Consiglio Nazionale delle Ricerche
Istituto di Calcolo e Reti ad Alte Prestazioni

A Hierarchical Probabilistic Model for Co-Clustering High-Dimensional Data

Gianni Costa, Francesco Folino,
Giuseppe Manco, Riccardo Ortale

RT-ICAR-CS-06-04 **Maggio 2006**

Consiglio Nazionale delle Ricerche, Istituto di Calcolo e Reti ad Alte Prestazioni (ICAR)
– Sede di Cosenza, Via P. Bucci 41C, 87036 Rende, Italy, URL: *www.icar.cnr.it*
– Sezione di Napoli, Via P. Castellino 111, 80131 Napoli, URL: *www.na.icar.cnr.it*
– Sezione di Palermo, Viale delle Scienze, 90128 Palermo, URL: *www.pa.icar.cnr.it*

Consiglio Nazionale delle Ricerche
Istituto di Calcolo e Reti ad Alte Prestazioni

A Hierarchical Probabilistic Model for Co-Clustering High-Dimensional Data

Gianni Costa[1], Francesco Folino [1],
Giuseppe Manco [1], Riccardo Ortale[1]

Rapporto Tecnico N.: **RT-ICAR-CS-06-04**

Data: **Maggio 2006**

---

[1] Istituto di Calcolo e Reti ad Alte Prestazioni, ICAR-CNR, Sede di Cosenza, Via P. Bucci 41C, 87036 Rende(CS)

A Hierarchical Probabilistic Model for Co-Clustering High-Dimensional Data

Gianni Costa, Francesco Folino, Giuseppe Manco, and Riccardo Ortale

ICAR-CNR
Via Bucci 41c
87036 Rende (CS) - Italy

Abstract. We propose a hierarchical, model-based co-clustering framework for handling high-dimensional datasets. The technique views the dataset as a joint probability distribution over row and column variables. Our approach starts by initially clustering rows in a dataset, where each cluster is characterized by a different probability distribution. Subsequently, the conditional distribution of attributes over tuples is exploited to discover co-clusters underlying the data. An intensive empirical evaluation confirms the effectiveness of our approach, even when compared to well-known co-clustering schemes available from the current literature.

1 Introduction

Increasing attention has been recently paid to clustering high dimensional data, since this task is of great practical importance in several emerging application settings such as text analysis, bioinformatics, e-commerce, astronomy, statistics and psychology and insurance industry [1, 2, 11, 13]. However, clustering high-dimensional data poses some challenging issues.

Foremost, data sparseness and/or skewness as well as attribute irrelevancy and/or redundancy typically impose to look for valuable clusters within several subsets of the original attribute space. This inevitably penalizes the effectiveness of clustering and further exacerbates its time requirements, since high dimensional data tends to exhibit different clusters on distinct attribute subsets. Although standard dimension reduction techniques [6] can be used to detect the relevant dimensions, these can be different for distinct clusters, which invalidates such a preprocessing task.

Also, the identification of cohesive clusters is a major concern. In most cases, cohesion is measured in terms of the syntactic similarity of the objects in a cluster. However, several irrelevant attributes might distort the actual degree of proximity between object tuples. Moreover, clustering schemes yield *global patterns*, that do not apparently capture our general understanding of complex phenomenons. Indeed, in a high-dimensional setting, specific groups of objects tend to be co-related only under certain subsets of attributes. Hence, though semantically-related, two tuples with (possibly several) differences in their attribute values would hardly be recognized as actually similar by any global

model. In principle, object cohesion is better viewed in terms of *local patterns*. Precisely, the individual data tuple can be intended as a mixture of latent concepts, each of which being a suitable collection of characterizing attributes. Accordingly, two tuples are considered as actually similar if both represent at least a same concept. Viewed in this perspective, the identification of local patterns, i.e. of proper combinations of object tuples and attributes, leads to the discovery of natural clusters in the data, without incurring into the foresaid difficulties.

Co-clustering has recently gained attention as a powerful tool, that allows to circumvent the aforementioned limitations while processing high-dimensional data. Due to its intrinsic capability at exploiting the latent relationships between tuples and their own attributes, it enables the discovery of coherent clusters of similar tuples and their interplay with corresponding attribute clusters. This has been the main motivation behind the development of a wealth of new, *ad hoc* techniques that simultaneously cluster both object tuples and their attributes.

Co-clustering techniques can be divided into the five main categories [14], discussed next. The simplest class of approaches [8, 16] is the one that applies existing clustering methods to find independent row and column partitions and then combines the results into meaningful co-clusters. Divide-and-conquer strategies, such as [9], divide the original co-clustering problem into multiple subproblems of smaller size, solve them recursively and then combine the resulting solutions into an actual solution for the initial problem. Greedy iterative algorithms [4, 3] search for co-clusters in the data matrix by progressively removing or adding rows or columns, in an attempt at maximizing some local-quality criterion. Techniques based on exhaustive co-cluster enumeration [15, 17] search for all possible co-clusters in the data matrix. Model-based techniques [7, 10, 18] assume a suitable model for the data generation process and learn estimates of model parameter values from the available data. To the best of our knowledge, the approach in [18] is the most resemblant to our proposal. However, in this regard, we emphasize that the application of the EM algorithm for learning suitable estimation of parameters is not direct, due to structural dependencies in the underlying model [18], that requires suitable approximations. To the purpose, [18] pursues the maximization of a variational approximation of data likelihood, via Generalized EM [19]. On the contrary, we assume a hierarchical model for the representation of the data generating process, that allows a more direct and natural exploitation of the EM algorithm.

