# Clustering with Multiple Graphs

Wei Tang Zhengdong Lu Inderjit S. Dhillon
Department of Computer Science
The University of Texas at Austin
{wtang, luz, inderjit}@cs.utexas.edu

UTCS Technical Report TR-09-37

December 4, 2009

**Abstract**

In graph-based learning models, entities are often represented as vertices in an undirected graph with weighted edges describing the relationships between entities. In many real-world application, however, entities are often associated with relations of different types and/or from different sources, which can be well captured by multiple undirected graphs over the same set of vertices. How to exploit such multiple sources of information to make better inferences on entities remains an interesting open problem. In this paper, we focus on the problem of clustering the vertices based on multiple graphs in both unsupervised and semi-supervised settings. As one of our contributions, we propose Linked Matrix Factorization (LMF) as a novel way of fusing information from multiple graph sources. In LMF, each graph is approximated by matrix factorization with a graph-specific factor and a factor common to all graphs, where the common factor provides features for all vertices. Experiments on both synthetic and SIAM journal data show that (1) we can improve the clustering accuracy through fusing multiple sources of information with several models, and (2) LMF yields superior or competitive results compared to other graph-based clustering methods.

## 1 Introduction

Relational data are ubiquitous, and the associated modeling and inference tasks have become important topics in both machine learning and data mining[7]. The common tools modeling relational data often represent them as an undirected graph with vertices representing entities and (weighted or unweighted) edges describing the "relationships" between entities. In many application domains, these relationships are of different types or are obtained from different sources, which can be well represented by multiple undirected graphs over the same set of vertices with edges from different graphs capturing the heterogeneous relations. As one example of such multiple graphs, let us consider the proximity between researchers. Two researchers are considered to be similar if they have co-authored some papers, while it is also reasonable to assume two authors to have similar interest (probably to a lower level) if they both cited the same papers in their published work or they published in the same venues. These different type of relationships between authors naturally form multiple undirected graphs over the same set of authors. How to exploit the multiple sources of information to make better inferences about entities and relationships is an interesting open problem.

In this paper, we consider the particular graph mining task of clustering vertices into several groups in the presence of multiple types of proximity relations. We give an extensive comparison of several graph-based clustering algorithms, as well as their semi-supervised extensions. One major contribution of this paper is a novel method for extracting common factors from multiple graphs, called Linked Matrix Factorization (LMF), based on which various clustering methods can naturally apply. Experiments on both synthetic and real-world data show the efficacy of the proposed methods in combining the information from multiple sources. In particular, LMF yields superior results compared to other graph-based clustering methods in both unsupervised and semi-supervised settings.

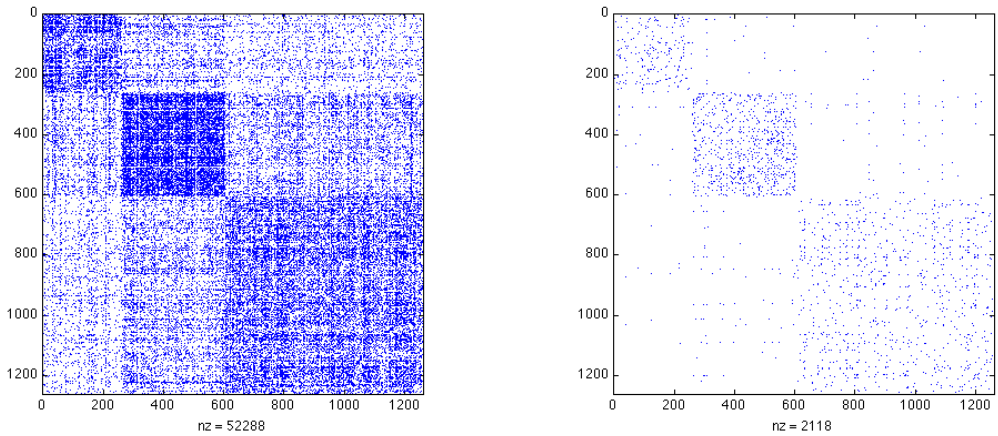



Figure 1: Example spy plots of similarity graphs between documents based on their abstracts (upper panel) and co-authorship (lower panel) .

**Road Map** The remainder of this paper is organized as follows. Section 2 discusses the characteristics of the data and the inadequate clustering performance on individual graphs. In Section 3, we discuss the extension of unsupervised clustering methods to multiple graphs. Section 4 is devoted to the formulation and optimization of Linked Matrix Factorization, and its connection to third order tensor decomposition [15, 6] and the stochastic block model [7]. In Section 5 we extend the unsupervised model in Section 3 to semi-supervised scenarios where constraints on the cluster assignments are known. Section 6 reports the experimental results in both unsupervised and semi-supervised scenarios, while Section 7 briefly discusses related work. Finally Section 8 summarizes the paper and discusses future work.

# 2 Data Characteristics

We first motivate our work by discussing the clustering problem on some real-world data. In many scientific publication domains such as CiteSeer or arXiv, the relationships between documents can often be described as multiple graphs with different link types. For example, there is information from the text by exploring document-abstract, document-title, and document-keyword matrices; information on co-authorship by exploring the document-author matrix; and information on citations. Information from different sources show very different characteristics. For example, the co-authorship graph is usually much sparser than the proximity based on abstracts, but intuitively each co-authorship edge is more informative. Figure 1 gives example of two such graphs on the same set of documents, where we plot the document-by-document matrix showing the presence of (non-zero) edges (called *spy plot*) with documents listed according to the intended clusters. Due to the extreme sparsity, some graphs alone do not contain complete information of the structure. Indeed, the co-author relationship shown in Figure 1 (lower panel) contains over 100 disconnected components, and is therefore unable to reveal the 3-cluster structure inherent in the data. It is useful but challenging to combine the distinct characteristics of different graphs–for example, in this data there are sparse but informative relations as well as abundant but less informative ones.

## 2.1 SIAM Journal Data Set

In this paper, we consider the data from eleven journals and proceedings for the period 1999-2004 published by the Society for Industrial and Applied Mathematics (SIAM). There are a total of 5022 articles in the data set, from which we generated two subsets:

- SIAM-different: containing 1260 articles published in SIAM J DISCRETE MATH, SIAM J OPTIMIZ and SIAM J SCI COMPUT;

- SIAM-similar: containing 1690 articles published in SIAM J MATRIX ANAL A, SIAM J NUMER ANAL and SIAM J SCI COMPUT.

Our task is to discover the natural cluster structure of journals based on the document similarities extracted from different sources. Note that SIAM-different is composed of three journals from different research areas and hence is easier to cluster, whereas SIAM-similar contains three journals on highly related research topics and is more difficult to cluster.

In both subsets, we consider document similarities from five different sources. The first three are obtained from document-term matrices; in particular, each document can be represented as a vector of non-trivial words from different parts of the articles, namely abstract, title or user-supplied keywords. We calculate the cosine similarity between each pair of documents within these different contexts to form the first three similarity matrices. The last two similarity matrices are obtained via the author and citation relations, respectively.

