2010 IEEE International Conference on Data Mining

Efficient Semi-supervised Spectral Co-clustering with Constraints

Xiaoxiao Shi†, Wei Fan*, Philip S. Yu†
†*Department of Computer Science, University of Illinois at Chicago*
**IBM T.J.Watson Research Center*
{*xshi9,psyu*}*@uic.edu, weifan@us.ibm.com*

***Abstract*—Co-clustering was proposed to simultaneously cluster objects and features to explore inter-correlated patterns. For example, by analyzing the blog click-through data, one finds the group of users who are interested in a specific group of blogs in order to perform applications such as recommendations. However, it is usually very difficult to achieve good co-clustering quality by just analyzing the object-feature correlation data due to the sparsity of the data and the noise. Meanwhile, one may have some prior knowledge that indicates the internal structure of the co-clusters. For instance, one may find user cluster information from the social network system, and the blog-blog similarity from the social tags or contents. This prior information provides some supervision toward the co-cluster structures, and may help reduce the effect of sparsity and noise. However, most co-clustering algorithms do not use this information and may produce unmeaningful results. In this paper we study the problem of finding the optimal co-clusters when some objects and features are believed to be in the same cluster a priori. A matrix decomposition based approach is proposed to formulate as a trace minimization problem, and solve it efficiently with the selected eigenvectors. The asymptotic complexity of the proposed approach is the same as co-clustering without constraints. Experiments include graph-pattern co-clustering and document-word co-clustering. For instance, in graph-pattern data set, the proposed model can improve the normalized mutual information by as much as 5.5 times and 10 times faster than two naive solutions that expand the edges and vertices in the graphs.**

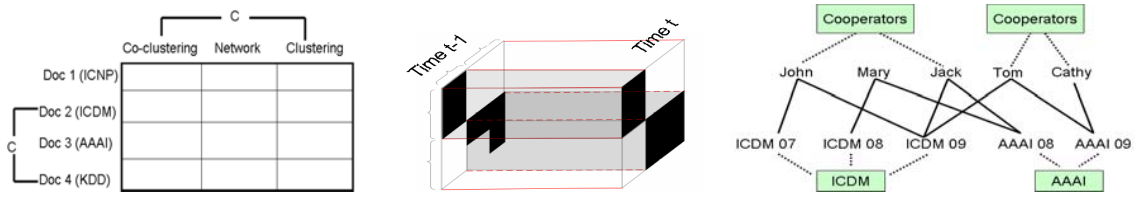
I. Introduction

Since clusters could exist in different subspaces of the feature space, co-clustering was proposed to simultaneously cluster objects and features, e.g., [1], [3]. From a matrix view point, a co-clustering algorithm simultaneously finds and links partitions on both row and column dimensions. However, most of the traditional co-clustering models mainly work in a completely unsupervised setting that does not take prior knowledge into consideration. One challenging problem is thus: if some rows or columns are known to be in the same cluster a priori, how does the algorithm find the optimal co-clusters consistent with the prior knowledge? Examples include:

- Object-feature co-clustering: We take document-word data as an example (Fig. 1(a)). Some documents are known to belong to the same category, while at the same time, some words are believed to have similar semantics. For instance, in computer science papers, those published in ICDM and KDD may belong to the same cluster, and "co-clustering" and "clustering" are known to be similar words.
- Evolutionary co-clustering (Fig. 1(b)): Applications such as product-user analysis, network traffic data analysis are usually modeled in the streaming environment. In this scenario, we are continuously given the up to date data that evolves over time. In this case, the co-clustering result at time t is assumed to evolve smoothly from time $t-1$. Technically, the co-clustering result at time t is usually constrained to be similar as that of time $t-1$.
- Social network mining via heterogeneous relational links (Fig. 1(c)): For instance, in the application of "author-conference" co-clustering, one wants to explore groups of authors that are active in some groups of conferences. Usually, we are given a bipartite graph indicating which author publishes paper in which conference. However, we can also find the links among the authors by the co-authorship, as well as the links among the conferences via the names or the research topics of the conferences. All the information can be transformed as row constraints and column constraints to improve the co-clustering result (Fig. 1(c)).