In this paper, we build on probabilistic techniques to develop an innovative, model-based algorithm for the discovery of actual co-clusters in high-dimensional data. The underlying intuition is that an object tuple can be thought of as the outcome of the following hierarchical, generative process: firstly pick a distribution over latent clusters; next, choose the associated concepts; eventually, generate the individual attribute values. An EM-based clustering strategy fits the probabilistic model of the foresaid generative model to the underlying data. Precisely, the joint probability distribution over row (i.e. tuple) and column (i.e. attribute) variables is exploited to initially find tuple clusters. Then, the conditional distribution of attributes over tuples is exploited to discover actual co-

clusters, i.e. for associating concept clusters with tuple clusters. A preliminary evaluation of our approach and a comparison with consolidated co-clustering schemes in the literature seem to confirm the validity of our intuition.

The plan of the paper is as follows. Section 2 formally describes the intuition behind our model-based co-clustering approach. Section 3 discusses the details of the method employed for learning suitable estimates of model parameters from the underlying data. An intensive experimental evaluation is described in section 4, that witnesses the effectiveness of our proposal. Finally, section 4 draws some conclusions and highlights directions, that are worth further research.

2 Problem Statement and Overview of the Approach

We begin by fixing a proper notation to be used throughout the paper. Data can be represented in binary format as a boolean incidence matrix D with rows $\{y_1, \ldots, y_m\}$ and columns $\{x_1, \ldots, x_n\}$, where each entry d_{ij} takes values into the set $\{0, 1\}$. The implicit meaning is that tuple x_j comprises attribute y_i if and only if d_{ij} takes value 1. Let X and Y be discrete random variables ranging over the sets $\{x_1, \ldots, x_n\}$ and $\{y_1, \ldots, y_m\}$. We are interested in simultaneously clustering X into K disjoint clusters, and Y into H disjoint clusters. That is, we aim at finding suitable column and row mappings, respectively defined as $C_X : \{x_1, \ldots, x_n\} \mapsto \{\hat{x}_1, \ldots, \hat{x}_K\}$ and $C_Y : \{y_1, \ldots, y_m\} \mapsto \{\hat{y}_1, \ldots, \hat{y}_H\}$

In our framework, we characterize the co-occurrence matrix in probabilistic terms, by estimating the joint distribution $p(X, Y)$ between X and Y. By Bayes' rule, the distribution can be modeled as $p(x, y) = p(y|x)p(x)$. Model-based clustering methods attempt to optimize the fit between the given data and some mathematical model. Such methods are often based on the assumption that the data to be clustered are generated by one of several distributions, and the goal is to identify the parameters of each. The foundation for probabilistic clustering is a statistical model called *finite mixtures*. A *mixture* is a set of probability distributions, representing clusters that govern the format for members of that cluster.

Each cluster has a different distribution. Any particular instance actually belongs to one and only one of the clusters, whose identity is however unknown. Moreover, the clusters are not equally likely: there is some probability distribution that reflects their relative populations. Within a mixture modeling framework, the above components can be described as reported below

$$p(x) = \sum_{k=1}^{K} p_k(x)\alpha_k \qquad\qquad p(y|x) = \sum_{h=1}^{H} p_h(y) \cdot \sum_{k=1}^{K} p(\hat{x}_k|x)\beta_{h,k}$$

where $p_k(x) = p(x|\hat{x}_k)$ is the probability of x within cluster $\hat{x}_k$, $p_h(y) = p(y|\hat{y}_h)$ is the probability of y within cluster $\hat{y}_h$, $\alpha_k = p(\hat{x}_k)$ is the probability of cluster $\hat{x}_k$, and $\beta_{h,k} = p(\hat{y}_h|\hat{x}_k)$ is the probability of cluster $\hat{y}_h$ given cluster $\hat{x}_k$. As a consequence, the mixture $p(x, y)$ can be finally modeled as

$$
\begin{aligned}
p(x,y) &= \sum_{h=1}^{H} p_h(y) \cdot \sum_{k=1}^{K} p(\hat{x}_k|x)\beta_{h,k} \cdot \sum_{k=1}^{K} p_k(x)\alpha_k \\
&= \sum_{h=1}^{H}\sum_{k=1}^{K}\sum_{u=1}^{K} p_h(y)\beta_{h,k}p_u(x)\alpha_u p(\hat{x}_k|x)
\end{aligned}
\tag{1}
$$

The idea in the above formula is learning latent concepts from the data as well as a collection of characterizing attribute values for each such a concept. In particular, each tuple can be seen as a mixture of various concepts, where some concepts are more or less probable according to the cluster where the tuple fits. Hence, a data tuple can be thought as the outcome of the following generative model: firstly pick a distribution over latent clusters; next, choose the concepts associated and finally generate the individual attribute values.