Details about the five link types are described below:

- The abstract similarity matrix $A^{(1)}$ is constructed from the document-abstract matrix. $A^{(1)}_{ij}$ is the cosine similarity between the abstracts of documents $i$ and $j$.
- The title similarity matrix $A^{(2)}$ is formed from the document-title matrix. $A^{(2)}_{ij}$ is the cosine similarity between the titles of documents $i$ and $j$.
- The keyword similarity matrix $A^{(3)}$ is computed from the document-keyword matrix. $A^{(3)}_{ij}$ is the cosine similarity between the keywords of documents $i$ and $j$.
- The author similarity matrix $A^{(4)}$ represents the number of common authors for each pair of documents.
- The citation similarity matrix $A^{(5)}$ has the citation relation between each pair of documents. $A^{(5)}_{ij} = A^{(5)}_{ji} = 1$ if there is citation between documents $i$ and $j$, and 0 otherwise.

Some statistics about the SIAM data sets are shown in Table 1. It can been seen that, in both data sets, the first three graphs are much denser than the last two graphs.

| | Description | $nnz$ (different) | $nnz$ (similar) |
|---|---|---|---|
| $A^{(1)}$ | abstract | 755,016 | 1,401,900 |
| $A^{(2)}$ | title | 108,364 | 277,338 |
| $A^{(3)}$ | keywords | 178,508 | 406,080 |
| $A^{(4)}$ | author | 2,118 | 4,742 |
| $A^{(5)}$ | citation | 1,328 | 2,248 |

Table 1: Statistics of the SIAM data. $nnz$ stands for the number of non-zero entries.

## 2.2 Clustering with Individual Graphs

We adopt *Normalized Mutual Information* (NMI) to measure the clustering performance. Let $\mathcal{Z}$ be the random variable denoting the underlying journal labels for documents, and $\hat{\mathcal{Z}}$ the random variable denoting the cluster assignments; then NMI can be computed as

$$NMI = \frac{I(\hat{\mathcal{Z}}; \mathcal{Z})}{\sqrt{H(\hat{\mathcal{Z}})H(\mathcal{Z})}}, \tag{1}$$

where $I(\hat{\mathcal{Z}}; \mathcal{Z}) = H(\mathcal{Z}) - H(\mathcal{Z}|\hat{\mathcal{Z}})$ is the mutual information between the random variables $\hat{\mathcal{Z}}$ and $\mathcal{Z}$, $H(\mathcal{Z})$ is the Shannon entropy of $\mathcal{Z}$, and $H(\mathcal{Z}|\hat{\mathcal{Z}})$ is the conditional entropy of $\mathcal{Z}$ given $\hat{\mathcal{Z}}$.

Figure 2 shows spy plots for all the five adjacency matrices belonging to the SIAM-different and SIAM-similar data sets, with documents being aligned to their published journals. Clearly each graph contains certain information

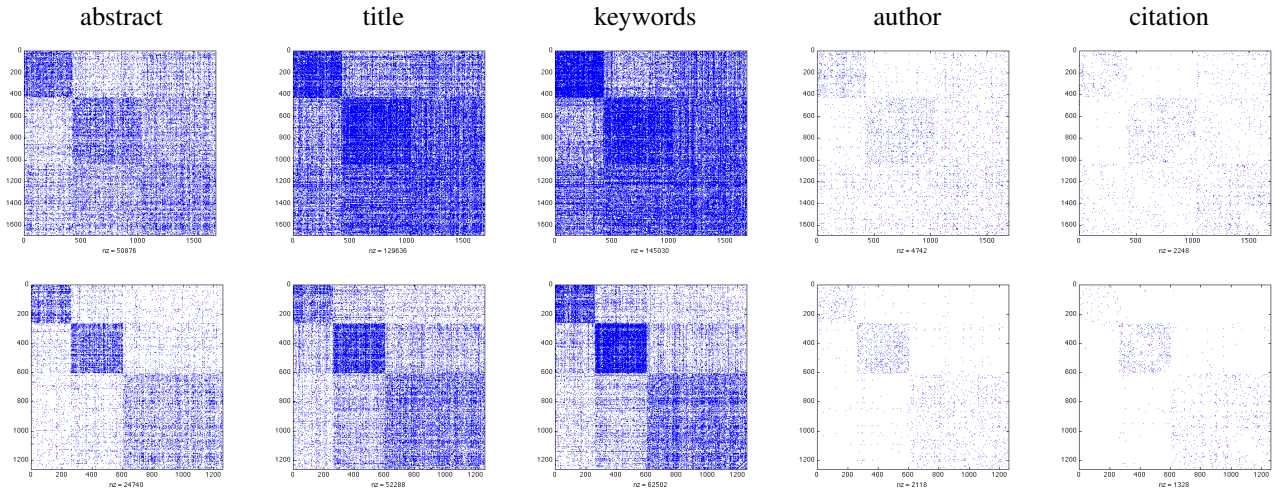



Figure 2: Spy plots of SIAM-similar data set (upper row) and SIAM-different data set (lower row). SIAM-different data set can be seen to be easier to cluster.

| | SIAM-different | SIAM-similar |
|---|---|---|
| abstract | 0.5893 | 0.2037 |
| title | 0.0324 | 0.2021 |
| keywords | 0.3731 | 0.2502 |
| author | 0.0042 | 0.0017 |
| citation | 0.0211 | 0.0078 |

Table 2: Clustering performance measured by NMI on two SIAM data sets.

about the relationships between documents. If we apply spectral clustering [10] on each individual graph, we get the clustering results shown in Table 2 in terms of NMI. It is clear from Figure 2 and Table 2 that although the edges in the last two graphs are highly consistent with the cluster structure of journals, they do not contain enough information to recover the clusters alone. As we will show in later sections, combining all the graphs, especially with our proposed LMF model, can yield significantly improved clustering results.

# 3 Unsupervised Clustering Models

Let us take one step back and consider the more general problem of clustering with multiple graphs. Suppose we are given $M$ undirected graphs whose adjacency matrices are $A^{(m)}, m = 1, 2, \cdots, M$, each of size $N \times N$, with vertices in all graphs corresponding to the same entities. We intend to find a clustering of the vertices based on the information from multiple sources.

Besides clustering each graph individually, we also have the following baseline models for combining the information from multiple graphs.

**Summation of Graphs** We find a combined adjacency matrix

$$A = \sum_{m=1}^{M} A^{(m)}.$$

With this new adjacency matrix $A$, we can perform spectral partitioning which can be achieved by computing the smallest eigenvectors of the graph Laplacian

$$L = D - A,$$

where $D$ is the diagonal degree matrix with $D_{ii} = \sum_j A_{ij}$. The use of eigenvectors can also be motivated as minimizing the "roughness" of vector $\mathbf{f} = [f_1, \cdots, f_N]^T$ over all the graphs:

$$\mathcal{G} = \sum_{m=1}^{M} \mathbf{f}^T L^{(m)} \mathbf{f} = \sum_{m=1}^{M} \sum_{i,j=1}^{N} A_{ij}^{(m)} (f_i - f_j)^2 \tag{2}$$

where $L^{(m)}$ is the graph Laplacian for the $m^{th}$ graph. Alternatively, we can use the normalized adjacency matrix, $\tilde{A} = \sum_{m=1}^{M} \tilde{A}^{(m)}$, where $\tilde{A}^{(m)} = (D^{(m)})^{-1/2} A^{(m)} (D^{(m)})^{-1/2}$.

