

Algorithms for Obtaining Overlapping Co-clusters (CS8980) Machine Learning Project Mid-term Progress Report

Rohit Gupta and Varun Chandola
{rohit,chandola}@cs.umn.edu

March 24, 2006

1 Frequent Itemsets Based Approach

We have developed a frequent itemsets based algorithm to obtain overlapping co-clusters in gene expression data. We consider the experiments as transactions and the genes as items. Following is a brief sketch of our algorithm

1. Pre-processing

- (a) Normalize the expression data. The popular approach is to normalize along the experiments to bring all experiment profiles to the same scale and then normalize along the genes.
 - (b) Binarize the data to make it suitable for frequent itemset generation. We have implemented following different schemes
 - Taking absolute values only, set all entries greater than a preset threshold, λ to 1 and the rest to 0.
 - Taking absolute values only, set all entries which lie above a preset percentile to 1 and the rest to 0.
 - Split each gene, g into three components g^- , g^0 and g^+ signifying the up and down regulation of the gene's expression. Thus the number of items will become three times the original number of items.
 - (c) Convert the gene expression matrix into a transaction format such that for each experiment we record the index of the gene which had a value 1 after binarization.
2. Run closed frequent itemset algorithm [4] to obtain frequent closed itemsets with a specified support threshold σ .

3. Post-processing

- (a) Prune frequent itemsets which have number of items (genes) less than a user specified threshold α .
- (b) For each remaining itemset, scan the transaction data to record all the transactions (experiments) in which this itemset occurs. The combination of these transactions (experiments) and the itemset (genes) will form the desired co-cluster.

The above algorithm makes use of three user-defined parameters - λ , σ and α . The choice of the parameters will be governed by the data set, desired resolution of the gene expression matrix decomposition and using domain knowledge. The choice of this parameter set $\Theta = (\lambda, \sigma, \alpha)$ determines the quality and the number of co-clusters obtained. For example choosing a high value for λ would result in co-clusters

comprising only the highly expressed (up or down) genes. This would also result in less number of co-clusters obtained. Similarly different values of σ and α would yield co-clusters at different resolutions. For example, high σ and high α would give only few but large sized co-clusters.

Closed frequent itemsets and maximal frequent itemsets are two choices for obtaining co-clusters. Note that generating all possible frequent itemsets would result in repetitive co-clusters. Consider the simple data shown in Figure 1 which contains two overlapping dense blocks. The closed frequent itemsets in this data

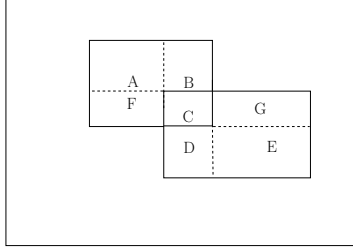


Figure 1: A data matrix with overlapping dense blocks

set will be $\{A+B+C+F\}$, $\{B+C+D\}$ and $\{C+D+E+G\}$. The maximal frequent itemsets in this data will be $\{F+C+G\}$. This assumes that $\{F+C+G\}$ is frequent. Hence we observe that maximal frequent itemsets will miss some desired blocks in the data. Thus we choose to generate closed frequent itemsets.

1.1 Plan of Action

1. So far we are limiting the frequent itemsets to be fully dense (all values should be 1). We can relax this constraint by using *Error Tolerant Frequent Closed Itemsets* [5].
2. Apply the proposed algorithm on gene expression data of *Yeast* and evaluate the results.

2 A Bregman Overlapping Co-clustering Algorithm

In this section we propose a meta algorithm exactly on the lines of the meta co-clustering algorithm which is presented by Banerjee et al [1]. The difference from the original algorithm is that our algorithm will allow overlapping among the co-clusters. In the original paper, the authors initially assume k row clusters and l column clusters. They then obtain co-clusters based on different combinations of the row and column clusters. Thus the maximum number of co-clusters that can be obtained will be $k \times l$. The formulation is such that every row is mapped to only one row cluster and every column is mapped to only one column cluster, thereby resulting in a *hard* co-clustering. We relax this constraint by allowing each row and column to be assigned to multiple row and column clusters respectively. Hence we can obtain overlapping co-clusters.

The original *hard* co-clustering algorithm characterizes the goodness of the co-clustering in terms of error between the original data matrix and its approximation which is reconstructed for a given co-clustering, while preserving certain statistics.

