Simultaneous Co-clustering and Learning with Dimensionality Reduction

Harish Ganapathy
CS 229 Project Report
E-mail: harishg@utexas.edu

***Abstract*—In this project, we are interested in solving predictions problems involving *dyadic* data. The "Netflix problem", where data may be organized into a matrix with the rows representing users, the columns representing movies and (some of) the matrix elements containing ratings, is an example of a setting where the data is naturally dyadic. The prediction problem here is essentially to fill in the missing entries of the matrix with predicted ratings. Recently, a technique called *simultaneous co-clustering and learning (SCOAL)* was proposed that was demonstrated to be effective in solving such dyadic prediction problems. SCOAL is essentially a divide-and-conquer approach that seeks to form clusters of feature vectors and build a separate local model for each one of these clusters. Such an approach inherently suffers from the risk of overfitting, especially when the number of clusters is large. Thus, to alleviate the effects of over-fitting, while at the same time, benefitting from the use of multiple models, we propose the use of dimensionality reduction in conjunction with SCOAL. We investigate two dimensionality reduction techniques: *principle components regression* and the *least absolute shrinkage/selection operator (LASSO)*. We show that both these techniques provide significant reduction in test error (up to 10% in some settings) when linear predictive models are employed within each cluster.**

I. Introduction

In dyadic prediction problems, the training data consists of a pair of objects called *dyads*, with associated labels. The goal is to predict labels for the dyads that have missing labels. Apart from the popular "Netflix problem", another example of a dyadic prediction problem arises in marketing where we are interested in predicting whether a person will purchase an item given a history of purchases. In such settings, it is of interest to determine clusters of similar users and/or similar movies. The technique of co-clustering, which simultaneously clusters both axes, naturally lends itself to solving such problems. Co-clustering has been used in other problems such as text clustering and micro-array data analysis that involve dyadic data. Once the clusters[1] are determined, one may build a predictive model on a per-cluster basis. Such an approach finds its roots in the general wisdom that partitioning the feature space into multiple homogeneous segments and building more accurate models for each segment, often works well. It follows that such a technique would be most effective when the underlying generative model for the data resembles a mixture of Gaussians model for instance.

A recent paper by Deodhar and Ghosh [1] proposes a novel extension to the above sequential approach of co-clustering *followed by* learning. They propose a *simultaneous* co-clustering and learning algorithm that essentially recognizes that the ratings or labels inherently contain information pertaining to the similarity amongst the users and must hence be used in the formation of co-clusters. Since the optimal joint co-clustering and learning algorithm is of course NP-hard, the proposed greedy algorithm alternates between *(i)* learning the best models given a co-clustering and *(ii)* learning the best co-clustering given a set of local models. This algorithm is implicitly made possible by the fact that the model that predicts a given matrix element, operates on the corresponding row (user) and column (movie) attributes. Therefore, the divide-and-conquer-based prediction algorithm proposed by Deodhar and Ghosh [1] is applicable to settings where such additional attribute information is indeed available.

While divide-and-conquer is often an effective approach to improving prediction performance, one has to guard against over-fitting the training data as the number of co-clusters increases (equiv. as the cluster size decreases). This is especially critical for high-dimensional data (attributes) where it would be quite easy to exactly fit a small number of labels (located inside a small co-cluster). To address this challenge, in this paper, we propose to extend the algorithm introduced by Deodhar and Ghosh [1] to include dimensionality reduction. This means that when building local models for each co-cluster or sub-matrix, we include an additional dimensionality reduction step, which reduces the size or number of parameters that constitute the model. We consider two methods of dimensionality reduction that overlay the SCOAL algorithm:

(i) Principal component regression (PCR) - Given a co-cluster with a set of user-movie attribute vectors, we retain only the strongest eigenmodes (through singular value decomposition) of this set of user-movie attribute vectors and build a model from this set of eigenmodes to the labels. The number of eigenmodes that is retained is determined by an upper bound on the eigenmass, which is an additional input to the algorithm. Thus, this approach exploits any low-dimensional structure that may be present in the feature vectors belonging to the co-cluster.

(ii) LASSO regularization - One may more directly eliminate attributes that are not useful, i.e., not correlated to the labels, by using a more standard technique such as regularization. While the SCOAL framework in general accommodates any type of model within a co-cluster, LASSO is applicable specifically when linear regression/classification is used within each co-cluster.

