Computer Science and Artificial Intelligence Laboratory
Technical Report

MIT-CSAIL-TR-2009-054 November 3, 2009

Co-Clustering with Generative Models

Danial Lashkari and Polina Golland

massachusetts institute of technology, cambridge, ma 02139 usa — www.csail.mit.edu

Co-Clustering with Generative Models

Danial Lashkari **Polina Golland**

CSAIL Technical Report
Massachusetts Institute of Technology
Cambridge, MA 02139

Abstract

In this paper, we present a generative model for co-clustering and develop algorithms based on the mean field approximation for the corresponding modeling problem. These algorithms can be viewed as generalizations of the traditional model-based clustering; they extend hard co-clustering algorithms such as Bregman co-clustering to include soft assignments. We show empirically that these model-based algorithms offer better performance than their hard-assignment counterparts, especially with increasing problem complexity.

1 Introduction

Co-clustering, or Bi-clustering, is the problem of simultaneously clustering rows and columns of a matrix of data points. The first such algorithm was introduced more than thirty years ago [1] but there has been a new interest in the problem due to recently emerging applications. Different variants of co-clustering problems have been posed in fields such as biological data analysis [2, 3, 4], data mining and information retrieval [5, 6, 7], studies of social networks, and computational cognitive science [8, 9]. These algorithms aim to discover distinct *types* of row and column indices based on their interaction via the data points. It has also been suggested in the case of high-dimensional data, if we apply co-clustering to simultaneously cluster both data points and features, we sometimes improve the regular clustering results on the data points [10].

In traditional clustering, generative models provide algorithms that can be understood as statistical generalizations of the k-means algorithm [11]. The basic k-means algorithm has a fully combinatorial structure which presents a challenge when searching in the space of solutions. In contrast, considering clustering as a probabilistic modeling problem, we can formulate it in a continuous space, enabling easier search for the optimal solution. In addition to the advantages of model-based clustering in searching the complex space of solutions, it also provides a way to express uncertainty in the results of clustering with its *soft* assignments. In this paper, we develop a general generative model for co-clustering which is similarly related to basic hard-assignment co-clustering algorithms such as Bregman co-clustering [10]. The simplicity of our model makes it possible to use mean field theory to derive co-clustering algorithms maintaining close connection to mixture-model clustering. In contrast, prior model-based methods, in the context of the more general problem of relational data clustering, take more complicated structures and rely on sampling algorithms [8, 9].

We consider a general framework where data points could be generated by any distribution. This approach unifies the treatment of the real-valued data encountered in many biological applications with that of positive-valued co-occurrence data. Co-occurrence data presents the frequency of a pair of two different types of objects occurring simultaneously, for instance, the counts indicating the number of times each user clicked on any item on a webpage. Several co-clustering algorithms have been developed specifically for this kind of data due to its frequent applications in data-mining [5, 6, 7].

The paper is organized as follows: In Section 2, we introduce our generative model, the method employed for solving the corresponding problem, and two algorithms developed with this model for

Gaussian and co-occurrence data. Section 3 explains the relationship between this generative model and the previous work on combinatorial (hard) co-clustering and mixture-model clustering. In Section 4, we present experimental results demonstrating performance of our algorithm in comparison with the existing hard co-clustering algorithms in the literature.

2 Generative Model for Co-Clustering

Let $\boldsymbol{z} = [z_{uv}]$ be a given set of $n \times m$ data points where all data points z_{uv} are elements of a set $\mathcal{Z}$. In general, our construction holds when $\mathcal{Z}$ is a vectorial set of any dimensions but in the examples discussed in the paper, it is a scalar set. Co-clustering explains the statistics of the data through a pair of functions

$$c : \{1, \cdots, n\} \rightarrow \{1, \cdots, k\}$$
$$g : \{1, \cdots, m\} \rightarrow \{1, \cdots, l\}$$

when k row and l column clusters are assumed. In a modeling framework, these functions are thought of as variables that contribute to the distribution of data points without being directly observed in the measurements. Therefore, we build a probabilistic model of the data on the product space of observed (data points) and hidden (co-clustering functions) variables.

2.1 Generative Model

We represent co-clustering functions in our model with $n \times k$ binary indicator variables $\boldsymbol{c} = [c_{u\tilde{u}}]$ and $m \times l$ indicator variables $\boldsymbol{g} = [g_{v\tilde{v}}]$ where the indices $\tilde{u}$ and $\tilde{v}$ denote row and column cluster labels, respectively. We assume that exactly one variable in each row of $\boldsymbol{c}$ and $\boldsymbol{g}$ could be 1 to guarantee that each point is mapped only to one cluster. With this constraint, any pair $(\boldsymbol{c}, \boldsymbol{g})$ is equivalent to a unique choice of co-clustering functions defined by $c(u) = \sum_{\tilde{u}} \tilde{u} c_{u\tilde{u}}$ and $g(v) = \sum_{\tilde{v}} \tilde{v} g_{v\tilde{v}}$. Our model describes the distribution of data points for a given configuration of hidden variables as

$$P(\boldsymbol{z}|\boldsymbol{c}, \boldsymbol{g}; \boldsymbol{\theta}) = \prod_{u,v} f(z_{uv}; \theta_{c(u)g(v)}) = \prod_{u,v} \left[\prod_{\tilde{u},\tilde{v}} \Big(f(z_{uv}; \theta_{\tilde{u}\tilde{v}}) \Big)^{c_{u\tilde{u}} g_{v\tilde{v}}} \right], \qquad (1)$$

where $f(\cdot; \theta)$ is a parameterized distribution on $\mathcal{Z}$ and $\boldsymbol{\theta} = [\theta_{\tilde{u}\tilde{v}}]$ is the set of $k \times l$ parameter vectors. Data points with the same row and column cluster $(\tilde{u}, \tilde{v})$ are assumed to be *i.i.d.* samples from a distribution $f(\cdot; \theta_{\tilde{u}\tilde{v}})$.

