity and non-negativity

Table 2: Mathematical formulations of various graph clustering methods.

timal $\mathbf{\Sigma}$ by solving

$$\underset{\substack{\text{rank}(\mathbf{\Sigma})\leq R\\ \mathbf{\Sigma}_{ii}\geq 0\\ \forall i\neq j, \mathbf{\Sigma}_{ij}=0}}{\arg\inf} \sum_{i=1}^{I} \frac{1}{2}\mathbf{\Sigma}_{ii}^2 - \mathbf{\Lambda}_{ii}\mathbf{\Sigma}_{ii}.$$

To this end, notice first that for any i we have

$$\inf_{\mathbf{\Sigma}_{ii}\geq 0} \frac{1}{2}\mathbf{\Sigma}_{ii}^2 - \mathbf{\Lambda}_{ii}\mathbf{\Sigma}_{ii} = \begin{cases} -\frac{1}{2}\mathbf{\Lambda}_{ii}^2 & \text{if } \mathbf{\Lambda}_{ii}\geq 0 \\ 0 & \text{if } \mathbf{\Lambda}_{ii} < 0 \end{cases}$$

and, therefore, the optimal $\mathbf{\Sigma}$ is given by setting

$$\mathbf{\Sigma}_{ii} = \begin{cases} \max\{\mathbf{\Lambda}_{ii}, 0\} & \text{if } i \leq R \\ 0 & \text{if } i > R \end{cases}$$

□

A.2.3 Derivation of (3.9).

$$\underset{\substack{\mathbf{U}^{(m)^T}\mathbf{U}^{(m)}=\mathbf{I}\\ \mathbf{B}^T\in\mathcal{I}}}{\arg\inf} \sum_{m=1}^{M}\sum_{k=1}^{K} \left|\left|\boldsymbol{\mathcal{Y}}_{::k} - \mathbf{A}_{km}\mathbf{U}^{(m)}\mathbf{U}^{(m)^T}\right|\right|^2 =$$

$$\underset{\substack{\mathbf{U}^{(m)^T}\mathbf{U}^{(m)}=\mathbf{I}\\ \mathbf{B}^T\in\mathcal{I}}}{\arg\inf} \sum_{m=1}^{M}\sum_{k=1}^{K} \text{Tr}\left(\left(\mathbf{A}_{km}\mathbf{U}^{(m)}\mathbf{U}^{(m)^T}\right)^2\right) +$$

$$-2\,\text{Tr}\left(\mathbf{A}_{km}\mathbf{U}^{(m)}\mathbf{U}^{(m)^T}\boldsymbol{\mathcal{Y}}_{::k}\right) =$$

$$\underset{\substack{\mathbf{U}^{(m)^T}\mathbf{U}^{(m)}=\mathbf{I}\\ \mathbf{B}^T\in\mathcal{I}}}{\arg\inf} \sum_{m=1}^{M}\sum_{k=1}^{K} \text{Tr}\left(\mathbf{A}_{km}{}^2\mathbf{U}^{(m)^T}\mathbf{U}^{(m)}\right) +$$

$$-2\,\text{Tr}\left(\mathbf{U}^{(m)^T}\mathbf{A}_{km}\boldsymbol{\mathcal{Y}}_{::k}\mathbf{U}^{(m)}\right) =$$

$$\underset{\substack{\mathbf{U}^{(m)^T}\mathbf{U}^{(m)}=\mathbf{I}\\ \mathbf{B}^T\in\mathcal{I}}}{\arg\sup} \sum_{m=1}^{M} \text{Tr}\left(\mathbf{U}^{(m)^T}\mathbf{Z}^{(m)}\mathbf{U}^{(m)}\right)$$

where $\mathbf{Z}^{(m)} := \sum_{k=1}^{K} 2\mathbf{A}_{km}\boldsymbol{\mathcal{Y}}_{::k} - ||\mathbf{A}_{:m}||^2\,\mathbf{I}$.