We demonstrate the gains of dimensionality reduction in the context of a recommender system application. The main results in the report are as follows:

- We consider the well-known MovieLens dataset that was made available by the GroupLens Research Project at the

[1]We use the terms clusters and co-clusters interchangeably in this report.

University of Minnesota [3]. We show that as the number of clusters increases, both LASSO and PCR decrease the mean-squared test error when overlaid on plain SCOAL. In particular, PCR provides gains of up to 10% in as the number of clusters increases.

- We consider the ERIM data set, widely-used in the marketing community, consisting of household panel data collected by A.C. Nielsen [5]. We show that SCOAL-PCR with linear local models does not offer any gains over plain SCOAL in this setting.
- We study the use of PCR in conjunction with non-linear local models. In particular, we apply regression trees within each co-cluster. As with the ERIM data set, we show that SCOAL-PCR does not offer any gains here.

II. Problem Definition

We describe the problem formulation in this section. There are a total of m customers/users and n items. Matrix Z contains the ratings that the users have given the items; $z_{mn} \in \mathbb{R}_+$ denotes the rating given to item n by user m. It is also possible that a user has not rated an item. This gives rise to a natural matrix prediction problem, where given a set of ratings, we are interested in predicting in the missing cells. Let matrix W with

$$w_{ij} = \begin{cases} 0, & \text{cell } (i,j) \text{ has a missing entry} \\ 1, & \text{otherwise} \end{cases} \tag{1}$$

encode the locations of the missing ratings. Let $\mathbf{p}_i \in \mathbb{R}^{d_1},\ i = 1, 2, \ldots, m$, (resp. $\mathbf{q}_j \in \mathbb{R}^{d_2},\ j = 1, 2, \ldots, n$,) be a d_1-dimensional (resp. d_2-dimensional) feature/attribute vector that is associated with customer i (resp. item j). Thus, associated with each user-item dyad, we can assemble a cumulative feature vector $\mathbf{x}_{ij} = \left[1\ \mathbf{p}_i^T \mathbf{q}_j^T\right]^T \in \mathbb{R}^{d_1+d_2+1}$. The regression problem of interest is to find a map from the set of features $\mathbf{x}_{ij}$ to the positive reals such that the missing ratings are predicted accurately, in the sense of mean-squared error.

In the next section, we describe the SCOAL algorithm that breaks up the data into smaller co-clusters or sub-matrices and builds a model within each co-cluster.

III. Simultaneous co-clustering and learning with dimensionality reduction

We introduce the following notation to facilitate co-clustering. Let K and L be the total number of row and column clusters respectively. In particular, let $\mathcal{C}_k,\ k = 1, 2, \ldots, K$ denote the k-th column cluster and $\mathcal{R}_l,\ l = 1, 2, \ldots, L$ denote the l-th row cluster. For convenience, let $r : \{1, 2, \ldots, m\} \to \{1, 2, \ldots, K\}$ and $c : \{1, 2, \ldots, n\} \to \{1, 2, \ldots, L\}$ denote mappings from the customer (resp. item) index to its associated row (resp. column) cluster. K and L are treated as inputs to the problem in our formulation, making the presented results applicable to settings where model selection is not feasible.

For a given K and L, the regression problem that SCOAL seeks to solve can be written as

$$\begin{array}{ll} \min & \sum_{i=1}^{m}\sum_{j=1}^{n} w_{ij}(z_{ij} - \boldsymbol{\beta}_{c(i)r(j)}^T \mathbf{x}_{ij})^2 \\ \text{w.r.t} & \mathcal{C}_i,\ \mathcal{R}_j,\ \boldsymbol{\beta}_{ij},\ i = 1, \ldots, K,\ j = 1, 2, \ldots, L. \end{array} \tag{2}$$

The above problem is NP-hard and hence, SCOAL is essentially a greedy approximation algorithm. Note that while (2) assumes the use of linear prediction , i.e., $\boldsymbol{\beta}_{c(i)r(j)}^T \mathbf{x}_{ij}$, within each co-cluster for notational simplicity, we may in general use any predictive model here. In Algorithm 1 below, we reproduce the pseudocode for the algorithm from [1]. As motivated in the introduction, the original approach in Algorithm 1 may tend to overfit the data especially as the size of the co-cluster decreases (equiv. as K and L increase). In order to address this issue, we propose two variants or overlays to the above approach.

