A Scalable Framework for Discovering Coherent Co-clusters in Noisy Data

Meghana Deodhar, Gunjan Gupta, Joydeep Ghosh DEODHAR,GGUPTA,GHOSH@ECE.UTEXAS.EDU
Department of ECE, University of Texas at Austin, 1 University Station C0803, Austin, TX, USA

Hyuk Cho, Inderjit Dhillon HYUKCHO,INDERJIT@CS.UTEXAS.EDU
Department of CS, University of Texas at Austin, 1 University Station C0500, Austin, TX, USA

Abstract

Clustering problems often involve datasets where only a part of the data is relevant to the problem, e.g., in microarray data analysis only a subset of the genes show cohesive expressions within a subset of the conditions/features. The existence of a large number of non-informative data points and features makes it challenging to hunt for coherent and meaningful clusters from such datasets. Additionally, since clusters could exist in different subspaces of the feature space, a co-clustering algorithm that simultaneously clusters objects and features is often more suitable as compared to one that is restricted to traditional "one-sided" clustering. We propose Robust Overlapping Co-Clustering (ROCC), a scalable and very versatile framework that addresses the problem of efficiently mining dense, arbitrarily positioned, possibly overlapping co-clusters from large, noisy datasets. ROCC has several desirable properties that make it extremely well suited to a number of real life applications.

1. Motivation

When clustering certain real world datasets, it has been observed that only a part of the data forms cohesive clusters. For example, in the case of microarray data, typically only a small subset of the genes cluster well and the rest can be considered non-informative (Gupta & Ghosh, 2006). Problems addressed by eCommerce businesses, such as market basket analysis and fraud detection involve huge, noisy datasets with coherent patterns occurring only in small pockets of the data. Moreover, for such data, coherent clusters could be arbitrarily positioned in subspaces formed by different, possibly overlapping subsets of features, e.g., different subsets of genes may be correlated across different subsets of experiments in microarray data. Additionally, it is possible that some features may not be relevant to any cluster.

Traditional clustering algorithms like k-means or approaches such as feature clustering (Dhillon et al., 2003a) do not allow clusters existing in different subsets of the feature space to be detected easily. Co-clustering simultaneously clusters the data along multiple axes, e.g., in the case of microarray data it simultaneously clusters the genes as well as the experiments (Cheng & Church, 2000) and can hence detect clusters existing in different subspaces of the feature space. In this paper we focus on real life datasets, where co-clusters are arbitrarily positioned in the data matrix, could be overlapping and are obfuscated by the presence of a large number of irrelevant points. Our goal is to discover dense, arbitrarily positioned and overlapping co-clusters in the data, while simultaneously pruning away non-informative objects and features.

2. Related Work

Density based clustering algorithms such as DBSCAN (Ester et al., 1996), OPTICS and Bregman Bubble Clustering (Gupta & Ghosh, 2006) have a motivation similar to our proposed approach and use the notion of local density to cluster only a relevant subset of the data into multiple dense clusters. However, all of these approaches are developed for one-sided clustering only, where the data points are clustered based on their similarity across the entire set of features. In contrast, both co-clustering (biclustering) and subspace clustering approaches locate clusters in

Appearing in *Proceedings of the 26th International Conference on Machine Learning*, Montreal, Canada, 2009.

subspaces of the feature space. The literature in both areas is recent but explosive, so we refer to the surveys and comparative studies in (Madeira & Oliveira, 2004; Parsons et al., 2004; Prelic et al., 2006) as good starting points. As we shall see in Section 3, none of the existing methods provide the full set of capabilities that the proposed method provides.

Co-clustering was first applied to gene expression data by Cheng and Church (2000), who used a greedy search heuristic to generate arbitrarily positioned, overlapping co-clusters, based on a homogeneity constraint. However, their iterative insertion and deletion based algorithm is expensive, since it identifies individual co-clusters sequentially rather than all at once. The algorithm also causes random perturbations to the data while masking discovered biclusters, which reduces the clustering quality. The plaid model approach (Lazzeroni & Owen, 2002) improves upon this by directly modeling overlapping clusters, but still cannot identify multiple co-clusters simultaneously. These algorithms are not very general as they assume additive Gaussian noise models. Neither can they effectively handle missing data.

In addition to the greedy, iterative algorithms discussed above, deterministic algorithms such as BiMax (Prelic et al., 2006) and OPSM (Ben-Dor et al., 2002) have also been proposed. The BiMax approach is based on a simple, binary data model, which results in a number of co-clusters that is exponential in the number of genes and experiments, making it impractical in case of large datasets. The order preserving sub matrix algorithm (OPSM) looks for submatrices in which the expression levels of all the genes induce the same linear ordering of the experiments. This algorithm although very accurate, is designed to identify only a single co-cluster. A recent extension to OPSM (Zhang et al., 2008) finds multiple, overlapping co-clusters in noisy datasets, but is very expensive in the number of features.