The clustering problem can be hence reformulated as the problem of estimating the parameters of the distributions involved in the above formula. The classical *Maximum Likelihood* (*ML*) Estimation technique is a way for estimating the parameters of a distribution based upon observed data drawn according to that distribution. Let Θ denote a set of parameters and let x, y be a data observed from the random variables X, Y with probability distribution $p_{X,Y}(x, y|\Theta)$, parameterized by the set of parameters Θ. The key idea in *ML* estimation is to determine the parameter Θ for which the probability of observing the outcome x is maximized. Function $\mathcal{L}(\Theta|X, Y) = p(X, Y|\Theta)$ is the *Likelihood function* and the *Maximum Likelihood* Estimation of the parameter Θ is the value which maximizes the likelihood function $\Theta_{ML} = \arg\max_{\Theta} \mathcal{L}(\Theta|X, Y)$.

In our framework, Θ represents the set of parameters governing p_k, p_h, α_k and $\beta_{h,k}$ for each h and k. We adopt a naive assumption here, that is $\beta_{h,k} \in \{0, 1\}$ and $\sum_h \beta_{h,k} = 1$. As a consequence, $p_h(y)$ can be modeled as the probability of term y within tuple cluster k (that is, $p_h(y) = p_k(y)$). This roughly consists in associating a single concept cluster with each tuple cluster, and in modeling the probability of an attribute within a tuple cluster. Thus, the set of parameters Θ involved in the estimation are now represented by sole parameters related to (tuple) cluster $\hat{x}_k$ (i.e., the parameters governing $p_k(x)$, $p_k(y)$ and α_k). Moreover, the probability distribution can be rewritten as

$$
p(x,y) = \sum_{k=1}^{K}\sum_{u=1}^{K} p_k(y)p_u(x)\alpha_u p(\hat{x}_k|x) \tag{2}
$$

In particular, by modeling $p_k(x)$ by means of a multinomial distribution, the estimation of the above parameters can be accomplished by means of the traditional Expectation Maximization algorithm, which is described below. Thus, we can suppose that $x_i = \{n_i^1, n_i^2, \ldots, n_i^m\}$, where $n_i^c \in \{0, 1\}$ for each c. A multinomial distribution models a Bernoulli's distribution in several outcomes. It is characterized by a parameter σ_c, that represents the probability that an event of class c happens. The multinomial distribution for the generic cluster $\hat{x}_k$

is parameterized by $\sigma_c^k = p_k(y_c)$:

$$p_k(x_i) = \prod_{c=1}^{m} (\sigma_c^k)^{n_i^c}$$

Thus, the model parameter Θ collects every σ_c^k and α_k, for $c = 1, \ldots, m$ and $k = 1, \ldots, K$.

3 Multinomial Expectation Maximization

The *Expectation Maximization*(*EM*) [12] algorithm is a classical technique for model-based clustering. Given the dataset and a pre-specified number of clusters, the algorithm learns, for each instance, the membership probability of each cluster and, for each cluster, its descriptive model, i.e., the parameters that govern its generative process.

The algorithm requires some initial estimates for the parameters of the mixture model; given such parameters, a single EM iteration provides new parameter estimates which are proven not to decrease the likelihood of the model. The process is repeated until convergence, i.e. the likelihood of the mixture model at the previous iteration is sufficiently close to the likelihood of the current model. More precisely, the algorithm proceeds as follows:

1. *Initialization*: $g := 0$; Set initial values $\Theta^{(0)}$ for the parameter set Θ; compute $\mathcal{Q}^{(g)} = \log(\mathcal{L}(\Theta^{(g)}|X, Y))$.
2. *E step*: Use $\Theta^{(g)}$ to compute the membership probability $p^{(g)}(\hat{x}_k|x)$ of each object x to each cluster $\hat{x}_k$
3. *M step*: Update the model parameters $\Theta^{(g+1)}$, using values computed in the E Step; compute $\mathcal{Q}^{(g+1)} = \log(\mathcal{L}(\Theta^{(g+1)}|X, Y))$.
4. If $\mathcal{Q}^{(g+1)} - \mathcal{Q}^{(g)} \leq \epsilon$, stop. Else set $g := g + 1$ and restart from step 2.