Algorithm 1 SCOAL

Input: convergence error threshold ε, Z, W, random co-clustering $\mathcal{C}_i,\ \mathcal{R}_j,\ i = 1, \ldots, K,\ j = 1, 2, \ldots, L$.
while True **do**
  Build models:
  for $k = 1, \ldots, K$ **do**
    for $l = 1, \ldots, L$ **do**
      Train linear regression model $\boldsymbol{\beta}_{kl}$ given data points $(\mathbf{x}_{ij}, z_{ij}),\ i \in \mathcal{C}_k,\ j \in \mathcal{R}_l,\ w_{ij} = 1$.
    end for
  end for
  Update row cluster assignment $r(\cdot)$: Assign each row to cluster that minimizes row error:
  for $i = 1, \ldots, m$ **do**
    $r(i) = \arg\min_{k=1,\ldots,K} \sum_{j=1}^{L} \sum_{s:c(s)=j} w_{is}(z_{is} - \boldsymbol{\beta}_{kj}^T \mathbf{x}_{is})^2$
  end for
  Update column cluster assignment $c(\cdot)$: Assign each row to cluster that minimizes row error:
  for $j = 1, \ldots, n$ **do**
    $c(j) = \arg\min_{l=1,\ldots,L} \sum_{i=1}^{K} \sum_{t:r(t)=i} w_{jt}(z_{jt} - \boldsymbol{\beta}_{li}^T \mathbf{x}_{jt})^2$
  end for
  Measure error at iteration n and store in error$[n]$.
  if $|\text{error}[n] - \text{error}[n-1]| \le \varepsilon$ **then**
    break while loop.
  end if
end while

The first variant, called *Principle Components Regression (PCR)* [2], essentially seeks to exploits any low-dimensional structure that may be present in the set of features attached to the particular co-cluster. The technique projects the features onto a lower-dimensional space and performs the regression using these "latent" features. Here, each latent feature is a linear combination of all the features.

The second variant is a more standard regularization approach called LASSO [4]. Here, an ℓ_1-penalty term is added to the objective function in (2) in order to encourage sparsity. Whereas PCR tries to succinctly summarize all the features in the form of a few latent features, LASSO attempts to eliminate features that are not useful.

The two proposed SCOAL overlays are presented below in more detail.

A. Principal components regression

We describe the PCR algorithm for a single co-cluster. Consider an arbitrary co-cluster $(\mathcal{C}_i, \mathcal{R}_j)$ and let $s = 1, 2, \ldots, |\mathcal{C}_i||\mathcal{R}_j|$ denote a serialized index obtained by counting top-down through each column of the co-cluster. Similarly, given a matrix $A \in \mathbb{R}^{m \times n}$, let $\text{Vec}\{A\} \in \mathbb{R}^{mn \times 1}$ denote the natural vectorization of this matrix by counting top-down through each column of the matrix. For convenience, let $A(\mathcal{S})$ denote a sub-matrix $\mathcal{A}$ formed by retaining rows indexed by set $\mathcal{S}$. Next, let $X = \mathbf{x_1 x_2} \ldots \mathbf{x}_{|\mathcal{C}_\mathbf{i}||\mathcal{R}_\mathbf{j}|}]^\mathbf{T}$ be a matrix that is

formed by collecting together all feature vectors belonging to a co-cluster. Let $\mathcal{S}_{kl} = \{i : w_i = 1,\ i = 1, 2, \ldots, |\mathcal{C}_i||\mathcal{R}_j|\}$ index the set of rated items in co-cluster (k, l). The basic idea of PCR is to use the SVD to form a set of latent features in a lower-dimension. To that effect we decompose $X(\mathcal{S}_{kl})$ as $X(\mathcal{S}_{kl}) = U\Lambda V^T$ where $U^TU = I$, $V^TV = I$ and Λ contains the singular values $\sigma_1^2 \geq \sigma_2^2 \geq \ldots \geq \sigma_{d_1+d_2+1}^2$ arranged in descending order. The columns of $T = U\Lambda$ are often referred to as the principle components. We then form a rank-p approximation

$$X(\mathcal{S}_{kl}) \approx T_p V_p^T, \tag{3}$$

where T_p and V_p retain the first p columns of T of V respectively.