Bregman Co-Clustering (BCC), proposed by Banerjee et al. (2007), is a highly efficient, generalized framework for partitional co-clustering (Madeira & Oliveira, 2004) that works with any distance measure that is a Bregman divergence, or equivalently any noise distribution from the regular exponential family. The BCC framework is however restricted to grid-based, partitional co-clustering and assigns every point in the data matrix to exactly one co-cluster, i.e., the co-clustering is exhaustive and exclusive.

Parsons et al. (2004) present a survey of subspace clustering algorithms, which includes bottom-up grid based methods like CLIQUE and iterative top-down algorithms like PROCLUS. However, most of them are computationally intensive, need extensive tuning to get meaningful results and identify uniform clusters with very similar values rather than clusters with coherent trends or patterns. The pCluster model (Wang et al., 2002) and the more recent reg-cluster model (Xu et al., 2006) generalize subspace clustering and aim to identify arbitrary scaling and shifting co-regulations patterns. However, unlike our proposed approach, these pattern-based, heuristic approaches do not use a principled cost function and do not scale well due to high complexity in the number of features.

3. Our Contributions

We propose Robust Overlapping Co-clustering (ROCC), a novel approach for discovering dense, arbitrarily positioned co-clusters in large, possibly high-dimensional datasets. Our approach is robust in the presence of noisy and irrelevant objects as well as features, which our algorithm automatically detects and prunes during the clustering process. ROCC is based on a systematically developed objective function, which is minimized by an iterative procedure that provably converges to a locally optimal solution. ROCC is also robust to the noise model of the data and can be tailored to use the most suitable distance measure for the data, selected from a large class of distance measures known as Bregman divergences.

The final objective of ROCC is achieved in two steps. In the first step, the Bregman co-clustering algorithm is adapted to automatically prune away non-informative data points and perform feature selection by eliminating non-discriminative features and hence cluster only the relevant part of the dataset. This step finds co-clusters arranged in a grid structure, but only a predetermined number of rows and columns are assigned to the co-clusters. Note however that this result cannot be achieved by simply removing some rows/columns from the BCC result. An agglomeration step then appropriately merges similar co-clusters to discover dense, arbitrarily positioned, overlapping co-clusters. Figure 1 contrasts the nature of the co-clusters identified by ROCC with those found by BCC and illustrates the way in which they are conceptually derived from the partitional model of BCC.

The ROCC framework has the following key features that distinguish it from existing co-clustering algorithms:

1. The ability to mine the most coherent co-clusters from large and noisy datasets.

2. Detection of arbitrarily positioned and possibly

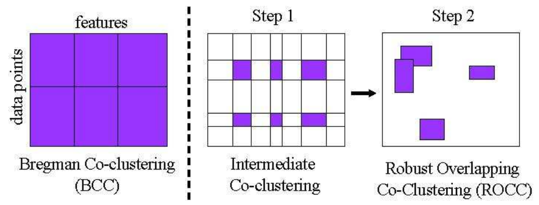
features
data points
Bregman Co-clustering (BCC)
Step 1
Intermediate Co-clustering
Step 2
Robust Overlapping Co-Clustering (ROCC)

Figure 1. Nature of clusters identified by BCC and ROCC. Shaded areas represent clustered elements, rearranged according to cluster labels, while non-shaded areas denote discarded values.

overlapping co-clusters in a principled manner by iteratively minimizing a suitable cost function.

3. Generalization to all Bregman divergences, including squared Euclidean distance, commonly used for clustering microarray data and I-divergence, commonly used for text data clustering (Dhillon et al., 2003b).

4. The ability to naturally deal with missing data values, without introducing random perturbations or bias in the data.

5. Efficient detection of all co-clusters simultaneously rather than sequentially, enabling scalability to large and high-dimensional datasets.

As far as we know, no existing co-clustering algorithm (Zhang et al., 2008; Xu et al., 2006; Banerjee et al., 2007; Cheng & Church, 2000) has all of the above properties. Our contribution is significant, since as described in Section 1 there exist several applications where all these properties are necessary for discovering meaningful patterns.