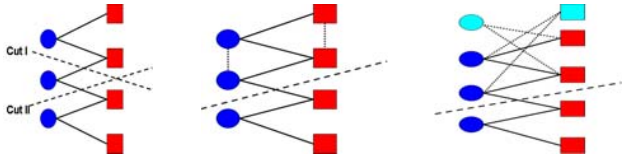
To solve applications such as the above three examples, a semi-supervised spectral co-clustering approach is introduced. Consider co-clustering in a bipartite graph as shown in Fig. 2(a). There can be several optimal partition assignments such as "Cut I" and "Cut II". Now suppose the top two ●s are believed to be in one group, and the top two ■s are together according to some background knowledge. To find the optimal co-clusters under these constraints, the challenge is on how to incorporate the constraints effectively without introducing much complexity to the co-clustering model. We first introduce two naive solutions to capture the constraints. One is a "link-based" method (LCM) that generates extra links between the vertices constrained together (Fig. 2(b)), and regard the co-clustering as a full graph partition problem. The second method is a "vertex-based" method (VCM) which generates an extra vertex for each constraint, and links the new vertex with the constrained vertices (Fig. 2(c)). Though the two approaches are straightforward, they are not efficient because the graph becomes more complicated and the problem size increases. We further propose SCM

1550-4786/10 $26.00 © 2010 IEEE
DOI 10.1109/ICDM.2010.64



(a) Document-word co-clustering (b) Evolutionary co-clustering (c) Co-clustering with various links

Figure 1. Examples of constrained co-clustering



(a) Co-clustering without Constraints (b) Link based Constraint Modeling (LCM) (c) Vertex based Constraint Modeling (VCM)

Figure 2. Two naive constraint models.

(spectral constraint modeling) to find the optimal co-clusters by incorporating penalty to the co-clustering assignments that violate the constraints. It is further formulated as a new trace minimization problem. Though finding the globally optimal solution is NP-complete, a relaxation of the discrete optimization problem can be efficiently solved. Experiments on graph-pattern co-clustering and document-word co-clustering show that the two naive solutions and SCM outperform the non-constrained approaches and a row-constrained co-clustering method. More importantly, SCM can achieve better results more efficiently. For example, when VCM achieves the cluster purity about 0.64 on one of the graph-pattern data sets, SCM achieves not only a better cluster purity of 0.89 but also 10 times faster.

II. Problem Formulation

We use small-bold letters such as $\mathbf{x}, \mathbf{u}, \mathbf{v}$ as column vectors, capital-bold letters such as $\mathbf{E}, \mathbf{M}, \mathbf{C}$ as matrices, and script letters such as $\mathcal{V}_r, \mathcal{D}, \mathcal{W}$ as vertex sets. $[\mathbf{E}]_{ij}$ denotes the (i,j)-th element of $\mathbf{E}$.

Co-clustering can usually be transformed as a partition problem on a bipartite graph. Denote the bipartite graph $G = (\mathcal{V}_r, \mathcal{V}_c, \mathbf{E})$ containing two sets of vertices $\mathcal{V}_r$ and $\mathcal{V}_c$. For convenience of discussion, we also call the vertices in $\mathcal{V}_r$ as "row vertices" as the documents (rows) in Fig. 1(a), while vertices in $\mathcal{V}_c$ as "column vertices" as the words (columns) in Fig. 1(a). Moreover, $[\mathbf{E}]_{ij}$ is the edge weight between the node $v_i \in \mathcal{V}_r$ and $v_j \in \mathcal{V}_c$. The adjacency matrix $\mathbf{M}$ of a bipartite graph is:

$$\mathbf{M} = \begin{bmatrix} \mathbf{0} & \mathbf{E} \\ \mathbf{E}^\top & \mathbf{0} \end{bmatrix} \tag{1}$$

Co-clustering A co-clustering of a bipartite graph $G = (\mathcal{V}_r, \mathcal{V}_c, \mathbf{E})$ is to partition the row vertices $\mathcal{V}_r$ into k groups and the column vertices $\mathcal{V}_c$ into l groups. We assume $k = l$ in the following discussion, in order to model the sub-clusters as bipartite graphs. But the proposed models can also handle the case when $k \neq l$ by a simple parameter setting, which is discussed in Section III-B. In this case, a co-clustering of a bipartite graph $G = (\mathcal{V}_r, \mathcal{V}_c, \mathbf{E})$ is to partition the graph into k sub-graphs $G_1, G_2, \cdots, G_k$ where each sub-graph is also a bipartite graph $G_i = (\mathcal{V}_{ri}, \mathcal{V}_{ci}, \mathbf{E}_i)$ for $i = 1, 2, \cdots, k$, and $\mathcal{V}_{ri} \subseteq \mathcal{V}_r, \mathcal{V}_{ci} \subseteq \mathcal{V}_c$. Traditional spectral co-clustering is to minimize the edge weights of the vertices in different sub-graphs by solving an eigenvalue system. More specifically, the solution is found by the left and right eigenvectors of the matrix $\mathbf{D}_r^{-1/2}\mathbf{E}\mathbf{D}_c^{-1/2}$ where $[\mathbf{D}_r]_{ii} = \sum_j \mathbf{E}_{ij}$ and $[\mathbf{D}_c]_{ii} = \sum_j \mathbf{E}_{ji}$.