We also assume a product prior for $(\boldsymbol{c}, \boldsymbol{g})$ treating $c(u)$'s as independent samples from an identical multinomial distribution. Taking a similar assumption about the $g(v)$'s, we write the full generative model of the observed and hidden variables as

$$\begin{aligned} P(\boldsymbol{z}, \boldsymbol{c}, \boldsymbol{g}; \boldsymbol{\theta}, \boldsymbol{\pi}, \boldsymbol{\rho}) &= P(\boldsymbol{z}|\boldsymbol{c}, \boldsymbol{g}; \boldsymbol{\theta}) P(\boldsymbol{c}; \boldsymbol{\pi}) P(\boldsymbol{g}; \boldsymbol{\rho}) \\ &= \left[\prod_{\tilde{u},\tilde{v}} \prod_{u,v} \Big(f(z_{uv}; \theta_{\tilde{u}\tilde{v}}) \Big)^{c_{u\tilde{u}} g_{v\tilde{v}}} \right] \left[\prod_{u',\tilde{u}'} \pi_{\tilde{u}'}^{c_{u'\tilde{u}'}} \right] \left[\prod_{v',\tilde{v}'} \rho_{\tilde{v}'}^{g_{v'\tilde{v}'}} \right] \end{aligned} \qquad (2)$$

where $\boldsymbol{\pi}$ and $\boldsymbol{\rho}$ correspond to the parameters of the multinomial priors over the row and column cluster indices, respectively. With this model, we can formulate co-clustering as learning (parameter estimation) with the log-likelihood function

$$(\boldsymbol{\theta}^*, \boldsymbol{\pi}^*, \boldsymbol{\rho}^*) = \max_{\boldsymbol{\theta}, \boldsymbol{\pi}, \boldsymbol{\rho}} \log P(\boldsymbol{z}; \boldsymbol{\pi}, \boldsymbol{\rho}, \boldsymbol{\theta}) \qquad (3)$$

and inference over all possible co-clustering configurations, that is, the MAP problem with the posterior distribution $P(\boldsymbol{c}, \boldsymbol{g}|\boldsymbol{z}; \boldsymbol{\theta}^*, \boldsymbol{\pi}^*, \boldsymbol{\rho}^*)$. Using the EM algorithm for this problem, we alternate between inference (E-step) and parameter estimation (M-step). Therefore, computing the posterior is needed in every step of the algorithm.

The expression for the log-joint-probability of the full data $(\boldsymbol{z}, \boldsymbol{c}, \boldsymbol{g})$

$$\log P(\boldsymbol{z}, \boldsymbol{c}, \boldsymbol{g}; \boldsymbol{\theta}, \boldsymbol{\pi}, \boldsymbol{\rho}) = \sum_{u,v,\tilde{u},\tilde{v}} c_{u\tilde{u}} g_{v\tilde{v}} \log f(z_{uv}; \theta_{\tilde{u}\tilde{v}}) + \sum_{u,\tilde{u}} c_{u\tilde{u}} \log \pi_{\tilde{u}} + \sum_{v,\tilde{v}} g_{v\tilde{v}} \log \rho_{\tilde{v}} \qquad (4)$$

includes interaction between the binary variables $(\boldsymbol{c}, \boldsymbol{g})$ in the first term. This means that the posterior distribution over row and column assignment variables is not factorable and its computation is hard, involving a summation over the number of configurations exponential in n and m. Therefore, we need an approximate inference method to efficiently compute the posterior distribution.

2.2 Mean Field Approximation

Using the mean field method [11], we approximate the posterior $P(\boldsymbol{c}, \boldsymbol{g}|\boldsymbol{z})$ with a distribution $Q(\boldsymbol{c}, \boldsymbol{g})$ of the form

$$Q(\boldsymbol{c}, \boldsymbol{g}) = \left(\prod_{u,\tilde{u}} q(\tilde{u}|u)^{c_{u\tilde{u}}}\right) \times \left(\prod_{v,\tilde{v}} r(\tilde{v}|v)^{g_{v\tilde{v}}}\right), \tag{5}$$

which is the product of independent multinomial distributions on $c(u)$ and $g(v)$ with corresponding parameters $q(\tilde{u}|u)$ and $r(\tilde{v}|v)$. It is easy to see that $E_Q[c_{u\tilde{u}}] = q(\tilde{u}|u)$ and $E_Q[g_{v\tilde{v}}] = r(\tilde{v}|v)$ where $E_Q[\cdot]$ denotes expectation with respect to the distribution $Q(\boldsymbol{c}, \boldsymbol{g})$. We now aim to minimize the mean field free energy

$$\begin{aligned} F(Q, P) &= E_Q[\log Q(\boldsymbol{c}, \boldsymbol{g})] - E_Q[\log P(\boldsymbol{z}, \boldsymbol{c}, \boldsymbol{g})] && (6) \\ &= \sum_{u,\tilde{u}} q(\tilde{u}|u) \log q(\tilde{u}|u) + \sum_{v,\tilde{v}} r(\tilde{v}|v) \log r(\tilde{v}|v) \\ &\quad - \sum_{u,v,\tilde{u},\tilde{v}} q(\tilde{u}|u) r(\tilde{v}|v) \log f(z_{uv}; \theta_{\tilde{u}\tilde{v}}) - \sum_{u,\tilde{u}} q(\tilde{u}|u) \log \pi_{\tilde{u}} - \sum_{v,\tilde{v}} r(\tilde{v}|v) \log \rho_{\tilde{v}} \ , && (7) \end{aligned}$$

where P and Q are functions of the variables $(\boldsymbol{\theta}, \boldsymbol{\pi}, \boldsymbol{\rho})$ and $(\boldsymbol{q}, \boldsymbol{r})$, respectively. We apply alternate minimization with respect to the posterior parameters $(\boldsymbol{q}, \boldsymbol{r})$ and the original parameters $(\boldsymbol{\theta}, \boldsymbol{\pi}, \boldsymbol{\rho})$, constructing a variational EM algorithm for solving problem (3).