4. Problem Definition

We begin with the formulation of the first step of the ROCC algorithm. Let m be the total number of rows (data points) and n the total number of columns (features). The data can be represented as an $m \times n$ matrix Z of data points and features. Let s_r and s_c be the specified number of rows and columns, respectively, to be retained after pruning. If the exact values are not known, it is sufficient to set s_r and s_c conservatively to large values since the algorithm (Section 5.3) does a second round of pruning as needed. Our aim is to simultaneously cluster s_r rows and s_c columns of Z into a grid of k row clusters and l column clusters. The co-clusters will hence be comprised of $s_r \times s_c$ entries selected from the $m \times n$ entries of Z. Let $\mathcal{K}$ and $\mathcal{L}$ denote the sets consisting of the s_r clustered rows and the s_c clustered columns respectively. Let ρ be a mapping from the s_r rows $\in \mathcal{K}$ to the k row clusters and γ be a mapping from the s_c columns $\in \mathcal{L}$ to the l column clusters. Let squared Euclidean distance be the selected distance measure [1]. We want to find a co-clustering defined by (ρ, γ) and sets $\mathcal{K}$ and $\mathcal{L}$ for the specified s_r and s_c that minimize the following objective function

$$\sum_{g=1}^{k}\sum_{h=1}^{l}\sum_{u\in\mathcal{K}:\rho(u)=g}\;\sum_{v\in\mathcal{L}:\gamma(v)=h} w_{uv}(z_{uv}-\hat{z}_{uv})^2\ , \qquad (1)$$

where z_{uv} is the original value in row u, column v of the matrix, assigned to row cluster g and column cluster h and $\hat{z}_{uv}$ is the value approximated within co-cluster g-h. w_{uv} is the non-negative weight associated with matrix entry z_{uv}, which allows the algorithm to deal with missing values and data uncertainties. For example, the weights for known values can be set to 1 and missing values can be effectively ignored by setting their weights to 0. The objective function is hence the element-wise squared error between the original and the approximated value, summed only over the clustered elements ($s_r \times s_c$) of the matrix Z. The value $\hat{z}_{uv}$ can be approximated in several ways, depending on the type of summary statistics that each co-cluster preserves. Banerjee et al. (2007) identify six possible sets of summary statistics, of increasing complexity, that one might be interested in preserving in the reconstructed matrix $\hat{Z}$, which lead to six different co-clustering schemes. Two of these approximation schemes for $\hat{z}_{uv}$ are described in Section 5.3.

In the next step of ROCC, the goal is to agglomerate similar co-clusters to recover the arbitrarily positioned co-clusters. In order to agglomerate co-clusters, we first define a distance measure between two candidate co-clusters ($cc1$ and $cc2$) as follows. Let cc denote the co-cluster formed by the union of the rows and columns in $cc1$ and $cc2$. The matrix entries $\hat{z}_{uv}$ in cc are approximated using the selected approximation scheme. The average element-wise error e for cc is computed as $e = \frac{1}{N}\sum_{z_{uv}\in cc}(z_{uv}-\hat{z}_{uv})^2$, where N is the number of elements in cc. The error e is defined to be the distance between $cc1$ and $cc2$.

5. ROCC Algorithm

5.1. Solving Step 1 of the ROCC Problem

A co-clustering (ρ, γ), that minimizes the objective function (1), can be obtained by an iterative algorithm.

[1] A more general description, which allows any Bregman divergence as the loss function, is given in Section 5.3.

The objective function can be expressed as a sum of row or column errors, computed over the s_r rows and s_c columns assigned to co-clusters. If row u is assigned to row cluster g, the row error is the error summed over the appropriate s_c elements in the row, i.e., if $\rho(u) = g$, then $E_u(g) = \sum_{h=1}^{l} \sum_{v \in \mathcal{L}:\gamma(v)=h} w_{uv}(z_{uv} - \hat{z}_{uv}(g))^2$. For a fixed γ, the best choice of the row cluster assignment for row u is the g that minimizes this error, i.e., $\rho^{new}(u) = \arg{}_g\min\, E_u(g)$. After computing the best row cluster assignment for all the m rows, the top s_r rows with minimum error are selected to participate in the current row clusters. A similar approach is used to assign columns to column clusters. Note that the rows/columns that are not included in the current s_r/s_c rows/columns assigned to co-clusters are still retained since they could be included in the co-clusters in future iterations.

Given the current row and column cluster assignments (ρ, γ), the values $\hat{z}_{uv}$ within each co-cluster have to be updated by recomputing the required co-cluster statistics based on the approximation scheme. This problem is identical to the Minimum Bregman Information (MBI) problem presented in (Banerjee et al., 2007) for updating the matrix reconstruction $\hat{Z}$. Solving the MBI problem for this update is guaranteed to decrease the objective function.