**Summation of Spectral Kernels** We first construct spectral kernels for each graph, i.e., kernel $K^{(m)}$ based on the eigen-spectrum of the graph Laplacian $L^{(m)}$, and then use the summation

$$K = \sum_{m=1}^{M} K^{(m)}$$

as the kernel summarizing all graphs. One particular example (called step-function kernel in [13]) is the model

$$K^{(m)} = \sum_{k=1}^{d} \mathbf{v}_k^{(m)} (\mathbf{v}_k^{(m)})^T$$

where $\mathbf{v}_k^{(m)}$ is the $k^{th}$ smallest eigenvector of graph Laplacian $L^{(m)}$ and $d \ll N$ is the number of eigenvectors used per individual graph. Clustering can then be obtained by performing kernel K-means on kernel $K$. Other choices of $K^{(m)}$ include the heat diffusion kernel and regularized inverse of graph Laplacian[13], but the discussion of them is omitted here due to their inferior performance on our task.

**Consensus Clustering** Consensus clustering reconciles clustering results about the same data set coming from different sources. In this paper we follow the models in [14], where three consensus clustering algorithms are proposed: Cluster-based Similarity Partitioning Algorithm, HyperGraph Partitioning Algorithm, and Meta-Clustering Algorithm. In our experiments (Section 6), we only report the best result from these three methods.

# 4 Linked Matrix Factorization

One major limitation of the baseline models is that they treat all graphs on an equal basis, and therefore cannot discriminate the informative sources and uninformative or noisy ones. A more sensible alternative is to extract the structure information shared by all the sources, and hence filter out irrelevant information or noise. Here we present Linked Matrix Factorization (LMF), a novel model for finding the common factor for all graphs .

## 4.1 Model

One natural model for unsupervised graph clustering is to approximate the given graph through a low-rank matrix factorization $A \approx P \Lambda P^T$, where $P$ is an $N \times d$ matrix and $\Lambda$ is an $d \times d$ symmetric matrix. Since we are given multiple graphs and the underlying entities are shared among graphs, a common factor matrix is desirable to link the multiple matrix factorizations together. Therefore, the objective of clustering over multiple graphs by matrix factorization can be formulated as minimizing

$$\mathcal{G} = \frac{1}{2} \sum_{i=1}^{M} \|A^{(m)} - P \Lambda^{(m)} P^T\|_F^2 + \frac{\alpha}{2} \left( \sum_{m=1}^{M} \|\Lambda^{(m)}\|_F^2 + \|P\|_F^2 \right), \tag{3}$$

where matrix $P$ is the common factor shared among graphs, $\Lambda^{(m)}$ captures the characteristics of each graph (note that we do not constrain $\Lambda^{(m)}$ to be diagonal), $\|\cdot\|_F$ denotes the Frobenius norm and $\alpha$ is the regularization parameter. Matrix $P$ can be regarded as a low dimensional embedding of entities characterized by multiple graphs, the differences being captured by $\Lambda^{(m)}$. The regularization terms on both $P$ and $\Lambda^{(m)}$ are added to improve numerical stability and to avoid overfitting. In addition to the generic form given in (3), there are several possible alternative modeling choices.

For example, instead of using the squared Frobenius norm, we could choose the relative entropy or other divergence measures for comparing $A^{(m)}$ and $P\Lambda^{(m)}P^T$. If the graphs were not symmetric, we could instead model each graph as $P\Lambda^{(m)}Q^T$. One could also enforce the columns of $P$ to be orthonormal and drop the regularization term. However, in this paper, we will only focus on the case where each $A^{(m)}$ is an undirected symmetric graph and the approximation error is measured by the squared Frobenius norm.

## 4.2 Optimization

Note that the solutions to LMF are not unique. For instance, let matrices $P^*$ and ${\Lambda^{(m)}}^*$ $(i = 1, \ldots, M)$ be the solutions to the optimization problem (3), then for any orthogonal matrix $R \in \mathbb{R}^{d\times d}$ ($R$ only needs to be non-singular if there is no regularization term), the matrices $P^*R$ and $R^{-1}{\Lambda^{(m)}}^*R^{-1}$ will also be solutions. Moreover, the objective function is not jointly convex in $P$ and $\Lambda^{(m)}$. Hence, we adopt an effective alternating minimization algorithm to find a locally optimal solution to LMF. First, matrix $P$ is optimized while fixing each $\Lambda^{(m)}$; then, each matrix $\Lambda^{(m)}$ is optimized while fixing matrix $P$. This procedure is repeated until convergence. In optimizing matrix $P$ and each $\Lambda^{(m)}$, we apply a quasi-Newton method, Limited memory BFGS (L-BFGS) [9], to optimize each factor in the inner loop.

The bottleneck in the L-BFGS algorithm is the evaluation of the objective in (3) and its gradient with respect to $P$ and each $\Lambda^{(m)}$, respectively. Taking the derivative of (3) with respect to $P$, we get

$$\frac{\partial \mathcal{G}}{\partial P} = -2\sum_{i=1}^{M}(A^{(m)} - P\Lambda^{(m)}P^T)P\Lambda^{(m)} + \alpha P, \tag{4}$$

and taking the derivative of (3) with respect to $\Lambda^{(m)}$ yields

$$\frac{\partial \mathcal{G}}{\partial \Lambda^{(m)}} = -P^T(A^{(m)} - P\Lambda^{(m)}P^T)P + \alpha\Lambda^{(m)}. \tag{5}$$

One can gain computational efficiency by taking advantage of the sparsity of $A^{(m)}$. In particular, the first term of the objective in (3) can be rewritten as:

$$\mathcal{G}' = \frac{1}{2}\sum_{i=1}^{M}\left(\|A^{(m)}\|_F^2 - 2\,\mathrm{Tr}(\Lambda^{(m)}P^TA^{(m)}P) + \mathrm{Tr}(P^TP\Lambda^{(m)}P^TP\Lambda^{(m)})\right), \tag{6}$$

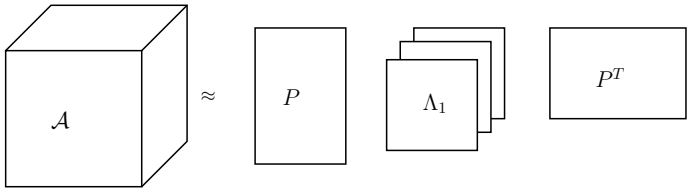
which can be evaluated in $O(d(nnz + Nd))$ time for each graph ($nnz$ represents the number of nonzero entries averaged over all graphs). Similarly, computing the gradient in (4) and (5) takes $O(d(nnz + Nd))$ time for each graph by utilizing the sparsity of $A^{(m)}$. Since the evaluation of the objective and its gradient share some computational steps, we can actually compute them at the same time within one loop over the multiple graphs. The total time complexity is $O(Md(nnz + Nd))$.