Since the terms involving $\boldsymbol{\pi}$, $\boldsymbol{\rho}$, and $\boldsymbol{\theta}$ appear separately in (7), updating the parameters in the M-step is straightforward, especially if the model f is cleverly chosen such that the objective function $F(Q, P)$ is a convex function of $\boldsymbol{\theta}$. In the E-step, for a given set of parameters, F is a convex function of $(\boldsymbol{q}, \boldsymbol{r})$ on the simplices where they are defined. The minimization problem over variables $\boldsymbol{q}$ or $\boldsymbol{r}$, when considered separately, is convex; therefore, we can keep one set constant and minimize F with respect to another. By repeating this process once for each set, we find Q which minimizes the cost function. In practice, we chose to alternately update $\boldsymbol{q}$ and $\boldsymbol{r}$ in different E-steps as it gave slightly better results. In both cases, it is clear that the algorithm converges to a local minimum of the approximate cost function (7).

2.3 Example: Gaussian Model

Let us consider the case where data points are real-valued and assume $f(z; \mu, \sigma)$ in (1) to be a Gaussian distribution $(2\pi\sigma^2)^{-\frac{1}{2}} e^{-(z-\mu)^2/2\sigma^2}$. Substituting this expression into (7), taking into account the normalization constraints over the variables, and minimizing the resulting expression with respect to the parameters $(\boldsymbol{q}, \boldsymbol{r})$, we find the update rules in the E-step as

$$q^{(t+1)}(\tilde{u}|u) \propto \pi_{\tilde{u}}^{(t)} \exp\left[-\frac{1}{2} \sum_{v,\tilde{v}} r^{(t)}(\tilde{v}|v) \left(\frac{z_{uv} - \mu_{\tilde{u}\tilde{v}}^{(t)}}{\sigma_{\tilde{u}\tilde{v}}^{(t)}}\right)^2\right] \tag{8}$$

$$r^{(t+1)}(\tilde{v}|v) \propto \rho_{\tilde{v}}^{(t)} \exp\left[-\frac{1}{2} \sum_{u,\tilde{u}} q^{(t)}(\tilde{u}|u) \left(\frac{z_{uv} - \mu_{\tilde{u}\tilde{v}}^{(t)}}{\sigma_{\tilde{u}\tilde{v}}^{(t)}}\right)^2\right], \tag{9}$$

where the superscript (t) indicates the values correspond to the t-th step of the iterations and all variables are further normalized to give valid multinomial parameters. For the M-step, using a similar treatment for the model parameters yields

$$\pi_{\tilde{u}}^{(t)} = \frac{1}{n} \sum_u q^{(t)}(\tilde{u}|u) \qquad \rho_{\tilde{v}}^{(t)} = \frac{1}{m} \sum_v r^{(t)}(\tilde{v}|v) \tag{10}$$

$$\mu_{\tilde{u}\tilde{v}}^{(t)} = \frac{\sum_{u,v} q^{(t)}(\tilde{u}|u) r^{(t)}(\tilde{v}|v) z_{uv}}{nm\pi_{\tilde{u}}^{(t)} \rho_{\tilde{v}}^{(t)}} \qquad \sigma_{\tilde{u}\tilde{v}}^{(t)} = \left[\frac{\sum_{u,v} q^{(t)}(\tilde{u}|u) r^{(t)}(\tilde{v}|v) (z_{uv} - \mu_{\tilde{u}\tilde{v}}^{(t)})^2}{nm\pi_{\tilde{u}}^{(t)} \rho_{\tilde{v}}^{(t)}}\right]^{\frac{1}{2}} . \tag{11}$$

Since parameter estimation for the Gaussian model is exact, this algorithm has guaranteed convergence as explained in Section 2.2.

2.4 Marginal-preserving Likelihood for Co-occurrence Data

We can use a Poisson distribution instead of Gaussian to model co-occurrence data matrices. In general, likelihood models of the form (1) assume the same distribution for the block of all data points assigned the same row and column cluster indices, i.e., the set of $\{z_{uv}, c(u) = \tilde{u}, g(v) = \tilde{v}\}$ for some pair $(\tilde{u}, \tilde{v})$. Accordingly, they assume the same frequency of occurrence for all row or column indices within a cluster. If we want the model to allow distinct frequencies of occurrence for different indices, we have to suitably modify the form of $P(\boldsymbol{z}|\boldsymbol{c}, \boldsymbol{g})$. Here, we present such a model as an example of possible extensions to our basic framework.

We assume that the data is represented by S co-occurrence indicator matrices, i.e., *i.i.d.* binary sample matrices $\boldsymbol{z}^s = [z^s_{uv}]$, $1 \leq s \leq S$, such that in each sample exactly one element is equal to 1 indicating co-ocurrence between the corresponding row and column. The frequency representation of the data can be written in terms of these co-occurrence observations as the average $\bar{\boldsymbol{z}} = \frac{1}{S}\sum_s \boldsymbol{z}^s$. Here, we write the model for one co-occurrence sample $[z^s_{uv}]$ and omit sample superscripts to make the expressions easier to follow.