A.2.4 Derivation of (3.10).

$$\underset{\substack{\mathbf{U}^{(m)^T}\mathbf{U}^{(m)}=\mathbf{I}\\ \mathbf{B}^T\in\mathcal{I}}}{\arg\inf} \sum_{m=1}^{M}\sum_{k=1}^{K} \left|\left|\boldsymbol{\mathcal{Y}}_{::k} - \mathbf{A}_{km}\mathbf{U}^{(m)}\mathbf{D}_{\mathbf{B}}^{(m)}\mathbf{U}^{(m)^T}\right|\right|^2 =$$

$$\underset{\substack{\mathbf{U}^{(m)^T}\mathbf{U}^{(m)}=\mathbf{I}\\ \mathbf{B}^T\in\mathcal{I}}}{\arg\inf} \sum_{m=1}^{M}\sum_{k=1}^{K} \text{Tr}\left(\mathbf{A}_{km}{}^2\mathbf{D}_{\mathbf{B}}^{(m)^2}\right) +$$

$$\text{Tr}\left(-2\mathbf{A}_{km}\mathbf{U}^{(m)}\mathbf{D}_{\mathbf{B}}^{(m)}\mathbf{U}^{(m)^T}\boldsymbol{\mathcal{Y}}_{::k}\right) =$$

$$\underset{\substack{\mathbf{U}^{(m)^T}\mathbf{U}^{(m)}=\mathbf{I}\\ \mathbf{B}^T\in\mathcal{I}}}{\arg\inf} \sum_{m=1}^{M} \text{Tr}\left(||\mathbf{A}_{:m}||^2\,\mathbf{D}_{\mathbf{B}}^{(m)^2}\right) +$$

$$\text{Tr}\left(-2\,||\mathbf{A}_{:m}||\,\mathbf{U}^{(m)}\mathbf{D}_{\mathbf{B}}^{(m)}\mathbf{U}^{(m)^T}\mathbf{Z}^{(m)}\right) =$$

$$\underset{\substack{\mathbf{U}^{(m)^T}\mathbf{U}^{(m)}=\mathbf{I}\\ \mathbf{B}^T\in\mathcal{I}}}{\arg\inf} \sum_{m=1}^{M} \left|\left|\mathbf{Z}^{(m)} - ||\mathbf{A}_{:m}||\,\mathbf{U}^{(m)}\mathbf{D}_{\mathbf{B}}^{(m)}\mathbf{U}^{(m)^T}\right|\right|^2$$

where $\mathbf{Z}^{(m)} := \sum_{k=1}^{K} \mathbf{A}_{km}\boldsymbol{\mathcal{Y}}_{::k}/\,||\mathbf{A}_{:m}||$.

A.3 Derivation of Space & Time Complexity

Space complexity. First note that $\mathbf{A}$ requires space $O(K)$ since each of its rows has only one non-zero element. Additionally, space $O(I^2)$ is required for storing $\mathbf{Z}^{(m)}$ which, given the sparsity of a wide range of real-world graphs, can potentially be greatly reduced to $O(\text{nnz}(\boldsymbol{\mathcal{X}}))$ if a coordinate list representation is used instead. Then, note that using eigensolvers that calculate full eigendecompositions makes $\mathbf{U}$ and $\mathbf{B}$ require space $O(I^2)$ and $O(I)$, respectively, but using an eigensolver that only calculates the necessary eigenpairs can reduce this to $O(IR)$ and $O(R)$, respectively. Lastly, we need to store $[\![\mathbf{U},\mathbf{U},\mathbf{B}]\!]$ which requires space $O(MI^2)$. Note that before calculating $[\![\mathbf{U},\mathbf{U},\mathbf{B}]\!]$ any existing $\mathbf{Z}^{(m)}$ can be discarded and the memory it used can become available to $[\![\mathbf{U},\mathbf{U},\mathbf{B}]\!]$. Therefore, the total space complexity of our method is $O(K + MI^2 + IR)$.