## 4.3 Alternate Interpretations

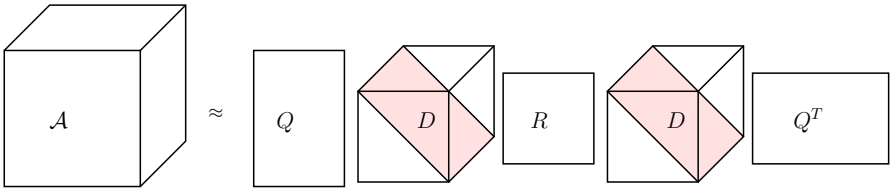
LMF can be viewed as a special case of low-rank tensor approximation and a relaxed version of the stochastic block model. Both interpretations shed light on understanding how information from different graphs is fused and how common structure is extracted in LMF.

### 4.3.1 Tensor Decomposition

LMF can be related to tensor decompositions of various types. To see this, we first re-arrange all adjacency matrices from all individual graphs into a third order tensor $\mathcal{A} \in \mathbb{R}^{N\times N\times M}$, where each frontal slice $A_{::m} = A^{(m)}$ for $m = 1, \ldots, M$. We first note that LMF on $\{A^{(1)}, \cdots, A^{(M)}\}$ can be viewed as a regularized variant of Tucker




(a) LMF interpreted as Tucker model




(b) 3-way DEDICOM model

Figure 3: An illustration of LMF, Tucker, and DEDICOM.

decomposition [15] with no compression in the third mode. A low-rank Tucker model for a general tensor $\mathcal{T} = (\mathbf{t}_{ij\tau}) \in \mathbb{R}^{M\times N\times L}$ is of the form

$$\mathcal{T} = (X, Y, Z) \cdot \mathcal{C}, \quad \mathbf{t}_{ij\tau} = \sum_{\lambda,\mu,\nu=1}^{d_1,d_2,d_3} \mathbf{c}_{\lambda\mu\nu}\mathbf{x}_{i\lambda}\mathbf{y}_{j\mu}\mathbf{z}_{\tau\nu},$$

where $X, Y, Z$ are $M \times d_1$, $N \times d_2$, $L \times d_3$ matrices, respectively, and $\mathcal{C} = (\mathbf{c}_{ij\tau})$ is a $d_1 \times d_2 \times d_3$ *core tensor*. In our case the tensor $\mathcal{A}$ is composed of multiple undirected (symmetric) graphs and we employ a symmetric Tucker model in first and second modes

$$\mathcal{A} \approx (P, P, I) \cdot \mathcal{C}.$$

The matrix $I$ multiplied in the third mode is simply the identity matrix. This is exactly the approximation used in LMF if we rewrite the $m^{th}$ frontal slice of core tensor $\mathcal{C}$ as $\Lambda^{(m)}$. See Figure 3(a) for an illustration.

DEDICOM (DEcomposition into DIrectional COMponents) is a decomposition model introduced by Harshman [6] for modeling directional (asymmetric) relationships.

In the three-way DEDICOM model, we are given multiple asymmetric relationship matrices $X_m,\ m = 1, \ldots, M$, and the decomposition is

$$X_m \approx QD^{(m)}RD^{(m)}Q^T \quad \text{for } m = 1, \ldots, M, \tag{7}$$

where $Q \in \mathbb{R}^{N\times d}$ specifies the latent components, $R \in \mathbb{R}^{d\times d}$ models the interactions between components, and the diagonal matrix $D^{(m)} \in \mathbb{R}^{d\times d}$ gives the weights of latent components specific for slice $A^{(m)}$, see Figure 3(b) for an illustration. Although the three-way DEDICOM is similar to LMF there are important differences: 1) DEDICOM uses a diagonal matrix $D^{(m)}$ to account for the difference between slices while LMF uses a more flexible symmetric matrix $\Lambda^{(m)}$, and 2) DEDICOM is generally used for modeling directed (asymmetric) data while LMF is tailored for undirected (symmetric) graphs.

#### 4.3.2 Stochastic Block Model

The stochastic block model (SBM) is widely used in modeling relational data [7], where we assume there are $C$ communities (called *blocks*), named $\mathcal{B}_1, \cdots, \mathcal{B}_C$. The probability of observing a relation (edge) between object $i$ and $j$ is

$$p_{ij} = \sum_{c_i=1}^{C}\sum_{c_j=1}^{C} p(c_i|i)p(c_j|j)q_{c_ic_j}$$

where $p(c_i|i)$ gives the soft membership of object $i$ to block $c_i$, and the probability $q_{c_i c_j}$ stands for the probability that an edge is formed between block $c_i$ and $c_j$. In other words, the blocks serve as latent variables, and the observed edges are ascribed by the interaction between these latent blocks. The task of learning is then to find the membership $p(\cdot|\cdot)$ and block interaction $q$, through maximizing the likelihood of the observed edges. One can generalize this stochastic block model to multiple types of relations, in which for each relation type we assume the same membership of vertices to blocks, but different block-interaction pattern. Thus, the probability of observing an edge between $i$ and $j$ in the $m^{th}$ type of relation is

$$p_{ij}^{(m)} = \sum_{c_i=1}^{C} \sum_{c_j=1}^{C} p(c_i|i)p(c_j|j)q_{c_i c_j}^{(m)},$$

which closely resembles the formulation for predicting the entry $(i, j)$ in the $m^{th}$ graph

$$\tilde{A}_{ij}^{(m)} = \sum_{n_i=1}^{d} \sum_{n_j=1}^{d} P_{in_i} P_{jn_j} \Lambda_{n_i n_j}^{(m)},$$

where $P$ specifies for each vertex the weights to each of the $d$ factors and $\Lambda^{(m)}$ coordinates the factors for the $m^{th}$ graph. It is easy to see that SBM for multiple relations essentially turns into LMF after the following relaxation and modification

- **Soft membership:**
  probability (SBM) $\rightarrow$ real numbers (LMF);
- **Block interaction:**
  probability (SBM) $\rightarrow$ real numbers (LMF);
- **Observation:**
  Bernoulli (SBM) $\rightarrow$ Gaussian (LMF).

Intuitively, the "soft membership" matrix $P$ from LMF (although not directly usable as a cluster indicator matrix), can still serve as features for all the vertices.

# 5 Semi-supervised Clustering Models

In many real-world applications, we often do not realistically expect the clustering to discover intended structure in a total unsupervised fashion. In those situations, we can often benefit from various types of weak supervision or side information. Here we consider the following two types of instance-level constraints on cluster assignments, which can naturally emerge in various situations [1]:

**must-link:** entity $i$ and entity $j$ are in the *same* cluster;

**cannot-link:** entity $i$ and entity $j$ are in *different* clusters.

There is a large body of work on using these *pairwise constraints* to boost the performance of clustering algorithms (see [1] for more details), but it has never been previously used in the context of combining multiple graph relations.