With co-clustering functions $(\boldsymbol{c}, \boldsymbol{g})$, we introduce a set of auxiliary variables

$$x_u = \sum_v z_{uv}, \quad y_v = \sum_u z_{uv}, \quad \tilde{z}_{\tilde{u}\tilde{v}} = \sum_{u,v} c_{u\tilde{u}} g_{v\tilde{v}} z_{uv}, \quad \tilde{x}_{\tilde{u}} = \sum_{\tilde{v}} \tilde{z}_{\tilde{u}\tilde{v}}, \quad \tilde{y}_{\tilde{v}} = \sum_{\tilde{u}} \tilde{z}_{\tilde{u}\tilde{v}}. \tag{12}$$

Here, $\boldsymbol{x}$ is an indicator vector for the row-index occurrence, that is, $x_u = 1$ implies that one of the elements in the u-th row of $\boldsymbol{z}$ is equal to 1, and $\tilde{\boldsymbol{z}}$ is an indicator matrix representing which block contains the observed 1. Similarly, $\boldsymbol{y}$, $\tilde{\boldsymbol{x}}$, and $\tilde{\boldsymbol{y}}$ are indicator variables for occurrences of column, row-cluster, and column-cluster indices. Note that all these variables are deterministic functions of $\boldsymbol{z}$ given $(\boldsymbol{c}, \boldsymbol{g})$. Furthermore, $\boldsymbol{z}$ and $\tilde{\boldsymbol{z}}$ could be equivalently represented by $(\boldsymbol{x}, \boldsymbol{y})$ and $(\tilde{\boldsymbol{x}}, \tilde{\boldsymbol{y}})$, respectively, and $\tilde{\boldsymbol{x}}$ and $\tilde{\boldsymbol{y}}$ are deterministic functions of $\boldsymbol{x}$ and $\boldsymbol{y}$. Using these relationships, we can show that

$$P(\boldsymbol{z}|\boldsymbol{c}, \boldsymbol{g}) = P(\boldsymbol{z}, \boldsymbol{x}, \boldsymbol{y}, \tilde{\boldsymbol{z}}, \tilde{\boldsymbol{x}}, \tilde{\boldsymbol{y}}|\boldsymbol{c}, \boldsymbol{g}) = P(\boldsymbol{x}|\tilde{\boldsymbol{x}}, \boldsymbol{c}, \boldsymbol{g}) P(\boldsymbol{y}|\tilde{\boldsymbol{y}}, \boldsymbol{c}, \boldsymbol{g}) P(\tilde{\boldsymbol{z}}|\boldsymbol{c}, \boldsymbol{g}) \tag{13}$$

$$= \frac{P(\boldsymbol{x}|\boldsymbol{c}, \boldsymbol{g})}{P(\tilde{\boldsymbol{x}}|\boldsymbol{c}, \boldsymbol{g})} \times \frac{P(\boldsymbol{y}|\boldsymbol{c}, \boldsymbol{g})}{P(\tilde{\boldsymbol{y}}|\boldsymbol{c}, \boldsymbol{g})} \times P(\tilde{\boldsymbol{z}}|\boldsymbol{c}, \boldsymbol{g}).,\tag{14}$$

To define the model, we assume multinomial distributions parameterized by η_u and γ_v, for instance, $P(x_u = 1|\boldsymbol{c}, \boldsymbol{g}; \boldsymbol{\eta}) = \eta_u$. If we further model the block variables with multinomial distributions parameterized with $\mu_{\tilde{u}\tilde{v}}$, $\kappa_{\tilde{u}}$, and $\nu_{\tilde{v}}$, for instance taking $P(\tilde{z}_{\tilde{u}\tilde{v}} = 1|\boldsymbol{c}, \boldsymbol{g}; \boldsymbol{\mu}) = \mu_{\tilde{u}\tilde{v}}$, then the model can be completely parameterized as

$$P(\boldsymbol{z}|\boldsymbol{c}, \boldsymbol{g}) = \prod_{u,v} \prod_{\tilde{u},\tilde{v}} \left[\left(\frac{\eta_u}{\kappa_{\tilde{u}}} \right) \left(\frac{\gamma_v}{\nu_{\tilde{v}}} \right) \mu_{\tilde{u}\tilde{v}} \right]^{c_{u\tilde{u}} g_{v\tilde{v}} z_{uv}}, \tag{15}$$

where we have expressed all the variables again in terms of z_{uv}. This form guarantees that the model preserves the marginal row and column distributions and the co-clustering structure determines the *correlation* between row and column indices instead of their joint distribution. In order to have this model normalized, we need to ensure that the parameters satisfy the constraints $\kappa_{\tilde{u}} = \sum_u c_{u\tilde{u}} \eta_u$ and $\nu_{\tilde{v}} = \sum_v g_{v\tilde{v}} \gamma_v$. These constraints interact with the hidden variables and make the parameter estimation hard.

Let us for the moment ignore the constraints on parameters and continue with solving the corresponding approximate problem. Substituting this model in (6), and recalling that the actual frequency data $\bar{\boldsymbol{z}}$ is the average of independent samples $\boldsymbol{z}^s$, we find that the optimal values for the parameters $\boldsymbol{\eta}$ and $\boldsymbol{\gamma}$ are simply $\eta_u = \bar{x}_u$ and $\gamma_v = \bar{y}_v$, the data row and column marginals. The M-step update rules for the prior parameters $\boldsymbol{\pi}$ and $\boldsymbol{\rho}$ are the same as (10), while the model parameters are updated as $\mu_{\tilde{u}\tilde{v}} = \sum_{u,v} q(\tilde{u}|u) r(\tilde{v}|v) \bar{z}_{uv}$, $\kappa_{\tilde{u}} = \sum_u q(\tilde{u}|u)\eta_u$ and $\nu_{\tilde{v}} = \sum_v r(\tilde{v}|v)\gamma_v$. Note that the last two equations imply that, although we have ignored the orginial parameter constraints, the solutions satisfy them in the expectation sense.