We omit here the formal derivation of the steps which are at the basis of the EM approach (details can be found in the appendix of an extended version of this paper [5]). It is worth noticing, however, that the objective of the derivation is a heuristic method for maximizing the *Log-Likelihood function*

$$\log(\mathcal{L}(\Theta|X, Y)) = \sum_{i,j} \log\left(p(x_i, y_j|\Theta)\right) \approx \sum_{i,j} z_{ik} \log\left(\sum_{k=1}^{K}\sum_{u=1}^{K} p_k(y) p_u(x_i) \alpha_u p(\hat{x}_k|x_i)\right)$$

where $z_{ik} \in \{0, 1\}$ is a random variable representing the true cluster generating the data. The latter term in the equation is transformed, for the matter of convenience, into

$$\mathcal{Q}(\Theta, \Theta^{(g-1)}) = E[\log(\mathcal{L}(\Theta|X, Y, Z))|X, Y, \Theta^{(g-1)}]$$

The E and M steps, at the generic iteration g, can be shown to be as follows:

E Step. Working on $\mathcal{Q}(\Theta, \Theta^{(g-1)})$ yields

$$p^{(g)}(\hat{x}_k|x_i) = \gamma_{ik}^{(g)} = \frac{\alpha_k^{(g-1)} p_k(x_i|\Theta^{(g-1)})}{\sum_{j=1}^{K} \alpha_j^{(g-1)} p_j(x_i|\Theta^{(g-1)})} \tag{3}$$

M Step. The target of this step consists in finding the best set Θ of parameters that maximizes the likelihood $\mathcal{Q}(\Theta, \Theta^{(g-1)})$. By mathematical manipulations, we obtain

$$\alpha_r^{(g)} = \frac{1}{n} \sum_{i=1}^{n} \gamma_{ir}^{(g)} \qquad\qquad \sigma_r^{z(g)} = \frac{\sum_{i=1}^{n} \gamma_{iz}^{(g)} (mn_i^r + 1)}{\sum_{i=1}^{n} \sum_{t=1}^{m} \gamma_{iz}^{(g)} (mn_i^t + 1)}$$

for $r = 1, \ldots, m$ and $z = 1, \ldots, K$.

3.1 Parameter Initialization

Although the *EM* algorithm is guaranteed to converge to a maximum, this is a *local* maximum and may not necessarily be the same as the *global* maximum. Notice, in particular, that the first iteration needs some initial values for $\alpha_k^{(0)}$ and $\sigma_c^{k(0)}$. The better the initial choices, the higher the probability that the computed local maximum is also a global maximum.

Notice that trivial initializations, such as the one in which every mixing probability $\alpha_k^{(0)} = \frac{1}{K}$ and $\sigma_c^{k(0)} = \hat{\sigma}_c$ for each cluster k, do not work, since it poses the algorithm in an equilibrium (stall) condition. In particular, the side effect of such initializations is that the parameters (and consequently the likelihood) assume initial values and do not change throughout the next iterations.

We adopt a different strategy, which combines random sampling and k-Means. The idea is to select k initial instances $x_1, \ldots, x_k$ from the dataset by means of a random sampling. Then, the parameters σ_j^k relative to cluster k can be initialized by exploiting the values n_i^j derived from each instance x_i, plus laplacian smoothing (to avoid the situations where $\sigma_j^k = 0$). The α_k instead are assumed equally probable. Also, the choice of the k initial points can be strengthened by multiple executions of the k-means algorithm on the data, and choosing the best centroids for the estimation.

3.2 Estimating the Number of Clusters

The estimation of the correct number of clusters is accomplished by resorting to a Cross-Validation approach based on a penalized Log-Likelihood principle, as described below. We aim at finding the model parameters Θ maximizing the probability $p(\Theta|X, Y)$. By Bayes' rule,

$$P(\Theta|D) = P(D|\Theta)P(\Theta)$$

In logarithmic terms,

$$\log(P(\Theta|D)) = \log P(D|\Theta) + \log P(\Theta) = \log(\mathcal{L}(\Theta|D)) + \log P(\Theta)$$

The idea in the above formula is to counterbalance two opposing requirements: the fitting of the data and the complexity of the model. The log-likelihood function, which measures the fitting of the data to the model, increases when the value K increases: in particular, it reaches its maximum when $K = n$. By the converse, the probability of the model can be encoded by resorting to the minimum description length principle, which states that simpler models are preferable to more complex ones. Thus, the probability of a model is inversely proportional to the number of its parameters (which in turn depend from the value K). In practice, $P(\Theta)$ can be modeled as an exponential distribution w.r.t the size of Θ, i.e. $P(\Theta) = \alpha n^{-km}$ where α is a normalizing factor. Thus,

$$\log(P(\Theta|D)) = \log(\mathcal{L}(\Theta|D)) - km \log n + \log \alpha$$

The evaluation strategy hence consists in computing $\log(P(\Theta|X, Y))$ for each possible model represented by Θ, and in choosing the model where it is maximal. In particular, the strategy can be summarized as follows:

1. fix the values K_{min} and K_{max};
2. choose the number C of cross-validation trials;
3. for each trial c:
   - sample a subset D_{train} from D;
   - for K ranging from K_{min} to K_{max}:
   -- compute $\log(P(\Theta_K|D_{train}))^c$;
4. for each K, average the values $\log(P(\Theta_K|D_{train}))^c$ over c;
5. choose the value K^* such that $\log(P(\Theta_{K^*}|D_{train}))^{avg}$ is maximal.