We consider two ways to incorporate these pairwise constraints into the clustering algorithm, both of which require viewing the unsupervised learning methods described in Sections 3 and 4 as means of feature extraction for vertices. This applies explicitly for LMF ( the features for vertex $i$ are given by the $i^{th}$ row of $P$), and implicitly for spectral kernel cases where the obtained kernel can be viewed as the inner product of feature vectors. The first semi-supervised method is metric learning, which directly adapts the distance metric in the corresponding feature space to fit the given pairwise constraints. The second method is to express the pairwise constraints as a penalty term in the unsupervised learning objective function, based on which the feature vectors for clustering are learned.

## 5.1 Metric Learning

Metric learning seeks a distance metric in feature space that fits our clustering or classification preference [5, 16]. Typically we learn a squared Mahalanobis distance

$$d(\mathbf{f}_i, \mathbf{f}_j) = (\mathbf{f}_i - \mathbf{f}_j)^T \Sigma (\mathbf{f}_i - \mathbf{f}_j),$$

or equivalently the positive definite matrix $\Sigma$. The general idea is to learn a metric so that distances between must-linked pairs are small, and distances between cannot-linked pairs are large. Among the metric learning models, we consider Information-Theoretic Metric Learning (ITML) [5] since it is scalable to handle millions of pairwise constraints. The ITML method learns the metric $\Sigma$ through the following optimization

$$\begin{aligned} \min_{\Sigma} \quad & D_{\ell d}(\Sigma, \Sigma_0) \\ \text{s.t.} \quad & (\mathbf{f}_i - \mathbf{f}_j)^T \Sigma (\mathbf{f}_i - \mathbf{f}_j) \le l, (i,j) \in \mathcal{M}, \\ & (\mathbf{f}_i - \mathbf{f}_j)^T \Sigma (\mathbf{f}_i - \mathbf{f}_j) \ge u, (i,j) \in \mathcal{C}, \\ & \Sigma \succeq 0, \end{aligned}$$

where $D_{\ell d}(\Sigma, \Sigma_0)$ denotes the log-determinant divergence between $\Sigma$ and $\Sigma_0$ [5], $l$ and $u$ are pre-determined scalars, and $\mathcal{M}$ and $\mathcal{C}$ stand respectively for the set of must-link and cannot-link constraints. The learning of $\Sigma$ can be performed rather efficiently through cyclic Bregman projections, which is a significantly faster method as compared to semi-definite programming and can readily handle up to millions of constraints.

Since metric learning model only learns a *linear* transformation in the feature space, its modeling capacity can be somewhat limited. Take the XOR data for example, one cannot find a linear transformation to the feature space to separate the data, and metric learning is hence futile in this case. One way to overcome this limitation is to introduce more modeling flexibility is by mapping data points into a (potentially infinite dimensional) space through a non-linear function $\phi(\cdot)$ and consider the inner product $K_{ij} =< \phi(\mathbf{f}_i), \phi(\mathbf{f}_j) >$ as the kernel. The kernelized ITML learns a new kernel $K$ based on a given $K_0$ through

$$\begin{aligned} \min_{K} \quad & D_{\ell d}(K, K_0) \\ \text{s.t.} \quad & K_{ii} + K_{jj} - 2K_{ij} \le l, (i,j) \in \mathcal{M}, \\ & K_{ii} + K_{jj} - 2K_{ij} \ge u, (i,j) \in \mathcal{C}, \\ & K \succeq 0. \end{aligned}$$

ITML can serve as a post-processing step for the unsupervised learning in Section 3 to incorporate the pairwise constraints. For example, the learned spectral kernels from the graphs can be used as the initial kernel $K_0$, and after that, kernel K-means is performed to obtain the clustering. For LMF, the rows of learned $P$ matrix will be treated as feature vectors , and ITML can be used to learn a kernel $K$ based either on linear kernel $K_0 = PP^T$ or a Gaussian kernel with rows of $P$ as the feature vectors for vertices.

## 5.2 Semi-supervised Feature Extraction

One limitation of the metric learning as a post-processing step is that it is often futile to correct the learned bad feature. One can often alleviate this by learning more "discriminative" features with the semi-supervision from the pairwise constraints. We consider the following objective function for feature $F \equiv \{\mathbf{f}_1, \cdots, \mathbf{f}_N\}$,

$$\mathcal{L}(F) = e(F, \{A^{(1)}, \cdots, A^{(M)}\}) + \gamma s(F; \mathcal{M}, \mathcal{C}), \tag{8}$$

where $e(F, \{A^{(1)}, \cdots, A^{(M)}\})$ stands for the "empirical error" term from unsupervised learning described in Section 3 (called *unsupervised term*), $s(F; \mathcal{M}, \mathcal{C})$ stands for the extra penalty term from given pairwise constraints (called *supervised term*), and the parameter $\gamma$ controls the balance between the two terms. The unsupervised term could either be the objective function for LMF as in (3), or the measurement of roughness associated with the (combined) graph

Laplacian as in (2). The supervised term is designed to ensure that the features of must-linked pairs are close and the features of cannot-linked pairs are far away from each other,

$$s(F;\mathcal{M},\mathcal{C}) = \sum_{(i,j)\in\mathcal{M}} \|\mathbf{f}_i - \mathbf{f}_j\|^2 - \sum_{(i,j)\in\mathcal{C}} \|\mathbf{f}_i - \mathbf{f}_j\|^2 \equiv \mathrm{Tr}(FL_pF^T),$$

where $L_p$ is defined as $D_p - A_p$ with $A_p$ being the "adjacency" matrix from pairwise constraints

$$A_{p,ij} = \begin{cases} -1 & (i,j) \in \mathcal{M}; \\ 1 & (i,j) \in \mathcal{C}; \\ 0 & \text{otherwise.} \end{cases}$$

and $D_p$ is the diagonal "degree" matrix with $D_{p,ii} = \sum_j A_{p,ij}$.

For the LMF model, the optimization for the semi-supervised objective can be performed using the same optimization routine with minimal change. For the graph spectral methods based on a graph Laplacian $L_p$, the objective function is essentially

$$\mathrm{Tr}(F\tilde{L}F^T) = \mathrm{Tr}(FLF^T) + \mathrm{Tr}(FL_pF^T),$$

where $L$ is the graph Laplacian of summation of graphs. The above problem can be solved by finding the eigenvectors of matrix $\tilde{L}$ with the smallest eigenvalues.

# 6 Experiments

In this section, we present results of clustering with multiple graphs on both synthetic and SIAM journal data. As we will show in the results, 1) we can improve the clustering performance by simultaneously modeling multiple sources of information, and 2) the pairwise constraints help in the semi-supervised learning scenario.

To evaluate the performance of LMF in the unsupervised setting, we compare it with several baseline methods introduced in Section 3: 1)SpecC: spectral clustering algorithm [10] on single graph; 2)mSpC-A: spectral clustering algorithm on the sum of adjacency matrices; 3) mSpC-B: spectral clustering algorithm on the sum of normalized adjacency matrices; 4)SpecK: sum of spectral kernels from each graph; 5)Consensus: consensus clustering with SpecC as the base component.