Using these equalities, we arrive at the E-step updates

$$q^{(t+1)}(\tilde{u}|u) \propto \pi^{(t)}_{\tilde{u}} \exp\left[S \sum_{v,\tilde{v}} \bar{z}_{uv} r^{(t)}(\tilde{v}|v) \log\left(\frac{\mu^{(t)}_{\tilde{u}\tilde{v}}}{\kappa^{(t)}_{\tilde{u}} \nu^{(t)}_{\tilde{v}}}\right)\right] \tag{16}$$

$$r^{(t+1)}(\tilde{v}|v) \propto \rho^{(t)}_{\tilde{v}} \exp\left[S \sum_{u,\tilde{u}} \bar{z}_{uv} q^{(t)}(\tilde{u}|u) \log\left(\frac{\mu^{(t)}_{\tilde{u}\tilde{v}}}{\kappa^{(t)}_{\tilde{u}} \nu^{(t)}_{\tilde{v}}}\right)\right]. \tag{17}$$

These equations could be further simplified but we leave them in this form which is easier to implement numerically.

3 Relationship to Bregman Co-clustering and Mixture-Model Clustering

In this section, we show that our work and a class of hard co-clustering algorithms called Bregman Co-clustering have common modeling assumptions [10]. In particular, the two algorithms derived in Sections 2.3 and 2.4 find MAP solutions while Block Average Co-clustering (BAC) [10] and the Information Theoretic Co-clustering algorithm (ITC) solve the ML problems for the same models [7]. Making the connection to the mixture-modeling, we further show how this corresponds to soft vs. hard clustering assignments.

3.1 Bregman Co-clustering and ML Problem

We first briefly review the basic ideas of Bregman co-clustering [10]. Bregman co-clustering assumes random variables $U \in \{1, \cdots, n\}$ and $V \in \{1, \cdots, m\}$ represent the row and column indices, respectively. The distribution ν of the product random variable $U \times V$ associates a set of weights to different data points. In most common cases, since there is no preference for any elements of the data matrix, $\nu(u, v) = \frac{1}{nm}$ for all u and v. We can also think of the data $\boldsymbol{z}$ as a function $z : \mathcal{U} \times \mathcal{V} \rightarrow \mathcal{Z}$ and define a random variable Z over its range, with a corresponding distribution induced by ν, namely, $P(Z = z_{uv}) = \nu(u, v)$.

Given a pair of co-clustering functions c and g, we define random variables $\tilde{U} \in \{1, \cdots, k\}$ and $\tilde{V} \in \{1, \cdots, l\}$ such that $\tilde{U} = c(U)$ and $\tilde{V} = g(V)$. Then the algorithm constructs a matrix approximation function $\tilde{z} : \mathcal{U} \times \mathcal{V} \rightarrow \mathcal{Z}$ and a corresponding random variable $\tilde{Z}$ with the constraint that certain summary statistics of the co-clustering are shared between Z and $\tilde{Z}$. For instance, BAC imposes $E[Z|\tilde{u}, \tilde{v}] = E[\tilde{Z}|\tilde{u}, \tilde{v}]$ for all $\tilde{u}$ and $\tilde{v}$. With any such assumption about the form of $\tilde{z}$, the goal of co-clustering is to find the pair of co-clustering functions (c, g) and the approximation function $\tilde{z}$ such that the average distortion $\mathbb{E}_\nu d_\phi(Z, \tilde{Z})$ is minimized, where the distortion $d_\phi(\cdot, \cdot)$ is a *Bregman divergence* [12].

Bregman divergences are used in many machine learning applications, including clustering. Common examples include Euclidean distance for real-valued vectors, and KL-divergence for probability vectors. There is an equivalence relation between regular exponential families and Bregman divergences, namely, any regular exponential family $f(z; \theta) = e^{z\theta - \psi(\theta)}$ can be represented as $f(z; \mu) = C(z)e^{-d_\phi(z,\mu)}$ where the new parameter μ is the expected value of Z under $f(z; \theta)$ [12]. We establish the connection between our modeling approach and the Bregman matrix approximation using this equivalence. Assuming the above form for our model in (1), we compute the maximum likelihood configuration of the hidden variables.

$$\begin{aligned}(\boldsymbol{c}^*, \boldsymbol{g}^*) &= \underset{\boldsymbol{c},\boldsymbol{g}}{\operatorname{argmax}} \; \max_{\boldsymbol{\mu}} \; \log P(\boldsymbol{z}|\boldsymbol{c}, \boldsymbol{g}; \boldsymbol{\mu}) = \underset{\boldsymbol{c},\boldsymbol{g}}{\operatorname{argmax}} \; \max_{\boldsymbol{\mu}} \sum_{u,v,\tilde{u},\tilde{v}} -c_{u\tilde{u}} g_{v\tilde{v}} d_\phi(z_{uv}, \mu_{\tilde{u}\tilde{v}}) + \text{const.} \\ &= \underset{\boldsymbol{c},\boldsymbol{g}}{\operatorname{argmin}} \; \min_{\tilde{Z}_{\text{BAC}}} \; E_\nu d_\phi(Z, \tilde{Z}),\end{aligned} \tag{18}$$

where ν is uniform and the last equality follows because it can be shown that $E[Z] = \operatorname{argmin}_\mu E d_\phi(Z, \mu)$ and that, in the case of BAC, $E[Z|\tilde{U}, \tilde{V}] = \operatorname{argmin}_{\tilde{Z}} \; E_\nu d_\phi(Z, \tilde{Z})$ [10]. Since expression (18) is the BAC cost function, we conclude that the ML solution in our generative model is equivalent to the BAC solution. Minimizing this cost function is a hard combinatorial problem. The algorithm proposed in [10] for finding its solution can be viewed as a generalization of the k-means algorithm to co-clustering.