Time complexity. First we see that forming $\mathbf{Z}^{(m)}$ requires $O(\text{nnz}(\mathbf{A}_{:m})\,I^2)$ time, and, therefore, forming all of them requires $O(KI^2)$ time. Alternatively, if we again take advantage of graph sparsity and the coordinate list representation, the time complexity of forming $\mathbf{Z}^{(m)}$ can be bounded above by $O(\text{nnz}(\mathbf{A}_{:m})\,\text{nnz}(\boldsymbol{\mathcal{X}}))$, and, therefore, the total time complexity of forming all of them can be bounded above by $O(K\,\text{nnz}(\boldsymbol{\mathcal{X}}))$. Then, we update $\mathbf{U}$ and $\mathbf{B}$ by calculating the necessary eigenpairs from each $\mathbf{Z}^{(m)}$ which in a naive implementation will have a total time complexity of $O(MI^3)$. Calculating the M eigendecompositions in parallel reduces this time complexity to $O(I^3)$, and although it increases the space complexity of $\{\mathbf{Z}^{(m)}\}_{m=1}^{M}$ to $O(MI^2)$ since all of them now need to be accessed simultaneously, it does not increase the total space complexity of our method. Additionally, the time complexity of the eigendecompositions can be in practice further dramatically reduced if only R eigenpairs are computed for each eigendecomposition and an eigensolver that can take advantage of sparsity is used [5]. The next step is to calculate $[\![\mathbf{U},\mathbf{U},\mathbf{B}]\!]$ and note that $[\![\mathbf{U},\mathbf{U},\mathbf{B}]\!]_{::m} = \mathbf{U}\,\text{diag}(\mathbf{B}_{m:})\,\mathbf{U}^T$ which costs $O(\text{nnz}(\mathbf{B}_{m:})\,I^2)$. Therefore, the total time complexity for computing $[\![\mathbf{U},\mathbf{U},\mathbf{B}]\!]$

will be $O\left(RI^2\right)$. The last step is to update $\mathbf{A}$ which costs $O\left(KMI^2\right)$ for all three constraint types studied. Therefore, for t iterations the total time complexity of our method is $O\left(tMI^3 + t(R + KM)I^2\right)$.

A.4 Experiment Setup Details. Despite the fact that the embedding calculation method is the main novel algorithmic contribution in the original papers where the baseline methods were proposed, their authors have made algorithmic choices for the other segments as well. However, it is not clear whether and why these choices would lead to optimal clustering performance. Therefore, since we are mostly interested in comparing the quality of the embeddings produced by each method, we define enhanced versions for all methods by considering combinations of their embeddings calculation method with various methods of data preprocessing, embeddings postprocessing and embeddings clustering. Then, the combination that produces the best clustering performance is reported as the performance of each enhanced method. Specifically, for data preprocessing we consider both raw data and the normalized Laplacians of the graph views. We use the normalized Laplacians for directed graphs as defined in [21] which is a direct extension of the normalized Laplacians for undirected graphs we discussed in Appendix A.1.3. Then, for the node embeddings of the n-th calculated views cluster, we consider both their raw unnormalized form and their normalization as either $\mathbf{U}\,\mathrm{diag}\left(\mathbf{B}_{n:}\right)$ or $\mathbf{U}\,\mathrm{diag}\left(\mathbf{C}_{n:}\right)$ with $\mathbf{C}_{ij} := \sqrt{|\mathbf{B}_{ij}|}$. After this, embeddings may optionally be further normalized to be unit vectors. Finally, we consider k-means, maximum likelihood and inner product thresholding for clustering the embeddings.

Based on the theoretical arguments we made in Section 3, we define original GenClus to use normalized Laplacians for the preprocessing step, and a nonnegative constraint for $\mathbf{A}$ and $\mathbf{B}$. However, we still allow it to be paired with any embedding post-processing and clustering method, in the same way that enhanced GenClus does. Note that, although this design choice may seem that it could prove an unfair advantage of GenClus over the baselines, we believe that it still was the right decision for two main reasons. First, as with the other baselines, we have no robust theoretical justification for choosing a specific combination of embedding postprocessing and clustering method. Second, we noticed in our experiments that the performance of our method was not significantly affected by altering this combination.