## 6.1 Synthetic Data

This synthetic 500-vertex example is designed to show that LMF can handle data sets where individual graphs have widely varying characteristics. The example consists of two similarity graphs over the same set of entities. The first graph contains complete information about cluster structure but has a lot of noise; while the second graph only contains partial information about the cluster structure. The spy plots of the synthetic data set are presented in Figure 4. The clustering performance measured in NMI and confusion matrix are presented in Figure 5. LMF is seen to be the best method, closely followed by mSpc–B.

## 6.2 SIAM Data: Unsupervised Clustering

The results of unsupervised clustering on SIAM-similar and SIAM-different are presented in Table 3, with each column representing a different combination of individual graphs. The results are consistent with our observation that SIAM-similar is a much harder problem than SIAM-different. We observe that LMF is the only method that can consistently benefit from including more graphs. LMF leads with 7 out of 8 graph combinations on both SIAM-different and SIAM-similar. Most importantly, the best performance from all models (NMI = $0.714$) is achieved by LMF with all five graphs.

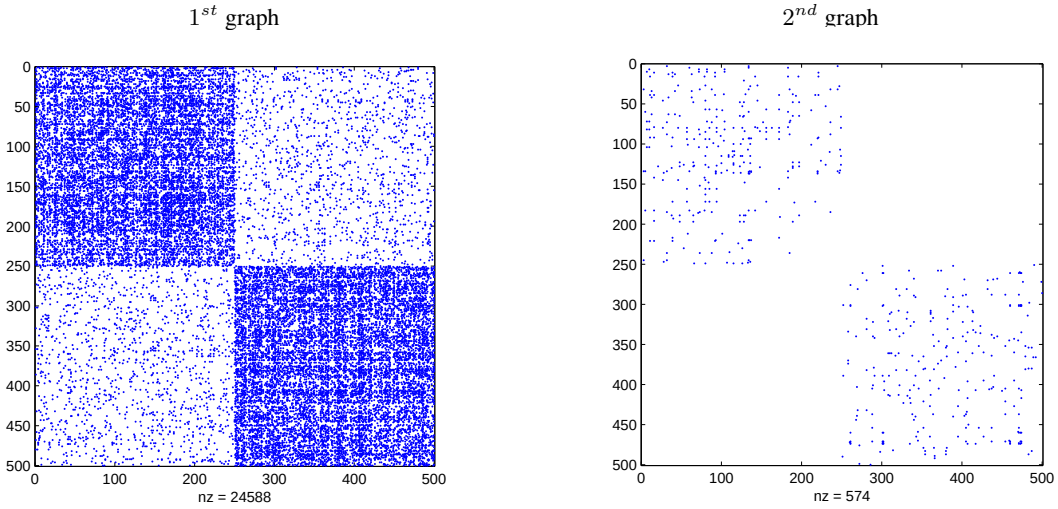



Figure 4: Spy plots of the synthetic data.

SpecC ($1^{st}$ graph)
NMI = 0.6422

| | |
|---|---|
| 235 | 15 |
| 19 | 231 |

SpecC ($2^{nd}$ graph)
NMI = 0.0003

| | |
|---|---|
| 100 | 150 |
| 95 | 155 |

mSpC–A
NMI = 0.6935

| | |
|---|---|
| 241 | 9 |
| 19 | 231 |

mSpC–B
NMI = 0.7492

| | |
|---|---|
| 241 | 9 |
| 12 | 238 |

SpecK
NMI = 0.6350

| | |
|---|---|
| 235 | 15 |
| 20 | 230 |

LMF
NMI = **0.7626**

| | |
|---|---|
| 240 | 10 |
| 10 | 240 |

Figure 5: The confusion matrices of competing methods, where rows represent the actual classes and columns the clusters.

**Choice of Rank & Regularization Parameter** In the above experiments, we set the parameters for LMF rather arbitrarily. Nevertheless it is meaningful to examine the impact of rank and regularization on the clustering performance based on LMF. Instead of varying both factors simultaneously, we will fix one and let the other vary. Figure 7 (left panel) shows the clustering performance on SIAM-different (averaged over 10 trials with different initializations) based on $P$ from LMF with rank varying from 10 to 50 and a fixed regularization parameter $\alpha = 0.5$. As seen in Figure 7, the performance of LMF is rather robust to the rank in the range 20 to 40. Similarly, Figure 7 right panel, shows the impact of regularization parameter $\alpha$ with a fixed rank 30. The LMF performance is rather stable with $\alpha$ from 0 to 0.8, with the range 0.55 to 0.8 being slightly better than the rest before the accuracy plunges. Although it is not shown here, a similar story holds for SIAM-similar, where the LMF performance is reasonably robust to the change of both rank and regularization parameter.

## 6.3 SIAM: Semi-supervised Clustering

In this section, we discuss semi-supervised clustering results on both SIAM data sets with pairwise constraints. As we will show, metric learning and semi-supervised feature extraction demonstrate different behavior, while both of them help in improving the clustering performance.

**Metric Learning** We considered kernelized metric learning with the following choices of initial kernel $K_0$: 1) SpC-ML: spectral kernel based on sum of adjacency matrices, 2) SpK-ML: sum of spectral kernels based on individual

SIAM-different

| | $\{1,2,3\}$ | $\{1,2,3,4\}$ | $\{1,2,3,5\}$ | $\{1–5\}$ |
|---|---|---|---|---|
| mSpC-A | **0.657** | 0.649 | 0.621 | 0.630 |
| mSpC-B | 0.626 | 0.683 | 0.684 | 0.701 |
| SpecK | 0.636 | 0.455 | 0.637 | 0.638 |
| Consensus | 0.587 | 0.597 | 0.559 | 0.444 |
| LMF | 0.611 | **0.698** | **0.689** | **0.714** |

SIAM-similar

| | $\{1,2,3\}$ | $\{1,2,3,4\}$ | $\{1,2,3,5\}$ | $\{1–5\}$ |
|---|---|---|---|---|
| mSpC-A | 0.226 | 0.238 | 0.229 | 0.238 |
| mSpC-B | 0.244 | 0.235 | 0.238 | 0.234 |
| SpecK | 0.237 | 0.240 | 0.237 | 0.237 |
| Consensus | 0.212 | 0.202 | 0.186 | 0.151 |
| LMF | **0.246** | **0.249** | **0.251** | **0.253** |

Table 3: Results of unsupervised clustering on SIAM-different and SIAM-similar data sets, measured in NMI. Each column represents a different combination of individual graphs. The results are averaged over ten trials.