Incorporating Constraints Assume that some vertices are believed to belong to the same cluster, one thus expects the co-clustering result to be consistent with the prior knowledge. We first model the prior knowledge with a "must-link" constraint matrix $\mathbf{C}$ as

$$[\mathbf{C}]_{ij} = \begin{cases} 1 & \text{if } v_i \text{ and } v_j \text{ are to be in the same cluster} \\ 0 & \text{otherwise.} \end{cases} \tag{2}$$

In the above equation, the vertex v_i and v_j can be either from vertex set $\mathcal{V}_r$ or $\mathcal{V}_c$. We decompose the constraint matrix $\mathbf{C}$ as

$$\mathbf{C} = \begin{bmatrix} \mathbf{C}_{rr} & \mathbf{C}_{rc} \\ \mathbf{C}_{cr} & \mathbf{C}_{cc} \end{bmatrix} \tag{3}$$

where $\mathbf{C}_{rr}$ denotes the constraints of row vertices that are both in $\mathcal{V}_r$, and $\mathbf{C}_{rc}$ denotes the constraints of vertices with one in $\mathcal{V}_r$ and the other in $\mathcal{V}_c$. $\mathbf{C}_{cr}$ and $\mathbf{C}_{cc}$ are defined similarly and $\mathbf{C}_{cr} = \mathbf{C}_{rc}^\top$. Given a bipartite graph $G = (\mathcal{V}_r, \mathcal{V}_c, \mathbf{E})$, the co-clustering constraint is to maximize the following function:

$$\max_{G_1, G_2, \cdots, G_k} \sum_{v_i, v_j \in G_p} [\mathbf{C}]_{ij} \tag{4}$$

The global optimization is thus to minimize the following function:

$$\min_{G_1, G_2, \cdots, G_k} \sum_{v_i \in G_p, v_j \in G_q, p \neq q} \mathbf{E}_{ij} - \delta \sum_{v_i, v_j \in G_p} [\mathbf{C}]_{ij} \tag{5}$$

where δ is a constraint confidence parameter to regulate the importance of the constraints. To solve the above optimization problem, three approaches are introduced, which are different in how to model the constraints.

III. Spectral Constrained Co-clustering

Two naive constrained co-clustering models are first derived by changing the structure of the original graph, followed by the third approach which models the constraints as a matrix trace minimization problem.

A. Two naive solutions

The co-clustering constraints can be incorporated by directly modifying the bipartite graph.

Constraints as Additional Links Given a bipartite graph $G = (\mathcal{V}_r, \mathcal{V}_c, \mathbf{E})$, if vertex v_1 and v_2 are preferred to be together, we add an edge, or increase the edge weight between v_1 and v_2 as shown in Fig. 2(b). From a matrix point of view, the adjacency matrix becomes:

$$\mathbf{M}' = \begin{bmatrix} \delta\mathbf{C}_{rr} & \mathbf{E} + \delta\mathbf{C}_{rc} \\ (\mathbf{E} + \delta\mathbf{C}_{rc})^\top & \delta\mathbf{C}_{cc} \end{bmatrix} \tag{6}$$

Note that in this case, the graph is no longer a bipartite graph since there may be links between any two vertices. In this case, traditional spectral co-clustering [1] cannot solve the problem directly, and we have to perform spectral partition on the whole graph. Take Ncut [5] as an example. The selected k eigenvectors of the matrix $\mathbf{D}^{-1/2}(\mathbf{D}-\mathbf{M}')\mathbf{D}^{-1/2}$ are used to give the solution, where $[\mathbf{D}]_{ii} = \sum_j \mathbf{M}'_{ij}$.