Lastly, we would also like to point out that due to potential issues that we think we may have discovered with the optimization steps proposed in the original paper of Richcom, we decided to consider a slightly different variant in our experiments. Specifically, we propose Symmetric Richcom, which uses the same methods of preprocessing, postprocessing and embeddings clustering as the original Richcom, but has $\mathbf{V}$ constrained to be identical to $\mathbf{U}$.

A.4.1 Software & Hardware Specifications. All experiments related to clustering performance were run on a machine with two Intel® Xeon® E5-2680 v3 processors, 378GB of RAM and MATLAB 9.7.0.1190202 (R2019b). All timing experiments were run on MATLAB 9.7.0.1737446 (R2019b) Update 9 on a machine with an AMD Ryzen™ 5 5600 processor overclocked at 4.75GHz paired with 16GB of DDR4 dual-channel SDRAM and a 1600MHz memory clock. Lastly, in all experiments the mtimex routine [17] was used for faster matrix multiplications.

A.4.2 A Solver for Symmetric Richcom. We propose updating $\mathbf{U}$ and $\mathbf{A}$ according to the solver described in [15]. In fact, this is the solver used by ComClus for its updates of $\mathbf{U}$, $\mathbf{A}$ and $\mathbf{B}$. This provides further motivation for deriving the updates of Symmetric Richcom in this manner in the sense that it may help reduce the impact the solvers can have on the differences in the performance of the two models. To avoid a lengthy derivation of the updates, notice that we can very easily deduce them by directly modifying the solver of ComClus. Specifically, when solving for $\mathbf{U}$ in symmetric Richcom we use the same update as in ComClus but we substitute $\mathbf{W}$ with $\mathbf{AB}$. Similarly, when solving for $\mathbf{A}$ we use the same update as in ComClus but we substitute $\mathbf{W}$ with $\mathbf{Y}_{(3)}$ and $\mathbf{B}$ with $\mathbf{B}(\mathbf{U} \odot \mathbf{U})^T$. Also, note that such a solver is also capable of imposing sparsity and involves the same parameter ρ as ComClus.

A.4.3 Clustering Quality on Artificial Data.

Embeddings Calculation. Note that the following parametrization applies to both the original and the enhanced methods, unless stated otherwise. First, we calculate 100 samples of $\mathbf{U}$, $\mathbf{A}$ and $\mathbf{B}$ by modeling the data tensor as $[\![\mathbf{U}, \mathbf{U}, \mathbf{AB}]\!]$, where for each sample we consider a different random initialization for $\mathbf{U}$, $\mathbf{A}$ and $\mathbf{B}$ and a different instance of the same data tensor. R can range from 6 to 10, M is set to 3, the convergence criterion is always defined to be the relative change in the value of the objective function of each model, the convergence threshold for the outer loop can be either $1e-3$, $1e-6$ or $1e-9$, and the maximum number of iterations for the outer loop is set to 1000. Note that apart from ComClus the other models consist of only

one loop. Regarding enhanced GenClus, we allow all nine possible combinations of constraints on $\mathbf{A}$ and $\mathbf{B}$ discussed in Section 3.1, which is in contrast with original GenClus which we defined to have only non-negative constraints. Also, note we do not make use of any of the optimizations discussed in Appendix A.3. For ComClus, ρ can use values in the set of the 6 equally spaced values from 0 to 0.16, β can use values in the set of the 6 equally spaced values from 0.01 to 0.9145, while the threshold of convergence criterion of its inner loop is set to $1e-6$. For CMNC we set its delta to 1, while for Symmetric Richcom ρ can take values in the set of the 6 equally spaced values from 0 to 0.2. Also note that both CMNC and Symmetric Richcom require a fixed user-specified $\mathbf{B}$, which in our experiments we always set it to reflect the ground truth structure as closely as possible. Lastly, for both CMNC and Symmetric Richcom, and when both M and R are larger than or equal to their ground truth values, we begin by creating the $\mathbf{B}$ of optimal dimensions and structure. Then, any remaining rows are added as all-zeros, and in turn any remaining columns are added as random indicator vectors. Also, when M or R is less than the ground truth, then an appropriate number of rows or columns, respectively, is removed randomly.