This iterative procedure is described in Figure 2. Step 1(i) decreases the objective function due to the property of the MBI solution, while Steps 1(ii) and 1(iii) directly decrease the objective function. The objective function hence decreases at every iteration. Since this function is bounded from below by zero, the algorithm is guaranteed to converge to a locally optimal solution.

5.2. Solving Step 2 of the ROCC Problem

We now provide a heuristic to hierarchically agglomerate similar co-clusters. The detailed steps are:

(i) Pruning co-clusters. Since the desired number of co-clusters is expected to be significantly smaller than the number of co-clusters at this stage of the algorithm, co-clusters with the largest error values can be filtered out in this step. Filtering also reduces the computation effort required by the following merging step. If one has no idea of the final number of co-clusters, a simple and efficient filtering heuristic is to select the error cut-off value as the one at which the sorted co-cluster errors show the largest increase between consecutive values. The co-clusters with errors greater than the cut-off are filtered out. Alternatively, if the final number of co-clusters to be found is prespecified, it can be used to prune away an appropriate number of co-clusters with the largest errors.

(ii) Merging similar co-clusters. This step involves hierarchical, pairwise agglomeration of the co-clusters left at the end of the pruning step (Step 2(i)) to recover the true co-clusters. Each agglomeration identifies the "closest" pair of co-clusters that can be well represented by a single co-cluster model and are thus probably part of the same original co-cluster, and merges them to form a new co-cluster [2]. "Closest" here is in terms of the smallest value of distance as defined in Section 4. The rows and columns of the new co-cluster consist of the union of the rows and columns of the two merged co-clusters. Merging co-clusters in this manner allows co-clusters to share rows and columns and hence allows partial overlap between co-clusters. If the number of co-clusters to be identified is prespecified, one can stop merging when this number is reached. If not, merging is continued all the way until only a single co-cluster (or a reasonably small number of co-clusters) is left. The increase in the distance between successively merged co-clusters is then computed and the set of co-clusters just before the largest increase is selected as the final solution.

5.3. Overall ROCC Meta-Algorithm

In this section we put together the procedures described in Sections 5.1 and 5.2 and present the complete ROCC algorithm. The key idea is to over-partition the data into small co-clusters arranged in a grid structure and then agglomerate similar, partitioned co-clusters to recover the desired co-clusters. The iterative procedure (Section 5.1) is run with large enough values for the number of row and column clusters (k and l). Similarly, the s_r and s_c input parameters are set to sufficiently large values. Since the pruning step (Step 2(i) in Section 5.2) takes care of discarding less coherent co-clusters, setting $s_r \geq s_r^{\text{true}}$ and $s_c \geq s_c^{\text{true}}$ is sufficient. The resulting $k \times l$ clusters are then merged as in hierarchical agglomerative clustering until a suitable stopping criterion is reached. The pseudo-code for the complete algorithm is illustrated in Figure 2.

Approximation Schemes. The ROCC algorithm can use each of the six schemes (co-clustering bases) listed by Banerjee et al. (2007) for approximating the matrix entries $\hat{z}_{uv}$. For concreteness, we illustrate two specific approximation schemes with squared Euclidean distance, which give rise to block co-clusters and pattern-based co-clusters respectively [3]. The

[2] A variant of this algorithm can be derived by adopting Ward's method (Ward, 1963) to agglomerate co-clusters. Empirically we found little difference between the two approaches.

[3] These co-cluster definitions correspond to basis 2 and

meta-algorithm in Figure 2 uses C to refer to the selected co-clustering basis.

Block co-clusters. Let the co-cluster row and column indices be denoted by sets U and V respectively. In this case, a matrix entry is approximated as $\hat{z}_{uv} = z_{UV}$, where $z_{UV} = \frac{1}{|U||V|}\sum_{u \in U, v \in V} z_{uv}$ is the mean of all the entries in the co-cluster.

Pattern-based co-clusters. z_{uv} is approximated as $\hat{z}_{uv} = z_{uV} + z_{Uv} - z_{UV}$, where $z_{uV} = \frac{1}{|V|}\sum_{v \in V} z_{uv}$ is the mean of the entries in row u whose column indices are in V and $z_{Uv} = \frac{1}{|U|}\sum_{u \in U} z_{uv}$ is the mean of the entries in column v whose row indices are in U. This approximation can identify co-clusters that show a coherent trend or pattern in the data values, making it suitable for clustering gene expression data (Cho & Dhillon, 2008).