4 Experimental Evaluation

Hereafter we analyze the behavior of the framework proposed in the previous section. The analysis is performed with the main objective of assessing the quality of the identified structures, i.e. whether the discovered clusters correspond to the actual homogeneous groups in the dataset.

The effectiveness issues are extensively investigated. Experiments are conducted on both real and synthesized data. The result of each experiment is a matrix D where rows and columns are associated with their cluster of membership. Hence, a partition of the matrix in blocks where each block represents a cluster can give us a visual perception of the quality of the clustering result. Ideally, a good clustering would produce a block-triangular matrix, provided that a suitable sorting of both rows and columns is produced.

The incidence matrix also enables a simple quantitative analysis, aimed at evaluating the average density within a cluster , and to compare them with the inter-cluster density (i.e., the average density of tuples and attributes outside of their cluster of membership).

In addition, for each clustering result we computed the contingency table m, in which each column represents a discovered cluster, and each row represents a true class. The term m_{ij} corresponds to the number of tuples in D that were associated with cluster $\hat{x}_j$ and actually belongs to an ideal class C_i. Intuitively, each cluster $\hat{x}_j$ corresponds to the partition C_i that is best represented in $\hat{x}_j$ (i.e., such that m_{ij} is maximal).

For lack of space, in the following we only report the results on real-life datasets. The first dataset we analyze is the *SMART* collection from Cornell[1]. This collection consists of 3,891 documents organized into three main sub - collections: *Medline*, containing 1033 abstracts from medical journals; *Cisi*, containing 1460 abstracts from information retrieval papers; *Cranfield*, containing 1398 abstracts from aeronautical systems papers.

Through preprocessing, we obtained a series of datasets with increasing dimensionality, where a fixed dimensionality m was obtained by choosing the m most frequent terms. The corresponding dataset was then obtained by representing each document as a binary vector.

	$\hat{x}_1$	$\hat{x}_2$	$\hat{x}_3$
MED	1010	3	20
CISI	0	0	1460
CRAN	4	1383	11

(a) Contingency table

	$\hat{x}_1$	$\hat{x}_2$	$\hat{x}_3$
$\hat{y}_1$	$9.81 * 10^{-2}$	$3.59 * 10^{-2}$	$3.53 * 10^{-2}$
$\hat{y}_2$	$3.10 * 10^{-2}$	$1.24 * 10^{-1}$	$3.23 * 10^{-2}$
$\hat{y}_3$	$2.55 * 10^{-2}$	$3.00 * 10^{-2}$	$1.06 * 10^{-1}$

(b) Density matrix

(c) Incidence matrix

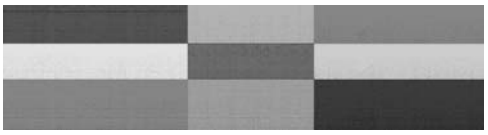
Fig. 1. Results for the *SMART* collection (m=500).

	$\hat{x}_1$	$\hat{x}_2$	$\hat{x}_3$
MED	18	1015	0
CISI	1460	0	0
CRAN	2	10	1386

(a) Contingency table

	$\hat{x}_1$	$\hat{x}_2$	$\hat{x}_3$
$\hat{y}_1$	$7.12 * 10^{-2}$	$1.92 * 10^{-2}$	$2.11 * 10^{-2}$
$\hat{y}_2$	$1.49 * 10^{-2}$	$5.75 * 10^{-2}$	$1.50 * 10^{-2}$
$\hat{y}_3$	$2.18 * 10^{-2}$	$2.08 * 10^{-2}$	$8.58 * 10^{-2}$

(b) Density matrix



(c) Incidence matrix

Fig. 2. Results for the *SMART* collection (m=1K).

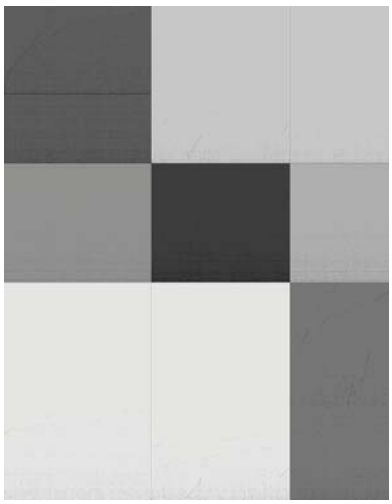
[1] ftp://ftp.cs.cornell.edu/pub/smart

	$\hat{x}_1$	$\hat{x}_2$	$\hat{x}_3$
MED	15	2	1016
CISI	1460	0	0
CRAN	7	1391	0

(a) Contingency table

	$\hat{x}_1$	$\hat{x}_2$	$\hat{x}_3$
$\hat{y}_1$	$1.96 * 10^{-2}$	$4.79 * 10^{-3}$	$4.65 * 10^{-3}$
$\hat{y}_2$	$8.11 * 10^{-3}$	$3.39 * 10^{-2}$	$7.80 * 10^{-3}$
$\hat{y}_3$	$2.68 * 10^{-3}$	$2.54 * 10^{-3}$	$1.41 * 10^{-2}$