Matching calculated and ground truth view clusters. After the model is calculated, we assign the m-th view to the n-th cluster when the n-th element of the m-th row of $\mathbf{A}$ has the largest magnitude among all elements of that row. The reason for this is that $[\![\mathbf{U}, \mathbf{U}, \mathbf{AB}]\!] = [\![\mathbf{U}, \mathbf{U}, \mathbf{B}]\!] \times_3 \mathbf{A}$ which implies that the m-th view will be reconstructed mostly based on the n-th frontal slice of $[\![\mathbf{U}, \mathbf{U}, \mathbf{B}]\!]$. Then, we are matching the calculated views clusters to the ground truth views clusters so that we can in turn compare the calculated nodes clusters with the appropriate ground truth nodes clusters. To this end, for each calculated views cluster we create a membership vector containing either a 1 or a 0 at the m-th position if the m-th view belongs or does not belong, respectively, to that cluster, and then we normalize it to have unit norm. Similarly, we also create membership vectors for the ground truth views clusters and we match each calculated views cluster to the ground truth views cluster with which the inner product of their corresponding membership vectors is maximized.

Clustering Performance Evaluation. To assess the performance of each method we will use the Adjusted Mutual Information (AMI). AMI is an adjusted version of the Normalized Mutual Information (NMI) designed to mitigate the flaw of NMI of getting larger values as the number of clusters gets closer to the total number of samples. Note that, although not reported here, in our experiments NMI and the Adjusted Rand Index (ARI) produced very similar results to AMI. The only notable difference is that sometimes the original Symmetric Richcom gives better ARI score than the original CMNC, which is not the case with AMI. However, since, as we will also see next, Symmetric Richcom and CMNC have the worst performance among all methods, the presentation of NMI and ARI scores is omitted.

A.4.4 Real-World Case Study. We preprocess the data by removing all airlines that offered less than 100 flights and then all airports corresponding to less than 30 flights, and we repeat this process until no further airline or airport is removed. This leaves us with 235 airports and 61 airlines whose flight counts are then organized into a tensor of size $235 \times 235 \times 61$. Note that since each of the 61 frontal slices of this tensor corresponds to the flight network of a specific airline, while the views clusters in this case will represent clusters of airlines. Similarly, the 235 rows and columns of each frontal slice correspond to the different airports between which the corresponding airline has been flying. Therefore, each node clustering will correspond to a clustering of airports.

A.4.5 Execution Time on Artificial Data. The quantities we will study are the number of nodes, I, the number of views, K, the node embeddings dimension, R, and view embeddings dimension, M. When varying I, we set K, R and M to 9, 3 and 3, respectively. When varying K, we set I, R and M to 60, 3 and 3 respectively. When varying R, we set I, K, and M to 240, 9 and 3, respectively. When varying M we set I, K and R to 120, 9 and 96, respectively. We also understand, that some could argue that these numbers should have been larger for a proper time complexity analysis. However, doing so would force us to completely exclude CMNC from the comparisons, since it would very quickly produce out-of-memory errors. In fact, as we will see next, this issue sometimes occurs even with the aforementioned experiment setup.

The graph generation process is similar to the one in Section 4.1, but here the intra-community edge density, γ, is always fixed to 0.15. When altering the number of nodes or views, the sizes of node clusters and views clusters are scaled proportionally. Similarly, the parameters of the embedding generation is parametrized in the same way as in Appendix A.4.3, but with a few modifications. First, we consider three equally spaced values instead of six for ComClus for both its β and ρ. Second, we consider a fixed threshold of $1e-6$ for all methods. And third, we calculate 5 samples for each

parameter combination instead of 100.