The input matrix is represented by Z which has m rows and n columns. Z can be thought of as a random variable which is a deterministic function of two underlying random variables U and V . U will take values in $\{1 \dots m\}$ such that each u represents a row in Z . Similarly, V will take values in $\{1 \dots n\}$ such that each v represents a column in Z ¹. The co-clustering is represented by two mappings, ρ and γ , such that $\rho(U) = \hat{U}$ and $\gamma(V) = \hat{V}$. \hat{U} and \hat{V} are actually matrices of size $m \times k$ and $n \times l$ respectively. Each column of \hat{U} is denoted by \hat{u} which corresponds to one of the k row clusters. Each column of \hat{V} is denoted by \hat{v} which corresponds to one of the l column clusters. \hat{U}_{ij} denotes the probability of i^{th} row of data matrix Z

¹In the original paper [1] they also have a joint probability measure of the pair (U,V) , denoted by ν . In our formulation we assume that ν is a uniform distribution and hence will not figure in rest of our discussion.

belonging to j^{th} row cluster. Similarly, \hat{V}_{ij} denotes the probability of i^{th} column belonging to j^{th} column cluster. It is worth noting that for *hard* co-clustering formulation, each row in \hat{U} and \hat{V} will contain only one non-zero entry. Effectively, every co-cluster will contain every row and column of Z with certain probability. Finally a preset threshold maybe used on \hat{U} and \hat{V} matrices to exclude elements which have low probability of occurrence in a particular co-cluster.

We then define a set of statistics, referred to as a *pertinent constraint set*, C in [1], using U, V, \hat{U} and \hat{V} . Following is a brief sketch of our proposed approach which builds on the original meta algorithm [1] –

1. Initialize the co-clustering matrices \hat{U} and \hat{V} . One-way clustering algorithms can be applied in both directions independently to initialize these matrices.
2. Using U, V, \hat{U} and \hat{V} , determine matrix approximation of data matrix Z , denoted by \hat{Z}
3. Till convergence
 - (a) For every $u_i \in U$
 - i. For every row cluster $\hat{u}_j \in \hat{U}$
 - A. Update $\hat{U}_{ij} = \text{Proximity between } Z(V|u_i) \text{ and } \hat{Z}(V|\hat{u}_j)$
 - B. \hat{V} will be held constant in this step
 - (b) Recompute \hat{Z} using the current values of U, V, \hat{U} and \hat{V}
 - (c) For every $v_i \in V$
 - i. For every column cluster $\hat{v}_j \in \hat{V}$
 - A. Update $\hat{V}_{ij} = \text{Proximity between } Z(U|v_i) \text{ and } \hat{Z}(U|\hat{v}_j)$
 - B. \hat{U} will be held constant in this step
4. Recompute \hat{Z} using the current values of U, V, \hat{U} and \hat{V}

In the above algorithm \hat{Z} is computed using \hat{U}, \hat{V} and C . C denotes a set of summary statistics based on combinations of (U, V, \hat{U}, \hat{V}) as described in [1]. The proximity between the original data matrix and its approximation based on current overlapping co-clustering is measured using Bregman Divergence between the two matrices. The choice of the Bregman Divergence to use will be governed by the application domain. Dhillon et al use KL-divergence on document data set [3] and squared euclidean distance for gene expression data [2] for hard co-clustering. It should be noted that the Bregman Divergence directly does not represent the elements of \hat{U} and \hat{V} , but will be used to assign the probabilities of u 's and v 's to \hat{u} 's and \hat{v} 's respectively.

It can be observed that the convergence to a local minima property of the hard co-clustering scheme in [1] holds true for our overlapping co-clustering scheme also.

2.1 Plan of Action

We have so far formulated a generalized scheme to obtain overlapping co-clusters from a data matrix. Our plan of action is as follows

- Determine the appropriate summary statistics C for gene expression data.
- Determine a suitable Bregman Divergence for gene expression data. We believe that squared euclidean distance might be a suitable measure (as shown for hard co-clustering case [2]).
- Determine appropriate convergence criterion for the algorithm. One possible approach is to stop when \hat{U} or \hat{V} do not change significantly over consecutive iterations. Another approach is to stop when the Bregman Divergence between Z and \hat{Z} does not change significantly over consecutive iterations.
- Create synthetic data matrix with known dense overlapping co-clusters and illustrate the effectiveness of the proposed algorithm.
- Apply the proposed algorithm on gene expression data of *Yeast* and evaluate the results.

References

- [1] A. Banerjee, I. Dhillon, J. Ghosh, S. Merugu, and D. S. Modha. A generalized maximum entropy approach to bregman co-clustering and matrix approximation. In *KDD '04: Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 509–514, New York, NY, USA, 2004. ACM Press.
- [2] H. Cho, I. S. Dhillon, Y. Guan, and S. Sra. Minimum sum-squared residue co-clustering of gene expression data. In *Proceedings of SIAM Data Mining Conference*, pages 114–125, 2004.
- [3] I. S. Dhillon, S. Mallela, and D. S. Modha. Information-theoretic co-clustering. In *KDD '03: Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 89–98, New York, NY, USA, 2003. ACM Press.
- [4] G. Grahne and J. Zhu. Efficiently using prefix-trees in mining frequent itemsets. In *FIMI'03 Workshop on Frequent Itemset Mining Implementations*, November 2003.
- [5] C. Yang, U. Fayyad, and P. S. Bradley. Efficient discovery of error-tolerant frequent itemsets in high dimensions. In *KDD '01: Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 194–203, New York, NY, USA, 2001. ACM Press.