Constraints as Vertices Another solution is to model the constraints as pseudo vertices. Intuitively, for each constraint, we generate a pseudo vertex and link the pseudo vertex with the constrained vertices as shown in Fig. 2(c). More specifically, the following process is performed in order to represent the modified graph more conveniently. Suppose we have r row vertices and c column vertices. The algorithm then generates c pseudo row vertices and r pseudo column vertices. If two row vertices v_i and v_j are constrained to be together, we then link v_i to the j-th pseudo column vertex, and v_j to the i-th pseudo column vertex, similarly for the constrained column vertices. From a matrix view point, given a bipartite graph $G = (\mathcal{V}_r, \mathcal{V}_c, \mathbf{E})$, it changes to another bipartite graph $\hat{G} = (\mathcal{V}'_r, \mathcal{V}'_c, \hat{\mathbf{E}})$ where

$$\hat{\mathbf{E}} = \begin{bmatrix} \mathbf{E} + \delta\mathbf{C}_{rc} & \delta\mathbf{C}_{rr} \\ \delta\mathbf{C}_{cc} & \mathbf{0} \end{bmatrix} \tag{7}$$

In this new formula, the new graph $\hat{G}$ is still a bipartite graph (Fig. 2(c)), one can apply traditional spectral co-clustering to obtain the result. For both methods, the main computational cost is to calculate the eigenvectors of certain matrix. Consider Lanczos method to compute the eigenvectors. The complexity of both algorithms are $O\big(kN(|\mathcal{V}_r| + |\mathcal{V}_c|)^2\big)$ where k is the number of eigenvectors desired, N is the number of Lanczos iteration steps. $(|\mathcal{V}_r|+|\mathcal{V}_c|)^2$ is the upper bound of the nonzero entries of the matrix $\mathbf{M}$ and $\hat{\mathbf{E}}$. In the next section, a spectral approach is proposed to directly model the constraints into the formula with a more efficient implementation.

B. Constrained Co-clustering as Trace Minimization (The SCM Approach)

In this section, we introduce the spectral constraint modeling (SCM) algorithm that directly models the objective as trace minimization problem. First of all, given a bipartite graph $G = (\mathcal{V}_r, \mathcal{V}_c, \mathbf{E})$, define the co-clustering partition matrix $\mathbf{X}$ as

$$\mathbf{X} = \begin{bmatrix} \mathbf{X}_r \\ \mathbf{X}_c \end{bmatrix} \tag{8}$$

where $\mathbf{X}_r$ is the partition on row vertex set $\mathcal{V}_r$, and $\mathbf{X}_c$ is the partition on column vertex set $\mathcal{V}_c$. The entry $[\mathbf{X}_r]_{ij} = 1$ if and only if the row vertex v_i belongs to cluster j. The normalized cut [5] on the bipartite graph is to minimize the following function:

$$\min_{\mathbf{X}} \mathbf{tr}\big(\mathbf{X}^\top \mathbf{L}\mathbf{X}\big) \tag{9}$$

where

$$\mathbf{L} = \mathbf{D} - \mathbf{M} \tag{10}$$

$$\mathbf{D} = \begin{bmatrix} \mathbf{D}_r & \mathbf{0} \\ \mathbf{0} & \mathbf{D}_c \end{bmatrix} \tag{11}$$

and $\mathbf{D}_r$ and $\mathbf{D}_c$ are diagonal matrices such that $[\mathbf{D}_r]_{ii} = \sum_j \mathbf{E}_{ij}$ and $[\mathbf{D}_c]_{ii} = \sum_j \mathbf{E}_{ji}$. The matrix $\mathbf{L}$ is called the Laplacian of the graph [5] that has several advantages such as symmetric and positive semidefinite. Inspired by Eq. (9), we next model the co-clustering constraints as a trace norm minimization problem.

Theorem 1 *Given a bipartite graph $G = (\mathcal{V}_r, \mathcal{V}_c, \mathbf{E})$ and its constraint matrix $\mathbf{C}$ defined as Eq. (2), the co-clustering constraint in Eq. (4) is equivalent to minimize the following function:*

$$\max_{G_1, G_2, \cdots, G_k} \sum_{v_i, v_j \in G_p} \mathbf{C}_{ij} = \min \mathbf{tr}\Big(\mathbf{X}^\top(\mathbf{S} - \mathbf{C})\mathbf{X}\Big) \tag{12}$$

where $\mathbf{S}$ is a diagonal matrix such that $[\mathbf{S}]_{ii} = \sum_j \mathbf{C}_{ij}$.