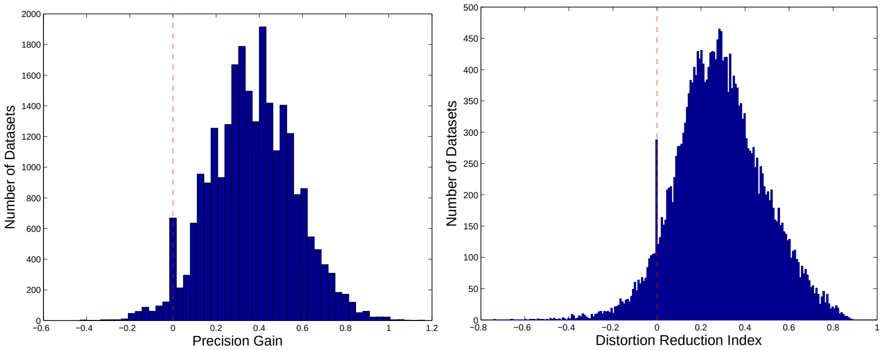

Figure 1: Comparison between the results of our Gaussian model-based co-clustering and BAC on 25,000 randomly generated data sets. Left: difference in precision values found by the two algorithms. Right: distortion reduction index. See text for details.

Information Theoretic Co-clustering (ITC), another example of Bregman co-clustering, is an algorithm for data sets representing the co-occurence of the row and column indices [7]. ITC requires the approximation variable $\tilde{Z}$ to match block, row and column statistics of the data set, i.e., $E[Z|u] = E[\tilde{Z}|u]$ and $E[Z|v] = E[\tilde{Z}|v]$ for all u and v. It also assumes normalized co-occurence data $\bar{\boldsymbol{z}}$, where all the data points sum to 1. Using the I-divergence $d_\phi(z,\mu) = z\log(z/\mu) - (z-\mu)$ as the distortion function, it can be shown that the optimal approximation $\tilde{Z}$ satisfies [10]

$$\tilde{Z}(U,V) = \frac{\mathbb{E}[Z|\tilde{U},\tilde{V}] \times \mathbb{E}[Z|U] \times \mathbb{E}[Z|V]}{\mathbb{E}[Z|\tilde{U}] \times \mathbb{E}[Z|\tilde{V}]}. \tag{19}$$

With these assumptions, the ITC cost function can be written as the co-clustering *Information Loss* $\Delta I = I(U;V) - I(\tilde{U};\tilde{V})$ [7]. If we evaluate our likelihood function (15) for the co-occurrence model in Section 2.4, we find that the ML solution of our model also minimizes the information loss:

$$(\boldsymbol{c}^*,\boldsymbol{g}^*) = \operatorname*{argmax}_{\boldsymbol{c},\boldsymbol{g}} \max_{\boldsymbol{\mu}} \sum_{u,v,\tilde{u},\tilde{v}} \bar{z}_{uv} c_{u\tilde{u}} g_{v\tilde{v}} \log\frac{\mu_{\tilde{u}\tilde{v}}\bar{x}_u\bar{y}_v}{\kappa_{\tilde{u}}\nu_{\tilde{v}}} = \operatorname*{argmin}_{\boldsymbol{c},\boldsymbol{g}} I(U;V) - I(\tilde{U};\tilde{V}) + \text{const.}$$

3.2 Mixture-modeling and Soft Clustering Assignments

If we replace all double indices (u,v) with a single index u in (18) and only consider the clustering function $c(\cdot)$, in the case of Euclidean distance, BAC turns into the k-means algorithm, while our generative model leads to the EM-based mixture modeling. The prior on the hidden variables, which we introduced in Section 2.1, corresponds to the mixture weights of the generative model clustering. Similar to the way soft cluster assignments are described as the posterior distribution over the hidden clustering variables in one dimensional model-based clustering, our final posterior variables $q^*(\tilde{u}|u)$ and $r^*(\tilde{v}|v)$ represent co-clustering soft-assignments. In the case of clustering, given the parameters, for instance cluster-means, cluster assignments of different data points are independent and easy to compute. In contrast, as we saw in Section 2.1, row and column cluster assignments become interdependent in co-clustering. Row assignments c become independent only given column assignments g and vice versa. This also means that the initialization of these co-clustering algorithms must be done in the space of assignments and not model parameters. The mean field approximation leads to a that solution has an immediate interpretation as soft-assignments: setting $c^*(u) = \operatorname{argmax}_{\tilde{u}} q^*(\tilde{u}|u)$ and $g^*(v) = \operatorname{argmax}_{\tilde{v}} r^*(\tilde{v}|v)$ gives an approximate MAP estimate for the hidden variables.