SIAM-different

| mSpC–A | | | mSpC–B | | | SpecK | | |
|---|---|---|---|---|---|---|---|---|
| 245 | 8 | 7 | 242 | 7 | 11 | 0 | 231 | 29 |
| 30 | 593 | 33 | 17 | 603 | 36 | 0 | 606 | 50 |
| 57 | 6 | 281 | 19 | 6 | 319 | 22 | 23 | 299 |

| Consensus | | | **LMF** | | |
|---|---|---|---|---|---|
| 17 | 11 | 232 | **241** | 8 | 11 |
| 6 | 442 | 208 | 13 | **614** | 29 |
| 18 | 4 | 322 | 21 | 8 | **315** |

SIAM-similar

| mSpC–A | | | mSpC–B | | | SpecK | | |
|---|---|---|---|---|---|---|---|---|
| 268 | 69 | 274 | 289 | 74 | 248 | 319 | 74 | 218 |
| 3 | 405 | 15 | 7 | 405 | 11 | 14 | 404 | 5 |
| 176 | 249 | 231 | 211 | 224 | 221 | 239 | 206 | 211 |

| Consensus | | | **LMF** | | |
|---|---|---|---|---|---|
| 262 | 68 | 281 | **399** | 39 | 173 |
| 50 | 354 | 19 | 2 | **366** | 55 |
| 257 | 164 | 235 | 219 | 176 | **261** |

Figure 6: The confusion matrices of competing methods on the SIAM data set, where rows represent the actual classes and columns the clusters.

graphs, 3) LMF-ML(L): linear kernel $PP^T$ based on LMF feature $P$, and 4) LMF-ML(G): Gaussian kernel with rows of $P$ as feature vectors: $K_0(i,j) = \exp(-\|p_i - p_j\|^2/\sigma^2)$. Figure 8 presents the results of metric learning on four different kernels with the number of randomly selected constraints varying from 0 to 3000. As seen from the figure, the three linear kernels (SpC-ML, SpK-ML, and LMF-ML(L) ) do not respond well to the pairwise constraints, while the Gaussian kernel based on LMF feature (LMF-ML(G))(interestingly, the Gaussian kernel gives a much poorer result in the unsupervised case, as seen in the figure), can lead to superior performance after 2000 constraints. This behavior can be explained by the fact that the linear kernels, although informative by themselves have low rank and hence do not provide enough modeling flexibility for ITML, while the Gaussian kernel gives enough room for ITML to improve performance.

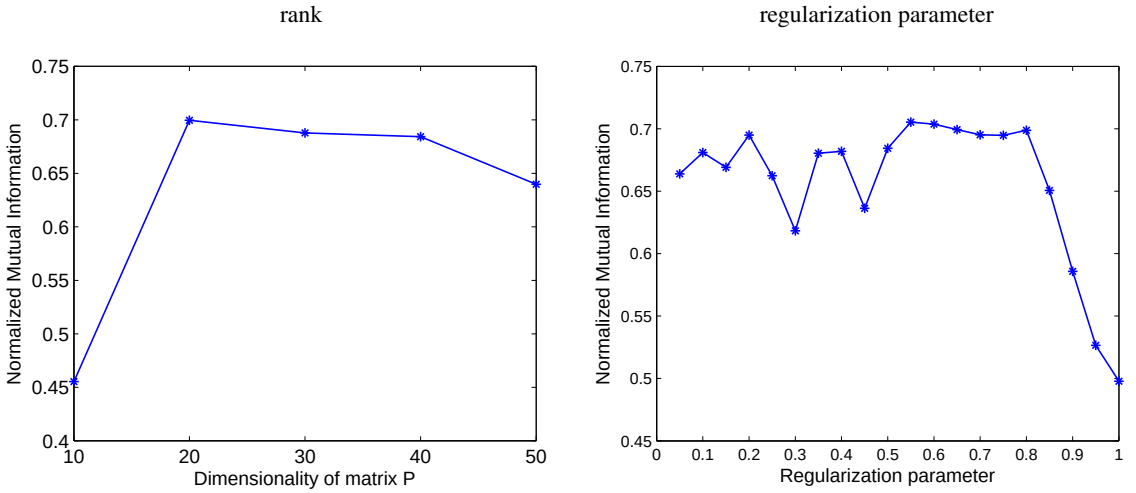



Figure 7: Clustering result of LMF on SIAM-different with varying rank and regularization parameter ($\alpha$).

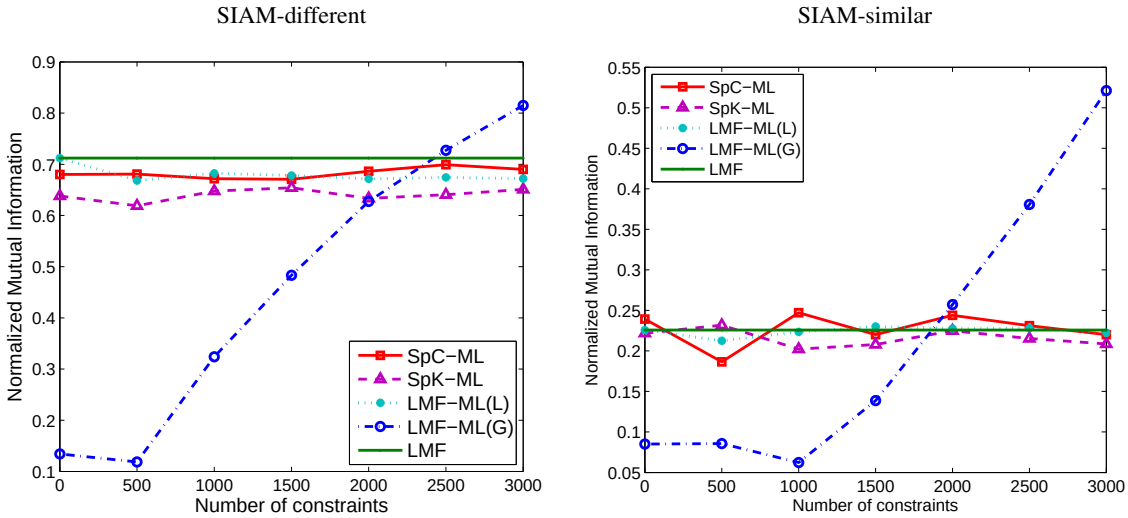



Figure 8: Semi-supervised clustering result on SIAM data with metric learning.

**Semi-supervised Feature Extraction** In this section, we report experimental results conducted to investigate the effectiveness of Semi-supervised Feature Extraction. As discussed in Section 5.2, we can have two semi-supervised feature extraction algorithms based on their unsupervised counterparts: 1)LMF-SSFE: feature extraction based on LMF, and 2)Eig-SSFE: based on the eigenvectors of graph Laplacian, as in (2). Figure 9 plots the clustering performance, measured in NMI, as a function of the number of constraints on SIAM-different and SIAM-similar data sets. As indicated by Figure 9, both algorithms can benefit from pairwise constraints, but LMF-SSFE does significantly better on SIAM-different and performs comparably to Eig-SSFE on SIAM-similar.