Distance Measures. In Section 4 we developed the objective function (1) assuming squared Euclidean distance as the distance measure. The objective function and the iterative procedure to minimize it can be generalized to all Bregman divergences (Banerjee et al., 2007). The selected Bregman divergence is denoted by d_ϕ in Figure 2.

```
Algorithm: ROCC
Input: $Z_{m \times n}$, $s_r, s_c, k, l$, basis $C$, $d_\phi$
Output: Set of co-clusters

Step 1
Begin with a random co-clustering $(\rho, \gamma)$
Repeat
    Step (i): Update co-cluster models, $\forall [g]_1^k, [h]_1^l$,
    Update statistics for co-cluster $(g, h)$ based on
    basis $C$ to compute new $\hat{z}$ values

    Step (ii): Update $\rho$
    (iia). $\forall [u]_1^m$,
    $\rho(u) = \arg{}_g\min \sum_{h=1}^{l} \sum_{v \in \mathcal{L}: \gamma(v)=h} w_{uv} d_\phi(z_{uv}, \hat{z}_{uv}(g))$
    (iib). $\mathcal{K}$ = the set of $s_r$ rows with least error
    from among the $m$ rows

    Step (iii): Update $\gamma$
    (iiia). $\forall [v]_1^n$,
    $\gamma(v) = \arg{}_h\min \sum_{g=1}^{k} \sum_{u \in \mathcal{K}: \rho(u)=g} w_{uv} d_\phi(z_{uv}, \hat{z}_{uv}(h))$
    (iiib). $\mathcal{L}$ = the set of $s_c$ columns with least error
    from among the $n$ columns

until convergence
Step 2: Post-process (see text for details)
(i) Prune co-clusters with large errors.
(ii) Merge similar co-clusters until stopping criterion is reached.
return identified co-clusters.
```

Figure 2. Pseudo-code for ROCC Meta-Algorithm

5.4. ROCC with Pressurization

The iterative minimization procedure in Step 1 of the ROCC algorithm begins with random initialization for ρ and γ, which could lead to poor local minima. A better local minimum can be achieved by applying an extension of the pressurization technique used by BBC (Gupta & Ghosh, 2006). Our strategy is to begin by clustering all the data and iteratively shave off data points and features till s_r rows and s_c columns are left. Let ${s_r^{press}}_{(j)}$ and ${s_c^{press}}_{(j)}$ denote the number of data points and features to be clustered using the Step 1 procedure (Figure 2) in the j^{th} iteration of pressurization. ${s_r^{press}}_{(1)}$ and ${s_c^{press}}_{(1)}$ are initialized to m and n respectively, after which these parameters are decayed exponentially till ${s_r^{press}}_{(j)} = s_r$ and ${s_c^{press}}_{(j)} = s_c$. The rate of decay is controlled by parameters β_{row} and β_{col}, which lie between 0 and 1. At iteration j, ${s_r^{press}}_{(j)} = s_r + \lfloor (m - s_r) * \beta_{row}^{j-1} \rfloor$ and ${s_c^{press}}_{(j)} = s_c + \lfloor (m - s_c) * \beta_{col}^{j-1} \rfloor$. The intuition is that by beginning with all the data being clustered and then slowly reducing the fraction of data clustered, co-clusters can move around considerably from their initial positions to enable the discovery of small, coherent patterns.

6. Experimental Results

6.1. Finding Co-clusters in Microarray Data

We now evaluate the performance of ROCC on two yeast microarray datasets, the Lee dataset (Lee et al., 2004) and the Gasch dataset (Gasch et al., 2000). The Lee dataset consists of gene expression values of 5612 yeast genes across 591 experiments and can be obtained from the Stanford Microarray Database (`http://genome-www5.stanford.edu/`). The Gasch dataset consists of the expression values of 6151 yeast genes under 173 environmental stress conditions and is available at `http://genome-www.stanford.edu/yeast_stress/`. Since the ground truth for both datasets is available only in the form of pairwise linkages between the genes that are known to be functionally related, we compare the quality of the co-clusters identified by different co-clustering algorithms by computing the overlap lift (Gupta & Ghosh, 2006) for the genes in each co-cluster. Overlap lift measures how many times more correct links are predicted as compared to random chance and is related to a normalized version of the proportion of disconnected genes measure used by (Prelic et al., 2006). On these datasets, the aim is to find the most coherent and biologically useful 150 to 200 co-clusters. We run ROCC (with pressurization) on the Lee dataset with the input parameters set to $s_r = 2000$, $s_c = 400$, $k = 50$ and $l = 10$

basis 6 defined by the BCC framework (Banerjee et al., 2007) respectively.

and on the Gasch dataset with $s_r = 500$, $s_c = 120$, $k = 80$, $l = 15$. Based on the final number of clusters to be identified, Step 2 of ROCC prunes all but the best 200 co-clusters and then continues merging until 150 co-clusters are left. The set of co-clusters just before the largest increase in merge distance is returned as the solution.