A.5 Real-World Case Study 2. Here we study the well known reality mining dataset [4] which documented the interactions of a group of students and faculty from the MIT Media Laboratory and MIT Sloan business school via special software on their phones. In our experiments, we aggregate all communications between 82 participants on an hourly basis and for a total of 15 days, which leads to a time-evolving graph with an adjacency tensor of size $82 \times 82 \times 360$.

Figure Figure 6a indicates that the communication patterns of the participants during the day have a slight but observable tendency to be different from the communication patterns at night. Specifically, Figure Figure 6c shows that during the day there exist two major clusters consisting mostly of Media Lab Graduate students and two clusters of Sloan Business School students. On the other hand, from Figure Figure 6b we observe that during the night the clusters of Media Lab students shrink in size, while the clusters of Sloan Business School students vanish altogether. This is in accordance with our intuition that during the night the intensity communications is expected to be lower.

References

[1] Stephen Boyd et al., *Distributed optimization and statistical learning via the alternating direction method of multipliers*, in Foundations and Trends® in Machine learning 3.1 (2011), pp. 1–122.

[2] Zitai Chen et al., *Tensor decomposition for multilayer networks clustering*, in Proceedings of the AAAI Conference on Artificial Intelligence, vol. 33, 01, 2019, pp. 3371–3378.

[3] Lieven De Lathauwer, *Decompositions of a higher-order tensor in block terms—Part II: Definitions and uniqueness*, in SIAM Journal on Matrix Analysis and Applications 30.3 (2008), pp. 1033–1066.

[4] Nathan Eagle and Alex (Sandy) Pentland, *Reality mining: sensing complex social systems*, in Personal and ubiquitous computing 10.4 (2006), pp. 255–268.

[5] Roger G Grimes, John G Lewis, and Horst D Simon, *A shifted block Lanczos algorithm for solving sparse symmetric generalized eigenproblems*, in SIAM Journal on Matrix Analysis and Applications 15.1 (1994), pp. 228–272.

[6] Ekta Gujral, Ravdeep Pasricha, and Evangelos Papalexakis, *Beyond rank-1: Discovering rich community structure in multi-aspect graphs*, in Proceedings of The Web Conference 2020, 2020, pp. 452–462.

[7] Kejun Huang, Nicholas D Sidiropoulos, and Athanasios P Liavas, *A flexible and efficient algorithmic framework for constrained matrix and tensor factorization*, in IEEE Transactions on Signal Processing 64.19 (2016), pp. 5052–5065.

[8] Stefan Klus and Natasa Djurdjevac Conrad, *Koopman-based spectral clustering of directed and time-evolving graphs*, in arXiv preprint arXiv:2204.02951 (2022).

[9] Abhishek Kumar, Piyush Rai, and Hal Daume, *Co-regularized multi-view spectral clustering*, in Advances in Neural Information Processing Systems, ed. by J. Shawe-Taylor et al., vol. 24, Curran Associates, Inc., 2011.

[10] Xinhai Liu et al., *Multiview partitioning via tensor methods*, in IEEE Transactions on Knowledge and Data Engineering 25.5 (2012), pp. 1056–1069.

[11] Andrew Ng, Michael Jordan, and Yair Weiss, *On spectral clustering: Analysis and an algorithm*, in Advances in neural information processing systems 14 (2001).

[12] Jingchao Ni et al., *ComClus: A self-grouping framework for multi-network clustering*, in IEEE transactions on knowledge and data engineering 30.3 (2017), pp. 435–448.

[13] Maximilian Nickel et al., *A Review of Relational Machine Learning for Knowledge Graphs*, in Proceedings of the IEEE 104.1 (2016), pp. 11–33.

[14] J Nocedal and SJ Wright, *Numerical optimization*, in Springer Series in Operations Research and Financial Engineering (2006).