In the context of our algorithm, we are interested in controlling the rank of the approximation through the eigenvalue mass. In other words, given an upper bound $\lambda \in (0, 1]$, we choose rank p such that

$$p = \min\left\{p' : \frac{\sum_{c=1}^{p} \sigma_c^2}{\sum_{c=1}^{d_1+d_2+1} \sigma_c^2} \geq \lambda\right\};$$

λ represents the fraction of eigenvalue mass that should be retained. An advantage of working with eigenvalue masses instead of direct rank values is that this allows the model to retain a fixed amount of "information" since eigenvalue masses may be treated as proxies for the amount of information in a particular direction. With rank p chosen as described above, we may form the low-rank approximation $T_p(\lambda)$ to $X(\mathcal{S}_{kl})$ and then solve the following PCR regression problem

$$\begin{array}{ll} \min & ||Y(\mathcal{S}_{kl}) - T_p(\lambda)^T\boldsymbol{\beta}||^2 \\ \text{w.r.t} & \boldsymbol{\beta}. \end{array} \tag{4}$$

within co-cluster (k, l), where $Y = \text{Vec}\{Z\}$. The closed-form solution to (4) is well-known and is given by

$$\boldsymbol{\beta}^*(\lambda) = (T_p(\lambda)^T T_p(\lambda))^{-1} T_p(\lambda)^T Y(\mathcal{S}_{kl}). \tag{5}$$

Note that while the SVD does indeed place an added computational burden, the model in (5) can be computed efficiently since the columns of T_p are orthogonal. Thus, the SCOAL-PCR variant to SCOAL may be obtained by replacing all features in Algorithm 1 by the corresponding latent features.

Once the models are constructed, we can now turn our attention to the prediction problem. In order to predict the missing entries, i.e., those indexed by $\{(i, j) : w_{ij} = 0\}$, we first project the feature matrix X_{test} onto the lower-dimensional space $T_{test}(\lambda) = X_{test}V_p(\lambda)$ and then apply the regression model to the transformed features to obtain the predictions $\text{Vec}\{\hat{Z}\} = T_{test}(\lambda)\boldsymbol{\beta}^*(\lambda)$. The error within co-cluster (k, l) is then measured as

$$\frac{1}{|\mathcal{S}_{kl}|}||\text{Vec}\{\hat{Z}\} - \text{Vec}\{Z\}||^2.$$

Having described the PCR overlay algorithm, we now move on to the LASSO approach that we review briefly as it is already well-studied in the literature.

B. LASSO regularization

As mentioned earlier, while PCR seeks to retain (in part) the information contained in all features, Lasso on the other hand tried to discard features by promoting sparsity in the linear models. The LASSO regression problem within each co-cluster can be written as

$$\begin{array}{ll} \min & \sum_{i=1}^{m}\sum_{j=1}^{n} w_{ij}(z_{ij} - \boldsymbol{\beta}^T\mathbf{x}_{ij})^2 + \gamma||\boldsymbol{\beta}||_1 \\ \text{w.r.t} & \boldsymbol{\beta} \end{array}. \tag{6}$$

for some appropriately chosen $\gamma > 0$, where $||\cdot||_1$ denotes the ℓ_1-norm. As γ increases, the amount of sparsity increases at the cost of training accuracy. The SCOAL-LASSO variant may be obtained by replacing Step 6 in Algorithm 1 by (6). Note that, in contrast to PCR, LASSO is only applicable when using linear models inside each co-cluster.

In the next section, we present some preliminary results that quantify the performance of the proposed dimensionality reduction techniques.

IV. Results: Recommender systems

We first apply the algorithms to the MovieLens dataset. The dataset consists of $100,000$ ratings from 943 users and 1682 movies. Each user has rated at least 20 movies. In the interest of expediting the run-time, we reduce the size of the data set. We choose the top 378 users that have rated the most number of movies. Following this initial pruning step, we prune further by choosing the top 673 movies that have the most number of ratings. This amounts to retaining 40% of the initial data set. Of the user attributes that are available, we consider age and occupation. The occupation is represented as a binary vector on a pre-specified set of occupations (e.g., administrator, artiste, doctor, etc.). The movie attributes that we use are release date and genre. The genre is again a binary vector on a pre-specified set of genres (e.g., action, adventure, animation, etc.). The feature dimensions are $d_1 = 21$ and $d_2 = 20$ for this application. In addition, we pre-process each feature vector by subtracting away the mean and dividing by the standard deviation.