If we assume a distribution $\nu'(u,v,\tilde{u},\tilde{v}) = \frac{1}{nm}q(\tilde{u}|u)r(\tilde{v}|v)$ on the indices, and substitute the optimal values of π and ρ from (10), the free energy cost function (7) could be written as

$$\tfrac{1}{nm}F(Q,P) = \tfrac{1}{m}I(U;\tilde{U}) + \tfrac{1}{n}I(V;\tilde{V}) + \mathbb{E}d_\phi(z(U,V),\tilde{z}(\tilde{U},\tilde{V})) \tag{20}$$

The standard model-based clustering can be thought of as a probabilistic compression problem in terms of $q(\tilde{u}|u)$ with the cost function $I(U;\tilde{U}) + \mathbb{E}d_\phi(z(U),\tilde{z}(\tilde{U}))$ [12]. Note that a cost function $I(U,V;\tilde{U},\tilde{V})$ means we treat the $n \times m$ matrix as a pool of nm data points and perform clustering of the all indices ignoring the row-column distinction. The compression cost of our co-clustering is $\frac{1}{m}I(U;\tilde{U}) + \frac{1}{n}I(V;\tilde{V})$.

Dataset	Newgroups included	# documents per group	Total # documents
Binary	talk.politics.mideast, talk.politics.misc	250	500
Multi5	comp.graphics, rec.motorcycles, rec.sports.baseball, sci.space, talk.politics.mideast	100	500
Multi10	alt.atheism, comp.sys.mac.hardware, misc.forsale, rec.autos, rec.sport.hockey, sci.crypt, sci.electronics, sci.med, sci.space, talk.politics.gun	50	500

Table 1: Datasets used in our experiments. We selected a random subset of different newsgroups.

4 Experimental Results

In this section, we demonstrate the performance of the proposed algorithms by applying them to synthetic and real-world data. The initialization of co-clustering algorithms must be done in the space of co-clustering assignments which, in contrast with the parameter space, grows exponentially in the size of the data matrix. Since the algorithms only find locally optimal solutions, this can severely restrict the algorithm's ability to find a good solution for large problems. We show that the results found with our algorithms yield better results than the previous hard-assignment variants. In all comparisons made in this section, we construct the hard version of our final results using the MAP estimates described in Section 3.2.

4.1 Gaussian Data

We first compare our Gaussian model co-clustering algorithm introduced in Section 2.3 and the BAC algorithm [10]. Since we showed in Section 3.1 that both methods have identical modeling assumptions for an appropriate choice of divergence, we use data generated from a Gaussian distribution and use Euclidean norm for the BAC algorithm. We generate the data by randomly sampling a matrix of $k \times l$ values from a zero-mean Gaussian distribution with variance σ_o^2. Using this matrix as the set of mean parameters $\boldsymbol{\mu} = [\mu_{\tilde{u}\tilde{v}}]$, we generate blocks of $n' \times m'$ samples from a unit-variance Gaussian distribution centered around any of its elements. The resulting $kn' \times lm'$ matrix is the synthetic data matrix $\boldsymbol{z}$ with known cluster assignments (c, g). In order to avoid a bias for matrix size, we generate 25,000 data sets with k and l randomly selected between 5 to 10, n' and m' randomly selected multiples of 10 between 50 and 200, and σ_o^2 sampled from a uniform distribution in the interval $[2, 5]$.

The comparison between the two algorithms is straightforward since both only require initialization for the co-clustering functions $(c^{(0)}, g^{(0)})$. We use the same sets of randomly generated assignments for both algorithms. We repeat the algorithms 5 times for each data set. As a measure of co-clustering performance, we employ *micro-averaged precision* [7] which is the proportion of the entire data indices correctly assigned to a category. Fig. 3.1 (left) shows that for a large number of data sets, we obtain higher precision than that of BAC. We further compare the results in terms of the cost function of BAC, namely, the average distance of any data point from its corresponding block mean. For any data set, We let D be the average distortion of the best result out of the 5 runs, and D' be that of our algorithm. We define the distortion reduction index as $(D - D')/(D + D')$. Fig. 3.1 (right) shows that for most data sets our co-clustering is better even in terms of BAC's own cost function. This suggests that our algorithm can better search the space of solutions around the initialization.

4.2 Co-occurence Data

We compare the performance of our model-based algorithm for co-occurrence data (Section 2.4) with the original hard-assignment algorithm [7] on the real world co-occurrence data. We use the same 20-Newsgroup data (*NG20*) employed in the empirical analysis of [7]. We perform preprocessing described in [13] and randomly select three different subsets of the documents defined as datasets Binary, Multi5, and Multi10 summarized in Table 1. We employ the feature selection procedure of [13] to choose 2,000 words for each dataset.

We run each algorithm 1,000 times. In each run we initialize both algorithms with the same random initialization. For our algorithm, the number of observations S acts as an extra parameter which we choose to be the constant value 10^6 in all our experiments. Moreover, we turn the initial configurations into soft-assignments by adding 0.1 to all $(\boldsymbol{q}^{(0)}, \boldsymbol{r}^{(0)})$ and renormalizing them. We use the information loss, cost function of ITC, as the performance measure. This measure is not exactly the cost function our algorithm is minimizing. If ΔI and $\Delta' I$ denote the information loss for ITC and our algorithm, we define the information gain of our algorithm for each initialization as $\Delta I - \Delta' I$. Fig. 2 presents the results which show that our soft assignment version of the ITC achieves better