Owing to limited space, we do not present the complete proof of Theorem 1. But it is straightforward to see that it has a similar structure as graph Laplacian. With Theorem 1, the optimization objective in Eq. (5) can thus be written as:

$$\begin{aligned} &\min_{G_1, G_2, \cdots, G_k} \sum_{v_i \in G_p, \bar{v_j} \in G_q, p \neq q} \mathbf{E}_{ij} - \delta \sum_{v_i, v_j \in G_p} \mathbf{C}_{ij} \\ &= \min_{\mathbf{X}} \mathbf{tr}\big(\mathbf{X}^\top \mathbf{L}\mathbf{X}\big) + \delta \mathbf{tr}(\mathbf{X}^\top(\mathbf{S} - \mathbf{C})\mathbf{X}) \\ &= \min_{\mathbf{X}} \mathbf{tr}\Big(\mathbf{X}^\top(\mathbf{L} + \delta\mathbf{S} - \delta\mathbf{C})\mathbf{X}\Big) \end{aligned} \tag{13}$$

Input: Bipartite edge weight matrix $\mathbf{E}$; Constraint matrix $\mathbf{C}$; Constraint confident parameter δ; Cluster number k

Output: Row partition matrix $\mathbf{X}_r$; Column partition matrix $\mathbf{X}_c$

1 Construct the diagonal matrices $\mathbf{D}_r, \mathbf{D}_c$ where $[\mathbf{D}_r]_{ii} = \sum_j \mathbf{E}_{ij}$ and $[\mathbf{D}_c]_{ii} = \sum_j \mathbf{E}_{ji}$.

2 Calculate $\mathbf{A}$ as defined in Eq. (16).

3 Perform SVD on matrix $\mathbf{A}$. Denote the left and right eigenvector of the 2nd to the $(k+1)$-th eigenvalues as $\mathbf{U}$ and $\mathbf{V}$ respectively.

4 Construct $\mathbf{Z}_r = (\mathbf{D}_r - \delta\mathbf{C}_{rr})^{-1/2}\mathbf{U}$ and $\mathbf{Z}_c = (\mathbf{D}_c - \delta\mathbf{C}_{cc})^{-1/2}\mathbf{V}$.

5 Run k-means on $\mathbf{Z}_r$ to get the row partition matrix $\mathbf{X}_r$; similarly get $\mathbf{X}_c$ from $\mathbf{Z}_c$ (different number of row clusters and column clusters can be set in k-means).

Algorithm 1: Spectral Constraint Modeling (SCM)

Theorem 2 *Given a bipartite graph $G = (\mathcal{V}_r, \mathcal{V}_c, \mathbf{E})$, denote the constraint matrix $\mathbf{C}$ as*

$$\mathbf{C} = \begin{bmatrix} \mathbf{C}_{rr} & \mathbf{C}_{rc} \\ \mathbf{C}_{cr} & \mathbf{C}_{cc} \end{bmatrix} \tag{14}$$

where $\mathbf{C}_{rr}, \mathbf{C}_{cc}$ denote the row vertex constraints and column vertex constraints respectively; $\mathbf{C}_{rc}$ and $\mathbf{C}_{cr}$ denote the constraints between row vertices and column vertices and $\mathbf{C}_{rc} = \mathbf{C}_{cr}^\top$. Eq. (13) can be solved via

$$\mathbf{X} = \begin{bmatrix} (\mathbf{D}_r + \delta\mathbf{S}_r - \delta\mathbf{C}_{rr})^{-\frac{1}{2}}\mathbf{u} \\ (\mathbf{D}_c + \delta\mathbf{S}_c - \delta\mathbf{C}_{cc})^{-\frac{1}{2}}\mathbf{v} \end{bmatrix} \tag{15}$$

where $\mathbf{D}_r, \mathbf{D}_c, \mathbf{S}_r, \mathbf{S}_c$ are diagonal matrices defined as:

$$[\mathbf{D}_r]_{ii} = \textstyle\sum_j \mathbf{E}_{ij} \qquad [\mathbf{D}_c]_{ii} = \textstyle\sum_j \mathbf{E}_{ji}$$
$$[\mathbf{S}_r]_{ii} = \textstyle\sum_j [\mathbf{C}_{rr}]_{ij} + \sum_t [\mathbf{C}_{rc}]_{it}$$
$$[\mathbf{S}_c]_{ii} = \textstyle\sum_j [\mathbf{C}_{cc}]_{ij} + \sum_t [\mathbf{C}_{cr}]_{it}$$

And $\mathbf{u}$, $\mathbf{v}$ are the left and right eigenvectors of a matrix $\mathbf{A}$ defined as

$$\mathbf{A} = (\mathbf{D}_r + \delta\mathbf{S}_r - \delta\mathbf{C}_{rr})^{-\frac{1}{2}}(\mathbf{E} + \delta\mathbf{C}_{rc})(\mathbf{D}_c + \delta\mathbf{S}_c - \delta\mathbf{C}_{cc})^{-\frac{1}{2}} \tag{16}$$