(b) Density matrix



(c) Incidence matrix

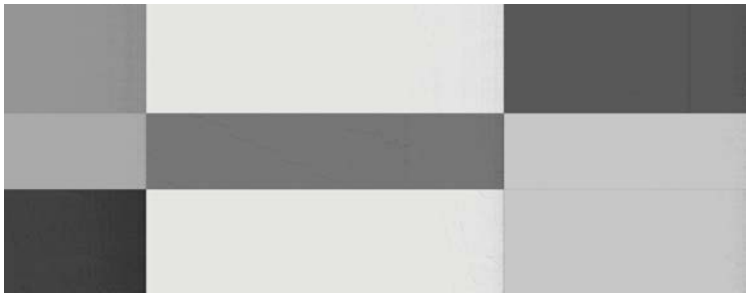
Fig. 3. Results for the *SMART* collection (m=5k).

	$\hat{x}_1$	$\hat{x}_2$	$\hat{x}_3$
MED	4	1018	11
CISI	0	2	1458
CRAN	1387	0	11

(a) Contingency table

	$\hat{x}_1$	$\hat{x}_2$	$\hat{x}_3$
$\hat{y}_1$	$2.16 * 10^{-2}$	$4.86 * 10^{-3}$	$5.04 * 10^{-3}$
$\hat{y}_2$	$1.39 * 10^{-3}$	$7.46 * 10^{-3}$	$1.44 * 10^{-3}$
$\hat{y}_3$	$2.10 * 10^{-3}$	$2.04 * 10^{-3}$	$9.98 * 10^{-3}$

(b) Density matrix



(c) Incidence matrix (inverted)

Fig. 4. Results for the *SMART* collection (m=10K).

Figures 1, 2, 3 and 4 show the clustering results for increasing values of m. As we can see, compactness and separability are quite good, as also testified

by the density matrices and contingency tables. Also, the proposed approach is effective to a large dimensionality in the number of attributes. In particular, results are quite more robust than those obtained with the ITCC co-clustering algorithm [7] (an example is reported in figures 5). The latter, indeed, allow high quality results only by fixing a high number of clusters in the Y dimension.

	$\hat{x}_1$	$\hat{x}_2$	$\hat{x}_3$
MED	980	2	51
CISI	0	0	1460
CRAN	1	1390	7

(a) Contingency table

	$\hat{x}_1$	$\hat{x}_2$	$\hat{x}_3$
$\hat{y}_1$	$6.97 * 10^{-4}$	$1.50 * 10^{-2}$	$2.52 * 10^{-4}$
$\hat{y}_2$	$1.30 * 10^{-2}$	$4.21 * 10^{-2}$	$1.26 * 10^{-2}$
$\hat{y}_3$	$5.36 * 10^{-3}$	$2.28 * 10^{-5}$	$6.10 * 10^{-5}$
$\hat{y}_4$	$1.48 * 10^{-2}$	$9.89 * 10^{-3}$	$3.03 * 10^{-3}$
$\hat{y}_5$	$1.06 * 10^{-2}$	$1.74 * 10^{-2}$	$2.03 * 10^{-2}$
$\hat{y}_6$	$1.33 * 10^{-2}$	$2.74 * 10^{-3}$	$6.10 * 10^{-3}$
$\hat{y}_7$	$6.72 * 10^{-5}$	$3.07 * 10^{-5}$	$6.02 * 10^{-3}$
$\hat{y}_8$	$5.05 * 10^{-3}$	$3.09 * 10^{-3}$	$1.71 * 10^{-2}$

(b) Density matrix

(c) Incidence matrix (inverted)

Fig. 5. ITCC Results for the *SMART* collection (m=10K).

A further dataset we analyse is *Internet Ads.*, available from the UCI Machine Learning repository[2]. The dataset contains 3,279 records and 1,554 boolean attributes. In addition, three further attributes are "categorical" in nature (although they are numeric, several values occur frequently). To summarize, the total number of possible items is 2832. This dataset represents a set of possible advertisements on Internet pages; each record represents a web page, and the features encode phrases occurring in the URL, the image's URL and alt text, the anchor text, and words occurring near the anchor text. Each record is labelled either as '*ad*' or as '*noad*'. The dataset is quite unbalanced, since there are 2,821 '*noad*s' and 458 '*ad*s'. Notwithstanding, separability is quite good, as it can be seen from figure 6. In particular, notice how clusters $\hat{x}_7$ and $\hat{x}_8$ represent the minority class.