[15] D Seung and L Lee, *Algorithms for non-negative matrix factorization*, in Advances in neural information processing systems 13 (2001), pp. 556–562.

[16] CM Theobald, *An inequality for the trace of the product of two symmetric matrices*, in Mathematical Proceedings of the Cambridge Philosophical Society, vol. 77, 2, Cambridge University Press, 1975, pp. 265–267.

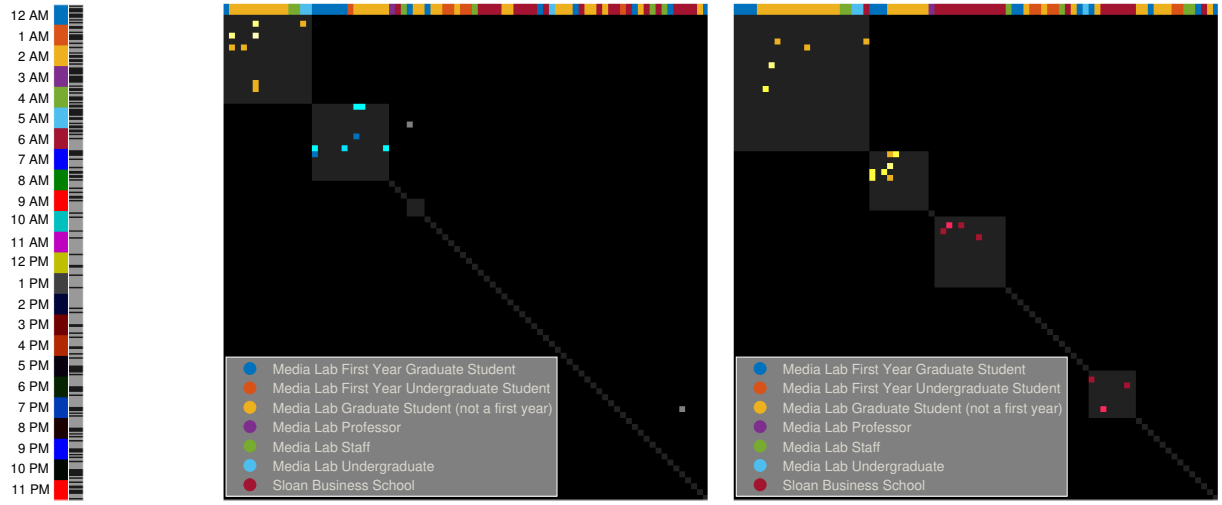
[17] James Tursa, *MTIMESX - Fast Matrix Multiply with Multi-Dimensional Support*, `https://www.mathworks.com/matlabcentral/fileexchange/25977-mtimesx-fast-matrix-multiply-with-multi-dimensional-support`, MATLAB Central File Exchange. Retrieved October 3, 2022, 2022.

[18] Ulrike Von Luxburg, *A tutorial on spectral clustering*, in Statistics and computing 17.4 (2007), pp. 395–416.

[19] Stephen J Wright, *Coordinate descent algorithms*, in Mathematical Programming 151.1 (2015), pp. 3–34.

[20] Dengyong Zhou and Christopher JC Burges, *Spectral clustering and transductive learning with multiple views*, in Proceedings of the 24th international conference on Machine learning, 2007, pp. 1159–1166.

[21] Dengyong Zhou, Jiayuan Huang, and Bernhard Schölkopf, *Learning from labeled and unlabeled data on a directed graph*, in Proceedings of the 22nd international conference on Machine learning, 2005, pp. 1036–1043.



(a) Hour clusters (b) Clusters of people during the night (c) Clusters of people during the day

Figure 6: Clustering of hours of day and people by GenClus for the reality mining dataset. The right colored bar in (a) shows the hour clusters while (b)-(c) show the adjacency matrices of a selected hour from the day and night clusters, respectively. The remaining color bars in (a)-(c) serve as ground truth labels.