The complete proof of Theorem 2 can be derived from Eq. 13. According to Theorem 2, the left and right eigenvectors of the matrix $\mathbf{A}$ in Eq. (16) can be applied to get the result, which we also describe in Algorithm 3. Note that the main computational cost is to perform SVD on the matrix $\mathbf{A}$ on Step 3. However, the matrix $\mathbf{A}$ is of the same size as the edge weight matrix $\mathbf{E}$, which means the algorithm complexity is the same as the traditional spectral co-clustering [1] in the worst case. Also consider Lanczos method to compute the eigenvectors. The complexity of the algorithm is thus $O(kN|\mathcal{V}_r||\mathcal{V}_c|)$ where k is the number of eigenvectors desired, N is the number of Lanczos iteration steps, and $|\mathcal{V}_r||\mathcal{V}_c|$ is the upper bound of the nonzero entries of $\mathbf{A}$. Thus, SCM is more efficient than LCM and VCM especially when $|\mathcal{V}_r| \gg |\mathcal{V}_c|$ or $|\mathcal{V}_r| \ll |\mathcal{V}_c|$. Furthermore, SCM is more effective to use the constraints because it directly models them into the optimization objective, instead of modeling them as more complicated graphs as LCM and VCM. More performance in detail are studied in the next section.

Table I
DESCRIPTION OF THE GRAPH DATA

IDX	Tasks	#Graphs	#Patterns
1	COX2 vs. Thrombin	195	118
2	COX2 vs. Estrogen	183	159
3	COX2 vs. MMP	171	157
4	MMP vs. Estrogen	98	128
5	MMP vs. Thrombin	110	143
6	Estrogen vs. Thrombin	122	151
7	4 classes	293	172

IV. EXPERIMENT RESULTS

We empirically analyze SCM in two real-world data sets.

A. Graph-pattern Co-clustering

The proposed method is next evaluated on a chemical graph data set (Stahl Data Set [6]). In this data set, each of the objects is represented as a chemical graph of which the nodes are some chemical atoms such as Carbon and the edges represent the connections between atoms. Each graph also has a label to indicate the name of the corresponding chemical compound. Note that we first transform the graph format data into vector based format by the following procedure: (1) mine frequent sub-graphs by the software GSpan [7] with minimum support as 0.2; (2) represent each graph as a tuple where the i-th entry indicates whether or not the graph contains the i-th subgraph patterns mined in Step (1). Following this process, the graph data set can be transformed into a "graph-pattern" vector-based data set, where the rows represent different graphs and the columns represent different frequent patterns. It is important to mention that the bipartite graph generated from this data set is dense because the frequent patterns are generated with a relatively high minimum support. The row vertices and column vertices can thus have many links. We will also study the methods on a sparse graph in Section IV-B.

Note that our goal is to give partition of the graph instances as shown in Table I. The results are evaluated by cluster purity ($\mathbf{Purity} = \sum_c \max_l \frac{|c \cap l|}{|c|}$) and normalized mutual information ($\mathbf{NMI} = \sum_c \sum_l p(c,l)\mathbf{log}\frac{p(c,l)}{p(c)p(l)}$), where c is a cluster and l represents a class label. Moreover, we also compare the algorithm running time of the three constrained based co-clustering models on a PC with CPU as Core 2 Duo 2.4G, and 3 GB memory.

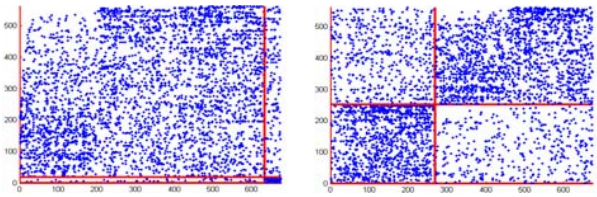
Construct graph-pattern co-clustering constraints 5% of the graphs are randomly drawn and their labels are

assumed observable to construct row constraints. In other words, those graphs with the same class label are constrained together. Furthermore, we use the edit distance of the patterns to construct the column constraints. For example, the edit distance of O=C=C=O and H2-C=O is 2 because at least two atoms have to be replaced to make the two graphs the same. In this case, similar patterns have small edit distance. We thus perform clustering on the frequent patterns by their edit distances, and constrain the patterns together if they are in the same cluster. For the number of clusters, we simply set it the same as the number of class labels. Note that how to choose the optimal number of clusters is a nontrivial model selection problem but beyond the scope of this paper.