A. Linear local models

In our first experiment, we apply linear regression within each co-cluster and evaluate the performance with and without dimensionality reduction through PCR. The experiment is conducted as follows:

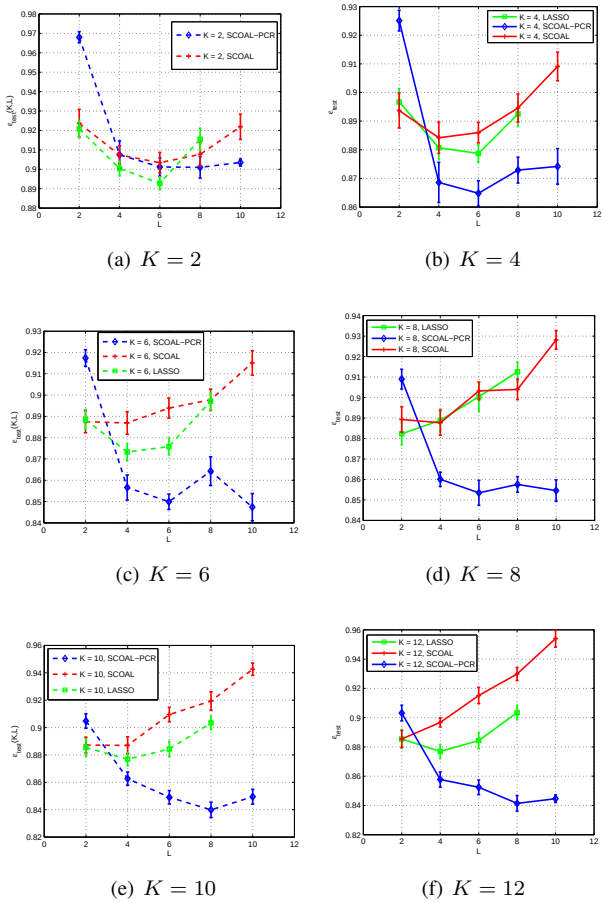
- *Step 1*: We choose some (K, L).
- *Step 2*: We partition the data into training and test data. This is accomplished by designating 80% of the available user-movie pairs as training and the remaining available data as test. Denote this partition by $\mathcal{P}_1$.
- *Step 3*: We vary the eigenvalue mass λ across set $\{0.1, 0.2, 0.3, \ldots, 1\}$ and for each value, we record the training error $\varepsilon_{train}(\lambda(\mathcal{P}_1), \mathcal{P}_1)$ under SCOAL-PCR.
- *Step 4*: We choose the value $\lambda^*(\mathcal{P}_1)$ with the minimum training error.
- *Step 5*: Once chosen, we apply SCOAL-PCR with eigenvalue mass λ^* to the test data and record the test error $e(\lambda^*(\mathcal{P}_1), \mathcal{P}_1)$.
- *Step 6*: We then repeat Steps 2 - 5 and record $e(\lambda^*(\mathcal{P}_i), \mathcal{P}_i)$ under different train/test partitions $\mathcal{P}_i,\ i = 1, 2, \ldots, 10$.
- *Step 7*: We finally compute the average test error $\varepsilon_{test}(K, L) = \frac{1}{10}\sum_{k=1}^{10} e(\lambda^*(\mathcal{P}_k), \mathcal{P}_k)$ and the associated standard deviation.

- *Step 8*: We repeat the above experiment under SCOAL-LASSO and record the corresponding average test error.
- *Step 9*: Finally, we repeat the above experiment with plain SCOAL in Algorithm 1 and record the corresponding average test error.

In Figs.1(a)-1(f), we compare the test error $\varepsilon_{test}(K, L)$ under SCOAL, SCOAL-PCR and SCOAL-PCR for different configurations $(K, L) \in \{2, 4, 6, 8, 10, 12\} \times \{2, 4, 6, 8, 10\}$. We can clearly make the following observations from the plots.

For the SCOAL algorithm, for fixed/small values of K such as $K \in \{2, 4\}$ in Figs.1(a) and 1(b), as L increases, we first see a decrease in test error due to a better model fit. However, as we approach large L, i.e., employ a large number of local models, we see the effects of over-fitting. For larger values of K such as $K \in \{10, 12\}$, since we are already in the regime of over-fitting, as L increases, we only see further over-fitting and an increase in error as shown in Figs.1(e) and 1(f).

The story is quite different with the SCOAL-PCR algorithm however. From Figs.1(a)-1(f), we see that SCOAL-PCR effectively combats over-fitting. In particular, SCOAL-PCR significantly outperforms SCOAL as L increases. The performance gain are quite pronounced as (K, L) increases and is plotted in Fig.2 at $L = 10$. We see that for large K, the decrease in test error is roughly 10%. *Thus, by using the PCR overlap, we are able to derive the dual benefits of using multiple local models while guarding against over-fitting*.