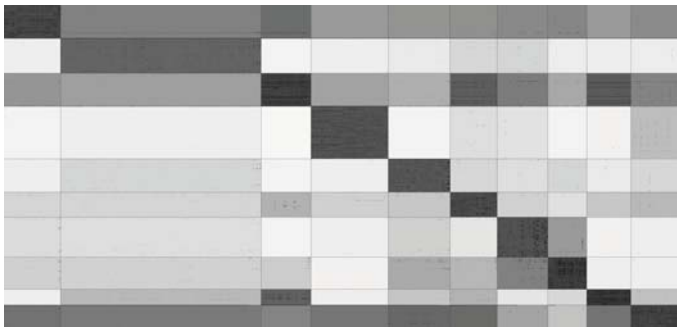
[2] http://www.ics.uci.edu/ mlearn/MLRepository.html

	$\hat{x}_1$	$\hat{x}_2$	$\hat{x}_3$	$\hat{x}_4$	$\hat{x}_5$	$\hat{x}_6$	$\hat{x}_7$	$\hat{x}_8$	$\hat{x}_9$	$\hat{x}_{10}$
ad	13	59	17	3	54	8	147	151	6	1
noad	262	917	225	372	247	220	99	37	208	233

(a) Contingency table

	$\hat{x}_1$	$\hat{x}_2$	$\hat{x}_3$	$\hat{x}_4$	$\hat{x}_5$	$\hat{x}_6$	$\hat{x}_7$	$\hat{x}_8$	$\hat{x}_9$	$\hat{x}_{10}$
$\hat{y}_1$	$8.74 * 10^{-2}$	$9.02 * 10^{-3}$	$1.11 * 10^{-2}$	$6.22 * 10^{-3}$	$6.72 * 10^{-3}$	$6.94 * 10^{-3}$	$7.37 * 10^{-3}$	$8.82 * 10^{-3}$	$6.29 * 10^{-3}$	$7.14 * 10^{-3}$
$\hat{y}_1$	$3.87 * 10^{-4}$	$1.48 * 10^{-2}$	$1.22 * 10^{-4}$	$3.15 * 10^{-4}$	$7.27 * 10^{-4}$	$1.34 * 10^{-3}$	$1.44 * 10^{-3}$	$4.09 * 10^{-4}$	$3.31 * 10^{-4}$	$4.80 * 10^{-4}$
$\hat{y}_1$	$6.43 * 10^{-3}$	$5.86 * 10^{-3}$	$1.19 * 10^{-1}$	$5.71 * 10^{-3}$	$5.17 * 10^{-3}$	$2.67 * 10^{-2}$	$9.24 * 10^{-3}$	$5.45 * 10^{-3}$	$4.10 * 10^{-2}$	$8.97 * 10^{-3}$
$\hat{y}_1$	$1.85 * 10^{-4}$	$3.73 * 10^{-4}$	$4.86 * 10^{-5}$	$5.34 * 10^{-2}$	$1.43 * 10^{-4}$	$1.16 * 10^{-3}$	$9.08 * 10^{-4}$	$1.66 * 10^{-4}$	$2.01 * 10^{-4}$	$3.01 * 10^{-3}$
$\hat{y}_1$	$3.20 * 10^{-4}$	$1.47 * 10^{-3}$	$1.29 * 10^{-4}$	$3.35 * 10^{-4}$	$4.14 * 10^{-2}$	$1.37 * 10^{-3}$	$1.12 * 10^{-3}$	$1.43 * 10^{-3}$	$7.34 * 10^{-4}$	$1.37 * 10^{-3}$
$\hat{y}_1$	$1.41 * 10^{-3}$	$1.47 * 10^{-3}$	$4.98 * 10^{-3}$	$1.74 * 10^{-3}$	$1.97 * 10^{-3}$	$6.80 * 10^{-2}$	$1.51 * 10^{-3}$	$1.09 * 10^{-3}$	$1.93 * 10^{-3}$	$3.39 * 10^{-3}$
$\hat{y}_1$	$1.23 * 10^{-3}$	$1.13 * 10^{-3}$	$1.27 * 10^{-4}$	$3.14 * 10^{-4}$	$1.72 * 10^{-3}$	$8.77 * 10^{-4}$	$5.60 * 10^{-2}$	$6.00 * 10^{-3}$	$2.63 * 10^{-4}$	$3.06 * 10^{-4}$
$\hat{y}_1$	$1.78 * 10^{-3}$	$1.57 * 10^{-3}$	$1.81 * 10^{-3}$	$2.23 * 10^{-4}$	$5.37 * 10^{-3}$	$3.76 * 10^{-3}$	$9.04 * 10^{-3}$	$1.22 * 10^{-1}$	$3.61 * 10^{-4}$	$3.03 * 10^{-4}$
$\hat{y}_1$	$3.34 * 10^{-4}$	$3.39 * 10^{-3}$	$2.54 * 10^{-2}$	$3.50 * 10^{-4}$	$2.09 * 10^{-3}$	$4.55 * 10^{-3}$	$8.55 * 10^{-4}$	$1.32 * 10^{-3}$	$1.51 * 10^{-1}$	$2.98 * 10^{-3}$
$\hat{y}_1$	$1.01 * 10^{-2}$	$9.36 * 10^{-3}$	$8.92 * 10^{-3}$	$9.76 * 10^{-3}$	$1.40 * 10^{-2}$	$1.19 * 10^{-2}$	$5.59 * 10^{-3}$	$2.30 * 10^{-3}$	$9.87 * 10^{-3}$	$1.09 * 10^{-1}$

(b) Density Matrix



(c) Incidence matrix

Fig. 6. Results for the *Internet ads* dataset.