Six approaches are applied to present the experiment results on the seven data sets (Fig. 3). The comparison models include two state-of-the-art approaches: the information-theoretic co-clustering [2] (denoted as "Infor-CO") and the traditional spectral co-clustering [1] (denoted as "S-CO"). The third comparison algorithm is the co-clustering with only the row constraint or label constraint. The other three approaches are LCM (Algorithm 1), VCM (Algorithm 2) and the proposed SCM (Algorithm 3). Recently in [4], a novel co-clustering method is proposed with an order-related constraint especially fit for microarray data sets. But the approach is not directly comparable to ours because it requires order-related constraints. The results in Fig. 3 are summarized over 10 runs with standard deviation and in each run the 5% of constraints are re-sampled. It can be observed from Fig. 3 that SCM can consistently achieve better results. For instance, in the first data set, it improves the normalized mutual information by as much as 1.5 times, while in the 6th data set, the improvement is as much as 5.5 times and in the 7th data set with multiple clusters, the improvement is about 1.2 times (compared with the two non-constrained methods). This is because the chemical compound graphs are decomposed and represented only by their frequent patterns, which may cause information loss and induce some noise. Constraints can effectively help reduce the noise and improve the co-clustering quality. In addition, in the first data set, the cluster purity of the method using only the row constraints is about 72%, while SCM achieves a cluster purity of 90%, which demonstrates the necessity to include column constraints to improve clustering quality. Among the three constrained co-clustering approaches, SCM can achieve far superior clustering quality especially on the first and the forth data sets. This is because when the bipartite graph is dense, LCM and VCM are not effective to represent the constraints as additional links or nodes on a graph that already has complicated links. Furthermore, among the spectral methods, SCM is also very efficient (at least 10 times faster than VCM according to Fig. 3(c)) because it maintains the algorithm complexity as the traditional spectral co-clustering [1].

Table II
DESCRIPTION OF THE TEXT DATA

IDX	Classes	#Documents	#Words
1	Comp vs. Rec	982	358
2	Comp vs. Sci	786	441
3	Comp vs. Talk	897	536
4	Rec vs. Sci	793	492
5	Rec vs. Talk	734	569
6	Sci vs. Talk	675	561
7	Comp vs. Rec vs. Sci vs. Talk	1657	781



(a) No constraint (b) With 5% of row constraints

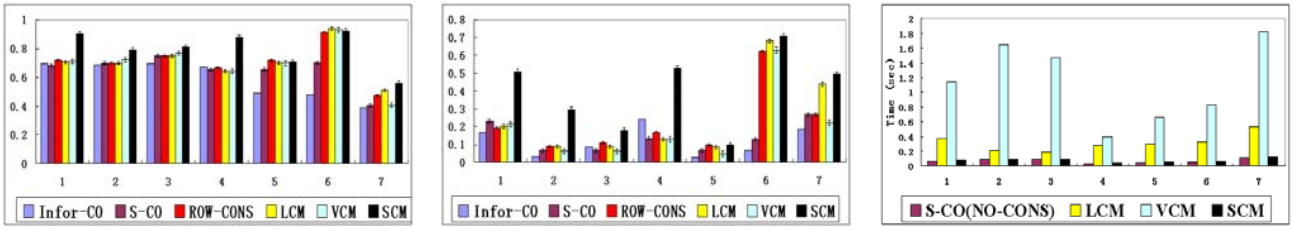
Figure 5. Sci V.S. Talk

B. Document-word Co-clustering

We next evaluate the proposed approaches on document-word co-clustering. The data set is generated from 20 Newsgroup, from which a subset of documents are drawn to construct 7 data sets as shown in Table II.