# 7 Related Work

**Multi-view Learning** In multi-view learning, each object is described from multiple aspects, called *views*, and the most common learning scheme is to find a classification that maximizes the agreement between different views. Most work in multi-view learning is for classification, where labeled samples are given [2, 11] and the unlabeled data are

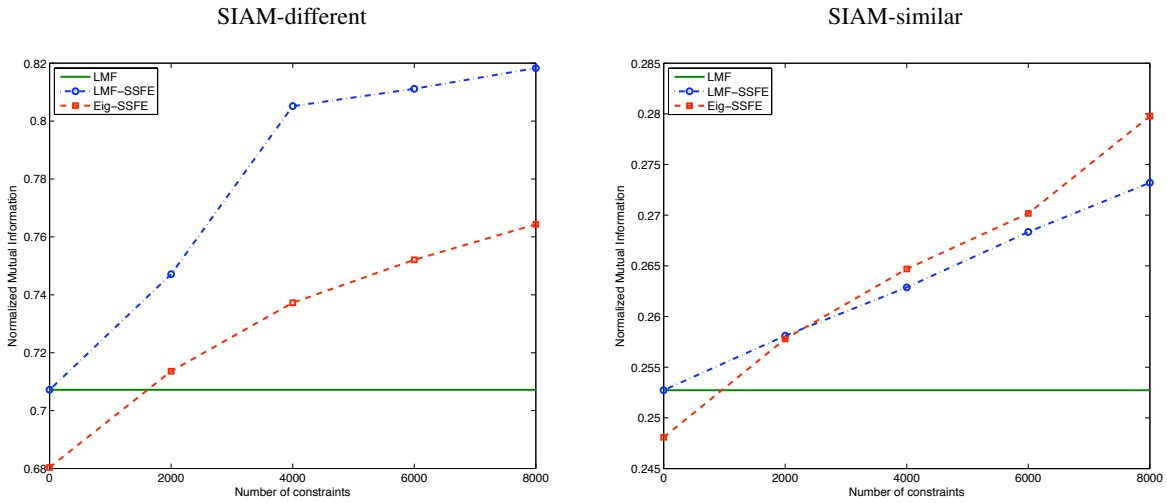



Figure 9: Semi-supervised clustering using semi-supervised feature extraction.

classified based on the average of the decision function from different views in either the inductive or transductive setting. Clustering based on multiple sources is a relatively new topic, and the published work is relatively rare, mainly because it is harder to enforce agreement between different views in a clustering scheme. Zhou et. al, [17] proposed an algorithm for graph partitioning based on multiple graphs, which is based on approximately minimizing the normalized cut objective function combined from all the graphs. Another direction is proposed by Chaudhuri et al [3] where multi-view clustering is performed by canonical correlation analysis (CCA) which extracts the most correlated direction shared by each view.

**Collective Matrix Factorization** Singh and Gordon [12] proposed a Collective Matrix Factorization method for relational learning, where the relational data are approximated by a set of matrix factorizations which share common factor matrix for the same set of entities between two relations. A similar work by Zhou et al. [18] considered combining multiple relations for document recommendation, where a single low-dimensional embedding of documents is learned through matrix factorization over multiple relational data. The major difference between LMF and these approaches is that we model the data as multiple undirected graphs over the same set of vertices while [12] and [18] model the relational data as multiple directed graphs connected by different sets of vertices. Another recent work by Koren et al. [8] considers combining the "who-rates-what" binary matrix with the rating matrix in matrix factorization for collaborative recommendation. Again, they model two different sets of entities in the graphs, which is different from our problem.

# 8 Conclusion and Discussion

In this paper, we discussed the general problem of clustering based on multiple similarity graphs in both unsupervised and semi-supervised settings. We extend several graph-based clustering methods to handle multiple graphs. As one of our major contribution, we proposed Linked Matrix Factorization (LMF) as a novel method for learning the characteristics common to all given graphs. Experiments show that: 1) in an unsupervised setting, LMF can effectively extract informative and reliable features for vertices, and yield better clustering performance than single graph methods and other graph-based models for combining multiple graphs, and 2) LMF, as a feature extraction model, responds fairly well to pairwise constraints.

However, several questions remained unanswered. The most obvious one is how to incorporate the reliability of different sources into the clustering model. When this information is available, it can be simply integrated into LMF by having the reconstruction error term from different graphs weighted accordingly. However, finding the weights

from the graph themselves, in both unsupervised and semi-supervised cases, remains an open problem. Techniques such as kernel-target alignment [4] have been developed by the machine learning community for weighing different information sources, but they typically work for supervised learning scenarios and from our experiments (not reported here), did not seem to help in our setting. Another direction, as suggested in [3], is to explicitly model the common characteristics of all sources through canonical correlation analysis of the feature vectors, which is in the same spirit as LMF. It will be interesting and useful to suitably extend these methods to the problem of multiple graphs, which will be one of the directions of our future research.

# References


[1] S. Basu, I. Davidson, and K. Wagstaff. *Constrained Clustering: Advances in Algorithms, Theory, and Applications*. Chapman & Hall/CRC, 2008.

[2] A. Blum and T. Mitchell. Combining labeled and unlabeled data with co-training. In *Proc. of COLT'98*, 1998.

[3] K. Chaudhuri, S. M. Kakade, K. Livescu, and K. Sridharan. Multi-view clustering via canonical correlation analysis. In *Proc. of ICML'09*, 2009.

[4] N. Cristianini, J. Kandola, A. Elisseeff, and J. Shawe-Taylor. On kernel-target alignment. In *Proc. of NIPS 14*, 2002.

[5] J. Davis, B. Kulis, P. Jain, S. Sra, and I. Dhillon. Information-theoretic metric learning. In *Proc. of ICML'07*, 2007.

[6] R. Harshman. Models for analysis of asymmetrical relationships among n objects or stimuli. In *First Joint Meeting of the Psychometric Society and the Society for Mathematical Psychology*, Hamilton, Ontario, Canada, August 1978. McMaster University.

[7] C. Kemp, T. L. Griffiths, and J. B. Tenenbaum. Discovering latent classes in relational data. Technical report, 2004.

[8] Y. Koren. Factorization meets the neighborhood: a multifaceted collaborative filtering model. In *Proc. of SIGKDD'08*, pages 426–434, 2008.

[9] Jorge Nocedal and Stephen J. Wright. *Numerical Optimization*. Springer, 1999.

[10] J. Shi and J. Malik. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22:888–905, 1997.

[11] V. Sindhwani and D. S. Rosenberg. An rkhs for multi-view learning and manifold co-regularization. In *Proc. of ICML'08*, pages 976–983, 2008.

[12] A. P. Singh and G. J. Gordon. Relational learning via collective matrix factorization. In *Proc. of SIGKDD'08*, 2008.

[13] A. Smola and R. Kondor. Kernels and regularization on graphs. In *Proc. of COLT'03*, 2003.

[14] A. Strehl and J. Ghosh. Cluster ensemble - a knowledge reuse framwork for combining multiple partitions. *JMLR*, 3:583–617, 2003.

[15] L. Tucker. Some mathematical notes on three-mode factor analysis. *Psychometrika*, 31(3):279–311, 1953.

[16] E. P. Xing, A. Y. Ng, M. I. Jordan, and S. Russell. Distance metric learning, with application to clustering with side-information. In *Proc. of NIPS 15*, volume 15, 2003.

[17] D. Zhou and C. Burges. Spectral clustering and transductive learning with multiple views. In *Proc. of ICML'07*, 2007.

[18] D. Zhou, S. Zhu, K. Yu, X. Song, B. L. Tseng, H. Zha, and C. L. Giles. Learning multiple graphs for document recommendations. In *Proc. of WWW'08*, pages 141–150, 2008.