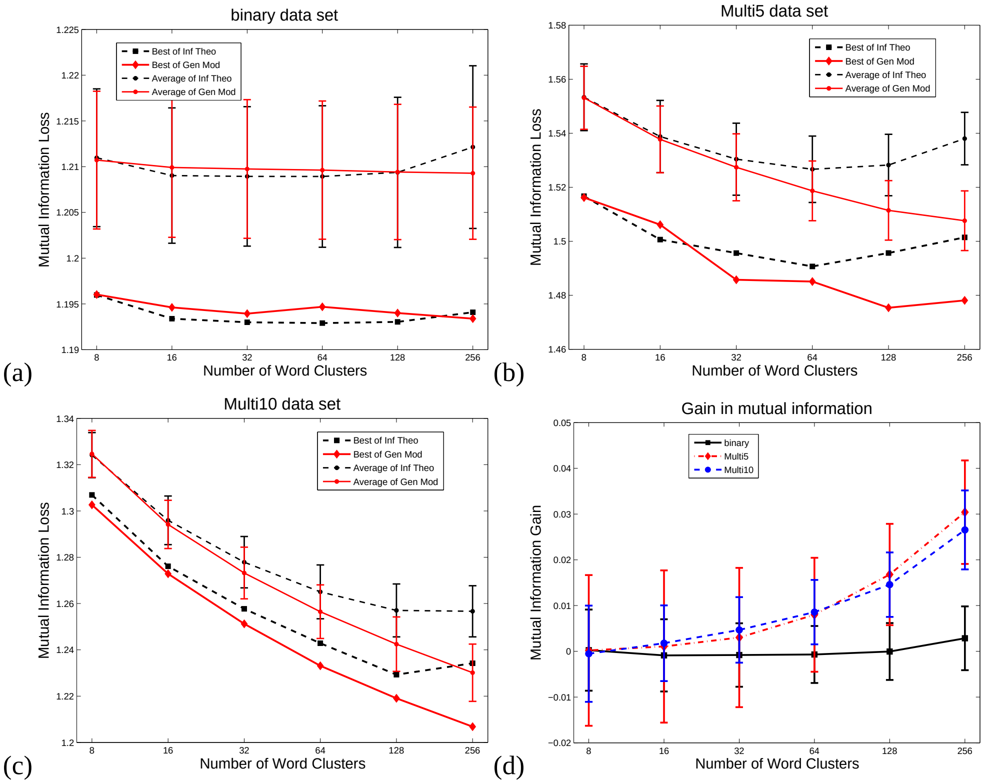

Figure 2: Results of the experiments on the *NG20* data (a)-(c). Mutual information loss in binary, Multi5, and Multi10 data sets, respectively, for different number of column (word) clusters. (d) The average gain of our algorithm over the ITC across all runs for different data sets.

measures of performance for most random initializations. As it is discussed in [7], the ITC is very sensitive to initialization. We aim to alleviate this problem by using our alternative soft-assignment maximum-likelihood problem. We see in Fig. 2d that with increasing complexity of the problem (from Binary to Multi5 and Mulit10), the relative performance of our algorithm improves.

In conclusion, we introduced a general generative model for co-clustering problems for different types of data and used the mean field approximation to derive a simple model-based algorithm. Our model reformulates the previous hard-assignment co-clusteirng algorithms as generative model problems, in the same way that EM-based mixture-model clustering extends k-means. We further demonstrated that with increasing complexity of the problem, our model-based algorithms achieve better results than their combinatorial counterparts when initialized with the same cluster configurations.

References

[1] J. A. Hartigan, "Direct Clustering of a Data Matrix," *J. Amer. Statistical Assoc.*, Vol. 67, pp. 123–129, 1972.

[2] Y. Cheng, and G. M. Church, "Biclustering of Expression Data," *Proceedings of ISMB*, AAAI Press, pp. 93–103, 2000.

[3] S. C. Madeira, A. L. Oliveira, "Biclustering Algorithms for Biological Data Analysis: A Survey," *IEEE/ACM Trans. Comput. Biol. Bioinf.*, Vol. 1, No. 1, pp. 24–44, 2004.

[4] Y. Kluger, R. Basri, J. T. Chang, M. Gerstein, "Spectral Biclustering of Microarray Data: Coclustering Genes and Conditions," *Genome Research*, Vol. 13, Part 4, pp. 703–716, 2003.

[5] I. S. Dhillon, "Co-clustering documents and words using bipartite spectral graph partitioning," *Proceedings of the 7th ACM SIGKDD (KDD'03)*, ACM Press, pp. 269–274, 2001.

[6] T. Hofmann, and J. Puzicha, "Statistical Models for Co-occurrence Data," *Technical Report*, Massachusetts Institute of Technology, Cambridge, MA, 1998.

[7] I. S. Dhillon, S. Mallela, and D. S. Modha, "Information Theoretic Co-clustering ," *Proceedings of 9th ACM SIGKDD (KDD'03)*, ACM Press, pp. 89–98, 2003.

[8] C. Kemp, J.. B. Tenenbaum, T. L. Griffiths, T. Yamada, and N. Ueda, "Learning systems of concepts with an infinte relational model," *Proceedings of AAAI-06*, 2006.

[9] B. Long, Z. Zhang, and P. S. Yu, "A Probabilistic Framework for Relational Clustering," *Proceedings of*

the 13th ACM SIGKDD (KDD'07), ACM Press, pp. 470–479, 2007.

[10] A. Banerjee, I. Dhillon, and J. Ghosh, S. Merugu, D. S. Modha,"A Generalized Maximum Entropy Approach to Bregman Co-Clustering and Matrix Approximation," *JMLR*, No. 8, pp. 1919–1986, 2007.

[11] D. J. C. MacKay, *Information theory, inference, and learning algorithms*, Cambridge, UK, Cambridge University Press, 2003.

[12] A. Banerjee, S. Merugu, I. Dhillon, and J. Ghosh, "Clustering with Bregman Divergences," *JMLR*, No. 6, pp. 1705-1749, 2005.

[13] N. Slonim, and N. Tishby, "Document Clustering using Word Clusters via the Information Bottleneck Method," *Proceedings of SIGIR-23*, pp. 208–215, 2000.