(a) $K = 2$ (b) $K = 4$

(c) $K = 6$ (d) $K = 8$

(e) $K = 10$ (f) $K = 12$

Fig. 1. Average test error versus number of column clusters L for the three algorithms on the MovieLens data set.

Finally, we have also plotted the test error due to the LASSO algorithm in Figs.1(a)-1(f). We see that as (K, L) increases, its performance lies between that of SCOAL and SCOAL-PCR. Note that the LASSO solution is computationally less demanding than SCOAL-PCR since the latter involves calculating the SVD of the feature matrix. Nevertheless, while, some of the gains may be attributed to the additional computation, the remainder is due to the fact that PCR summarizes features and retains as much information as possible while LASSO eliminates features. Before we move on to the next section,

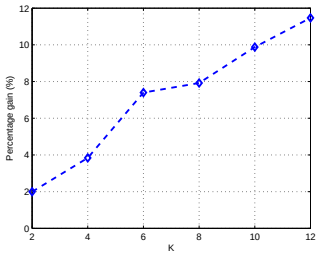


Fig. 2. Decrease in test error between SCOAL-PCR and SCOAL as a function of K at $L = 10$.

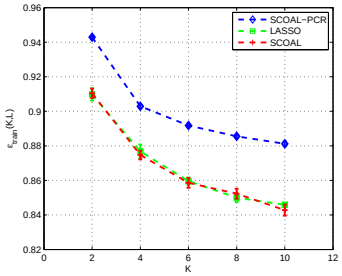


Fig. 3. Training error versus number of rows clusters L with $K = 2$.

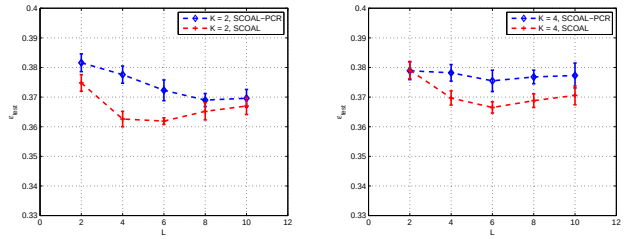
we plot the training error under SCOAL, SCOAL-LASSO and SCOAL-PCR for the cases $L = 2$ and $K \in \{2, 4, 6, 8, 10, 12\}$. The training error under SCOAL-PCR is seen to be the highest while the test error in Figs.1(a)-1(f) at $L = 2$ is also seen to be the largest. This could indicate high bias due to the fact that compressing the feature dimension simplifies the model. That said, more investigation is needed here. An analysis of the eigenvalue mass that is retained in each (K, L) configuration would shed light on the precise amount of compression that is being applied at each stage.

B. Results on ERIM dataset

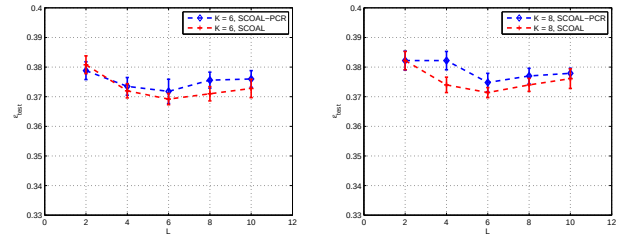
Next, we compare the performance of SCOAL vs. SCOAL-PCR (i.e., Step 1-Step 8) on the ERIM data set, consisting of household panel data collected by A.C. Nielsen [5]. In particular, the data set contains past information about 1714 products that were purchased by a set of 121 households (if one were to ignore households that have made less than two purchases). There are three product attributes - market share, price and the number of times the product was advertised. The household attributes are income, number of residents, male head employed, female head employed, total visits and total expense. Thus, the feature dimensions are $d_1 = 6$ and $d_2 = 3$ for this application. As was done with MovieLens, we pre-process each feature vector by subtracting away the mean and dividing by the standard deviation. Element (i, j) of the data matrix contains the number of units of product j that was

purchased by household i. The data matrix is sparse with around 75% of the values being zero. The data matrix also contains outliers - while 99.12% of the values are below 20, some values are very large and range up to 200.