Figure 3 compares the performance of ROCC with prominent co-clustering algorithms, i.e., Cheng and Church's Biclustering algorithm, the OPSM algorithm (Ben-Dor et al., 2002), the BiMax algorithm (Prelic et al., 2006), and the BCC algorithm on the Lee and Gasch microarray datasets. Through extensive experimentation, Prelic et al. (2006) show that the OPSM and the BiMax algorithms outperform other well known co-clustering algorithms like Samba (Tanay et al., 2002), ISA (Bergmann et al., 2003) and xMotif (Murali & Kasif, 2003) on real microarray data. The BiMax and OPSM results were generated using the BicAT software(`http://www.tik.ee.ethz.ch/sop/bicat/`) (Barkow et al., 2006). Since it would be infeasible to evaluate the exponential number of co-clusters identified by BiMax, we selected the first 200 co-clusters for comparison. Though OPSM is designed to return only the best co-cluster, it is extended in BicAT to return up to 100 largest co-clusters among those that achieve the optimal score. The value of the l parameter for OPSM was set to 10. The Biclustering algorithm [4] is run with the number of clusters equal to 200. The value of the parameter α is set to the average H-score (Cheng & Church, 2000) of the co-clusters in the ROCC solution with the highest overlap lift over varying s_r and s_c values, i.e., $\alpha = 0.032$ for Lee and $\alpha = 0.017$ for Gasch [5]. Since BCC clusters all the data, pruning is carried out by a post-processing step. This step sorts the rows and columns by their distance to the corresponding cluster representatives and selects the s_r rows and s_c columns with smallest errors. In the Lee and Gasch datasets respectively, around 15% and 3% of the matrix entries are missing. As described in Section 5, ROCC and BCC can ignore missing entries by appropriately setting the weight matrix. The missing entries in the data matrix input to the other algorithms are replaced by random values in the same range as the known expression values. Both ROCC and BCC use squared Euclidean distance and find pattern-based co-clusters. BCC uses the same s_r and s_c values as ROCC. The ROCC, BCC and Biclustering results are averaged over 10 trials, while OPSM and BiMax are deterministic.

Figure 3 shows that on both datasets, ROCC does much better than the other co-clustering approaches in terms of the overlap lift of the gene clusters. The figure also displays above each bar, the percentage of the data matrix entries clustered by the corresponding algorithm. On the Lee dataset, it is interesting that although ROCC clusters a much larger fraction of the data matrix entries than Biclustering, OPSM and BiMax, the co-clusters are of superior quality. The Gasch dataset is more noisy than Lee, which explains why a larger fraction of the dataset has to be pruned as compared to Lee to get meaningful clusters. Lesion studies confirmed that both step 1 and step 2 of the ROCC algorithm contribute to the improvement in performance, step 1 being more important. We empirically compared only step 1 with other approaches (BCC, BBC, k-means) on the Lee and Gasch datasets, for different fractions of retained data (for other algorithms the least fitting data was discarded in a post-processing step). ROCC with only step 1 was significantly better than all others for 10% or more of the data discarded. A more detailed description is presented in (Deodhar et al., 2008).

Most of the gene clusters identified by ROCC on the Lee dataset were biologically significant, with very low p-values. Table 1 summarizes some of the identified high purity gene clusters. The coverage (x/y) indicates that x out of the y known members of a category were found. In contrast, the 10 best gene clusters identified by Biclustering had an *average p-value* of 5.50e-04.

Table 1. Examples of biologically significant clusters found by ROCC on the Lee dataset.

# genes	Category(Coverage)	p-value
20	tRNA ligase (8/36)	6.63e-14
63	ER membrane (14/84)	3.886e-14
20	PF00270-DEAD (12/51)	<1e-14
12	Glycolysis (8/16)	<1e-14
24	PF00660-SRP1-TIP1 (22/30)	<1e-14

6.2. Simultaneous Feature Selection and Clustering

We now illustrate an interesting application of the ROCC algorithm to perform feature selection along one axis, while simultaneously clustering along the other. ROCC interleaves feature selection with clustering and iteratively improves both, which is intuitively better than independently performing feature selection *a priori* and then clustering using the identi-

[4] We used the implementation provided by Cheng and Church.

[5] We found the biclustering results to not be very sensitive to the choice of α (range of α values from 0.005 to 0.04 were tried).