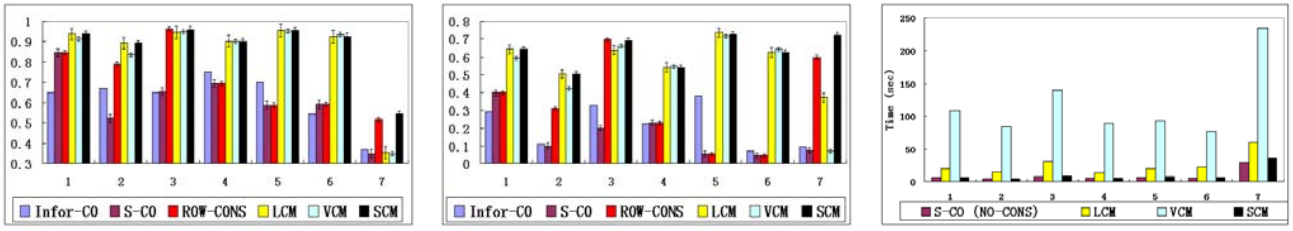
Construct document-word co-clustering constraints For document constraints, we assume that we can observe a small fraction (i.e., 5%) of labeled documents. Those documents with the same class label are then constrained together. Moreover, to construct word constraints, we provide the following methods: (1) we can use the open source software such as the Wordnet to find similar words; (2) we can also make use of the social tagging systems, such as Wikipedia, to explore the word similarity. To do so, we first perform a co-clustering on the well-labeled documents from these tagging systems. Those words assigned in the same cluster are constrained to be together. In our experiment, we use the later method to construct the word constraints from a subset of well-classified documents (5%) randomly drawn from 20 Newsgroup.

The results in Fig. 4 are summarized over 10 runs and in each run the constraints are randomly re-constructed. Fig. 4 shows that the proposed constrained co-clustering methods outperform the comparison models especially in the 6th data set where the cluster purity improves over 60% (as compared to the information based co-clustering and the spectral co-clustering), and the NMI improves by 11 times. Among the three constrained co-clustering methods, SCM finds the optimal co-clusters more efficiently. For example, when VCM costs over 80 seconds on the 2nd data set, SCM achieves a similar clustering quality with only 5 seconds. Note that in this experiment, the bipartite graph generated from the document-word data set is very sparse. In this way, LCM and VCM can effectively model the constraints by



(a) Cluster Purity Comparison (b) Normalized Mutual Information (c) Running Time of the Constrained-based Co-clustering

Figure 3. Graph-Pattern Co-clustering (only 5% of the graphs have row constraints)



(a) Cluster Purity Comparison (b) Normalized Mutual Information (c) Running Time of the Constrained-based Co-clustering

Figure 4. Word-Document co-clustering (only 5% of the documents have row constraints)

adding edges or nodes, and thus the difference among the clustering quality of the three methods becomes smaller. However, for the 7th dataset with 4 different categories, SCM does substantially better than LCM and VCM. This is because when the graph structure becomes more complicated (grows to 4 clusters instead of 2), LCM and VCM are not effective to explore the complicated structure by just adding links and nodes. We also plot Fig. 6 using the 6th data set to study why the co-clustering constraints help improve the results. It shows that without prior knowledge, the co-clustering may reach a local optimum because of the noise and the skewed distribution of the data (Fig. 5(a)). Row constraints and column constraints help improve the situation by using the background knowledge (Fig. 5(b)).

V. Conclusion

For many applications, some prior knowledge exists about the relationship among rows and columns for co-clustering applications. These information helps reduce the effect of noise and can generate meaningful result that is consistent with users' perspective. We propose SCM (Spectral Constraint Modeling) that directly models those prior constraints as a novel trace minimization problem. The key idea is to minimize the inter-co-cluster connections while at the same time give penalty to the co-clustering assignments that violate these constraints. The most important features is that it does not increase complexity of the graphs. Experiments include graph-pattern co-clustering and document-word co-clustering, in which SCM can substantially improve clustering quality. For instance, in the application of chemical compound graph-pattern analysis, SCM improves the normalized mutual information by as much as 1.5 times in the first data set and 5.5 times in the sixth data set. Most importantly, the running time of SVM is comparable to the traditional spectral co-clustering that does not use these constraints.

VI. Acknowledgements

This work is supported in part by NSF through grants IIS 0905215, DBI-0960443, OISE-0968341 and OIA-0963278.

References

[1] Dhillon, I. S.: Co-clustering documents and words using bipartite spectral graph partitioning. *KDD'01*. (2001)

[2] Dhillon, I. S., Mallela, S., and Modha, D.S.: Information Theoretic Coclustering. *KDD'03*. (2003)

[3] Madeira, S. C. and Oliveira, A. L.: Biclustering algorithms for biological data anslysis: A survey. *IEEE Transactions on Computational Biology and Bioinformatics*, *1(1)*, 24–45. (2004)

[4] Pensa, R. G., Robardet C., and Boulicaut, J.: Towards constrained co-clustering in ordered 0/1 data sets. *ISMIS'06*. (2006)

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

[6] Stahl, M. and Rarey, M.(Stahl Data Set): http://www.cheminformatics.org/datasets/

[7] Yan, X. and Han, J.: Gspan: Graph-based substructure pattern mining. *ICDM'02*. (2002)