The results are provided in Figs. 4(a)-4(d). We see that the PCR overlay is not as effective in this setting. More specifically, for lower values of K such as $K \in \{2, 4\}$, SCOAL-PCR underperforms SCOAL while for larger values $K \in \{6, 8\}$, the test error is roughly the same. There might be multiple reasons for this trend. Firstly, the feature dimension is already quite small (nine features) in the case of this data set and furthermore, the sparsity in data might lead to poor PCR models in the latent (compressed) feature space. Further investigation is needed here to determine the root cause for the lukewarm performance of PCR.



(a) $K = 2$ (b) $K = 4$



(c) $K = 6$ (d) $K = 8$

Fig. 4. Average test error versus number of column clusters L for the three algorithms on the ERIM data set.

In the next section, we revert back to the MovieLens data set but this time however, we brief study the use of non-linear local models.

C. Non-linear local models

While the SCOAL algorithm as presented in Algorithm 1 uses linear regression as the predictive model within each co-cluster, in its most general form [1], it is a framework that accommodates any model within each co-cluster.

We briefly study the impact of employing PCR dimension reduction in conjunction with regression trees inside each co-cluster. In other words, we first reduce the dimension of the feature matrix X as per (3) and then construct a regression tree. The prediction that is made at any given leaf consists of the centroid of all the training feature vectors that percolate down to that leaf. We control the size of the tree (and hence the tendency to overfit) by enforcing a "split" criterion - a node is allowed to split and add another level to the tree only if the number of training feature vectors that percolate down to that node exceeds 50% of the total size of the data in the co-cluster (the number of rows of feature matrix X).

We repeat Steps 1-Step 8 for an assortment of cluster sizes. The results are presented in Table I. As with the earlier results with linear models on the ERIM dataset, we do not see any significant gains due to PCR. A more comprehensive analysis is necessary at this point but was not possible due to lack of time.

TABLE I
SCOAL-PCR VERSUS SCOAL WITH REGRESSION TREES.

Algorithm	(K, L) configuration	Average test error
SCOAL	(2,2)	0.9620
SCOAL	(4,4)	0.8783
SCOAL	(6,2)	0.9220
SCOAL	(6,4)	0.8629
SCOAL-PCR	(2,2)	0.9664
SCOAL-PCR	(4,4)	0.8878
SCOAL-PCR	(6,6)	0.8797
SCOAL-PCR	(6,8)	0.8815
SCOAL-PCR	(8,6)	0.8765
SCOAL-PCR	(8,8)	0.8868

V. CONCLUDING REMARKS

In this project, we have shown that dimensionality reduction may be used effectively in conjunction with the SCOAL divide-and-conquer approach. In particular, PCR-based dimensionality reduction performs increasingly better as the number of co-clusters increases. SCOAL-PCR outperforms plain SCOAL by almost 10% as we approach 100 co-clusters when using linear models inside each co-cluster. Thus, in large-data applications where it might be too expensive to perform model selection and select a suitable (K, L) configuration, it might be a safe bet to employ the SCOAL-PCR algorithm with a large number of co-clusters. LASSO on the other hand provides a low-complexity alternative to PCR while paying a price in terms of test error. LASSO-based feature selection does still outperform plain SCOAL as the number of co-clusters increases.

Further research is needed to systematically determine and quantify the reasons for the gains, or lack thereof, due to PCR and/or LASSO. As mentioned earlier, a detailed analysis of the eigenvalue mass that is retained for each (K, L) configuration would give us some more insight. Finally, it is worth considering alternate dimensionality reduction techniques such as partial least squares, which projects the features onto a basis that captures most variance across *both* the features and the labels. Principle components regression ignores the labels and finds directions that maximize the variance amongst *only the* feature vectors.

REFERENCES

[1] M. Deodhar and J. Ghosh, "Simultaneous Co-clustering and Learning from Complex Data", Proc. 13th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, August 2007.

[2] I.T. Jolliffe, "A note on the Use of Principal Components in Regression," *Journal of the Royal Statistical Society, Series C (Applied Statistics)*, vol. 31, pp. 300-303, 1982.

[3] MovieLens data set, http://www.grouplens.org/system/files/ml-data.tar__0.gz, University of Minnesota.

[4] R. Tibshirani, "Regression shrinkage and selection via the lasso", *J. Roy. Statist. Soc. Ser. B*, pp. 267-288, 1996.

[5] Kilts Center for Marketing, ERIM Database, http://research.chicagobooth.edu/marketing/databases/erim/.