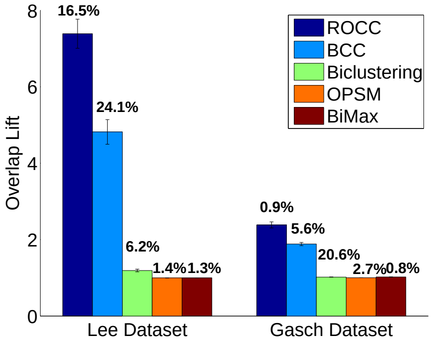

Figure 3. Comparison of ROCC with other co-clustering algorithms on the Lee and Gasch datasets. The number above each bar indicates the percentage of the data matrix entries clustered by each algorithm.

fied features (Law et al., 2004). Additionally, ROCC also clusters related features, achieving simultaneous dimensionality reduction.

We consider an exemplary application of ROCC in the above context to a lung cancer microarray dataset (Gordon et al., 2002) with 12533 genes and 181 human tissue samples. The samples belong to two lung cancer classes, malignant pleural mesothelioma (31 samples) and adenocarcinoma (150 samples). In this application, the aim is to cluster the samples, to recover the two existing sample groups in an unsupervised manner, using the expression values of the genes as features. Many of the genes are known to be non-informative and have noisy expression values, which makes feature selection an important issue. We use a version of the dataset that is pre-processed based on domain knowledge, where genes that do not show substantial variation in expression values across the samples are removed as described in (Cho & Dhillon, 2008), resulting in a set of 2401 genes. Even though the pre-processing step results in removing several non-discriminative genes, we apply ROCC to test if any more genes can be identified, that on pruning will improve sample cluster accuracy further. For this application, ROCC (with pressurization) is set up to cluster all the samples and prune along the “gene” axis. Note that the agglomeration procedure (Step 2) is not required for this application.

Sample clustering solutions are evaluated by computing the accuracy of the cluster labels with respect to the true class labels as defined in (Cho & Dhillon, 2008). Figure 4 displays the sample cluster accuracy of ROCC at different fractions of genes clustered. For comparison, the sample cluster accuracy values of BCC, which uses all the genes to obtain a co-clustering of genes and samples, and k-means, which uses all the genes as features to cluster the samples are also plotted as straight lines in the same figure. These experiments are performed on the column standardized dataset, where every column has zero mean and unit variance. BCC and ROCC use squared Euclidean distance and find pattern-based co-clusters with $k = 20$,$l = 2$. The results are averaged over 20 runs. One can see that ROCC gives almost perfect clustering, even with only 10% of the genes selected, significantly better than BCC and k-means.

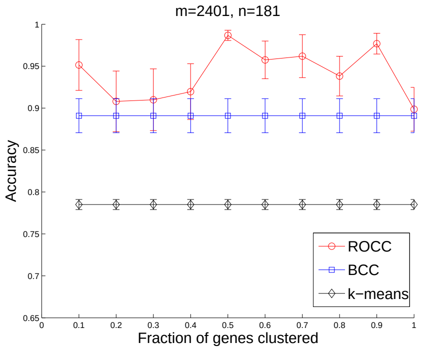

Figure 4. Lung Cancer data: sample clustering accuracy

7. Concluding Remarks

In this paper, we have presented Robust Overlapping Co-clustering as a comprehensive framework capable of dealing with several challenges in clustering real life datasets. ROCC is robust to the presence of irrelevant data points and features, and discovers coherent co-clusters very accurately as illustrated in Section 6. Moreover, though ROCC requires several input parameters to be supplied, i.e., s_r, s_c, k and l, it is relatively very robust to the choice of these parameters because of the post-processing steps as detailed in Section 5.3. While in this paper we focused on clustering microarray data, it would be worthwhile to investigate the applicability of suitable instances of the ROCC framework to clustering problems in different domains like text mining and market basket analysis.

Acknowledgments. This research was supported by NSF grants IIS 0325116, IIS 0307792, IIS 0713142 and CCF 0431257.

References

Banerjee, A., Dhillon, I., Ghosh, J., Merugu, S., & Modha, D. (2007). A generalized maximum entropy approach to Bregman co-clustering and matrix approximation. *Jl. Machine Learning Research*, *8*, 1919–1986.

Barkow, S., Bleuler, S., Prelic, A., Zimmermann, P.,

& Zitzler, E. (2006). Bicat: a biclustering analysis toolbox. *Bioinformatics*, *22(10)*, 1282–1283.

Ben-Dor, A., Chor, B., Karp, R., & Yakhini, Z. (2002). Discovering local structure in gene expression data: the order-preserving submatrix problem. *Proc. Research in Comp. Mol. Bio. '02* (pp. 49–57).