5 Conclusions and Future Works

In this paper, we defined a novel EM-based approach to the discovery of co-clusters in a high-dimensional setting. This exploits the joint probability distribution over row and column variables associated to the data co-occurrence matrix, in order to initially find row clusters. Then, the conditional distribution of attributes over tuples is exploited to discover actual co-clusters, i.e. for associating concept (i.e. column) clusters with row clusters. We studied the behavior of our algorithm and compared it against the performance of a well-known *ad hoc* co-clustering scheme. The empirical results of a preliminary evaluation show the effectiveness of our approach and, apparently, suggest that natural co-clusters can still be discovered by tuning a mono-dimensional clustering strategy.

Still, the proposed approach is based on a naive assumption that tuple clusters are associated with exactly a concept cluster. Although this assumption seems to work well in practice, it appears nevertheless a strong requirement, which is hence likely to miss some latent sub-concepts actually holding in the data. As a future development, we plan to investigate the extension of the proposed framework in order to enable multiple characterizations of a same tuple cluster, in terms of corresponding associations with as many concept clusters.

References

1. R. Agrawal, J. Gehrke, D. Gunopulos, and P. Raghavan. Automatic subspace clustering of high dimensional data for data mining applications. In *Proc. ACM SIGMOD'98 Conf.*, pages 94 – 105, 1998.
2. D. Barbará, J. Couto, and Y. Li. COOLCAT: an entropy-based algorithm for categorical clustering. In *Proc. ACM Conf. on Information and Knowledge Management (CIKM'02)*, pages 582–589, 2002.
3. A. Califano, G. Stolovitzky, and Y. Tu. Analysis of gene expression microarrays for phenotype classification. In *Proc. of the 8th Int. Conf. on Intelligent Systems for Molecular Biology*, pages 75–85, 2000.
4. Y. Cheng and G. M. Church. Biclustering of expression data. In *Proc. of the 8th Int. Conf. on Intelligent Systems for Molecular Biology*, pages 93–103, 2000.
5. G. Costa, F. Folino, G. Manco and R. Ortale. A Hierarchical Probabilistic Model for Co-Clustering High-Dimensional Data. Technical Report n.4, ICAR-CNR, 2006. Available at http://biblio.cs.icar.cnr.it/biblio/.
6. S.C. Deerwester et al. Indexing by latent semantic analysis. *Journal of American Society of Information Science*, 41(6), 1990.
7. I. S. Dhillon, S. Mallela, and D. S. Modha. Information-theoretic co-clustering. In *Proc. of the 9th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining (KDD)*, pages 89–98, 2003.
8. G. Getz, E. Levine, and E. Domany. Coupled two-way clustering analysis of gene microarray data. In *Proc. of the Natural Academy of Sciences USA*, pages 12079–12084, 2000.
9. J. A. Hartigan. Direct clustering of a data matrix. *Journal of the American Statistical Association (JASA)*, 67(337):123–129, 1972.
10. L. Lazzeroni and A. Owen. Plaid model for gene expression data. technical report, 2000.
11. A. McCallum, K. Nigam, and L. H. Ungar. Efficient clustering of high-dimensional data sets with application to reference matching. In *Proc. 6th Int. Conf. on Knowledge Discovery and Data Mining (KDD'00)*, pages 169 – 178, 2000.
12. G. McLachlan and D. Peel. *Finite Mixture Models*. Wiley, 2000.
13. L. Parsons, E. Haque, and H. Liu. Subspace Clustering for High Dimensional Data: A Review. *ACM SIGKDD Explorations*, 6(1):90 – 105, 2004.
14. S.K. Selim and M.A.Ismail. Biclustering algorithms for biological data analysis: A survey. *Transactions on Computational Biology and Bioinformatics (TCBB)*, 1(1):24 – 45, 2004.
15. A. Tanay, R. Sharan, and R. Shamir. Discovering statistically significant biclusters in gene expression data. *Bioinformatics*, 18(1):S136–S144, 2002.
16. C. Tang, Li. Zhang, I. Zhang, and M. Ramanathan. Interrelated two-way clustering: an unsupervised approach for gene expression data analysis. In *Proc. of the 2nd IEEE Int. Symposium on Bioinformatics and Bioengineering*, pages 41–48, 2001.
17. H. Wang, Wei Wang, J. Yang, and P. S. Yu. Clustering by pattern similarity in large data sets. In *Proc. of the 2002 ACM SIGMOD Int. Conf. on Management of Data*, pages 394–405, 2002.
18. G. Govaert, M. Nadif. An EM Algorithm for the Block Mixture Model. In *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27(4): 643 – 647, 2005.
19. A.P. Dempster, N.M. Laird, D.B. Rubin. Maximum Likelihood from Incomplete Data via EM Algorithm. In *J. Royal Statistical Society*, 39(B): 1 – 38, 1977.