Bergmann, S., Ihmels, J., & Barkai, N. (2003). Iterative signature algorithm for the analysis of large-scale gene expression data. *Phys. Rev. E. Stat. Nonlin. Soft Matter Phys.*, *67*.

Cheng, Y., & Church, G. M. (2000). Biclustering of expression data. *Proc. Intell. Syst. Mol. Bio. '00* (pp. 93–103).

Cho, H., & Dhillon, I. (2008). Co-clustering of human cancer microarrays using minimum sum-squared residue co-clustering. *IEEE/ACM Trans. on Comp. Bio. and Bioinfo.*, *5*, 385–400.

Deodhar, M., Cho, H., Gupta, G., Ghosh, J., & Dhillon, I. (2008). Robust overlapping co-clustering. *Dept. of ECE, Univ. of Texas at Austin, IDEAL-TR09*, Downloadable from `http://www.lans.ece.utexas.edu/papers/techreports/deodhar08ROCC.pdf`.

Dhillon, I., Mallela, S., & Kumar, R. (2003a). A divisive information-theoretic feature clustering algorithm for text classification. *Jl. Machine Learning Research*, *3*, 1265–1287.

Dhillon, I., Mallela, S., & Modha, D. (2003b). Information-theoretic co-clustering. *Proc. Know. Disc. and Data Mining '03* (pp. 89–98).

Ester, M., Kriegel, H., Sander, J., & Xu, X. (1996). A density-based algorithm for discovering clusters in large spatial databases with noise. *Proc. Know. Disc. and Data Mining '96*.

Gasch, A., Spellman, P., Kao, C., Carmel-Harel, et al.(2000). Genomic expression program in the response of yeast cells to environmental changes. *Molecular Cell Biology*, *11*, 4241–4257.

Gordon, G. J., Jensen, R. V., Hsiao, L., Gullans, S. R., et al. (2002). Translation of microarray data into clinically relevant cancer diagnostic tests using gene expression ratios in lung cancer and mesothelioma. *Cancer Research*, *62*, 4963–4967.

Gupta, G., & Ghosh, J. (2006). Bregman bubble clustering: A robust, scalable framework for locating multiple, dense regions in data. *Proc. Int. Conf. on Data Mining '06* (pp. 232–243).

Lazzeroni, L., & Owen, A. B. (2002). Plaid models for gene expression data. *Statistica Sinica*, *12*, 61–86.

Law, M., Figueiredo, M., & A.K.Jain (2004). Simultaneous feature selection and clustering using a mixture model. *IEEE Trans. PAMI*, *26*, 1154–1166.

Lee, I., Date, S., Adai, A., & Marcotte, E. (2004). A probabilistic functional network of yeast genes. *Science*, *306*, 1555–1558.

Madeira, S. C., & Oliveira, A. L. (2004). Biclustering algorithms for biological data analysis: A survey. *IEEE/ACM Trans. on Comp. Bio. and Bioinfo.*, *1*, 24–45.

Murali, T., & Kasif, S. (2003). Extracting conserved gene expression motifs from gene expression data. *Pacific Symposium on Biocomp.*, *8*, 77–88.

Parsons, L., Haque, E., & Liu, H. (2004). Subspace clustering for high dimensional data: a review. *SIGKDD Explor. Newsl.*, *6*, 90–105.

Prelic, A., Bleuler, S., Zimmermann, P., Wille, A., & et. al (2006). A systematic comparison and evaluation of biclustering methods for gene expression data. *Bioinformatics*, *22(9)*, 1122–1129.

Tanay, A., Sharan, R., & Shamir, R. (2002). Discovering statistically significant biclusters in gene expression data. *Bioinformatics*, *18*, 136–144.

Wang, H., Wang, W., Yang, J., & Yu, P. (2002). Clustering by pattern similarity in large data sets. *Proc. Int. Conf. on Mgmt. of Data '02* (pp. 394–405).

Ward, J. (1963). Hierarchical grouping to optimize an objective function. *Jl. of American Stat. Assoc.*, *58*, 236 – 244.

Xu, X., Lu, Y., Tung, A., & Wang, W. (2006). Mining shifting-and-scaling co-regulation patterns on gene expression profiles. *Proc. Int. Conf. on Data Engg. '06* (p. 89).

Yoon, S., Nardini, C., Benini, L., & Micheli, G. D. (2005). Discovering coherent biclusters from gene expression data using zero-suppressed binary decision diagrams. *IEEE/ACM Trans. on Comp. Bio. and Bioinfo.*, *2*, 339–354.

Zhang, M., Wang, W., & Liu, J. (2008). Mining approximate order preserving clusters in the presence of noise. *Proc. Int. Conf. on Data Engg. '08* (pp. 160–168).