Simultaneous Unsupervised Learning of Disparate Clusterings

Prateek Jain*, Raghu Meka and Inderjit S. Dhillon

Department of Computer Sciences, University of Texas, Austin, TX 78712-1188, USA

Received 14 April 2008; accepted 05 May 2008
DOI:10.1002/sam.10007
Published online 3 November 2008 in Wiley InterScience (www.interscience.wiley.com).

Abstract: Most clustering algorithms produce a single clustering for a given dataset even when the data can be clustered naturally in multiple ways. In this paper, we address the difficult problem of uncovering disparate clusterings from the data in a *totally unsupervised manner*. We propose two new approaches for this problem. In the first approach, we aim to find good clusterings of the data that are also *decorrelated* with one another. To this end, we give a new and tractable characterization of decorrelation between clusterings, and present an objective function to capture it. We provide an iterative "decorrelated" k-means type algorithm to minimize this objective function. In the second approach, we model the data as a sum of mixtures and associate each mixture with a clustering. This approach leads us to the problem of learning a convolution of mixture distributions. Though the latter problem can be formulated as one of factorial learning [8,13,16], the existing formulations and methods do not perform well on many real high-dimensional datasets. We propose a new regularized factorial-learning framework that is more suitable for capturing the notion of disparate clusterings in modern, high-dimensional datasets. Furthermore, we provide kernelized version of both of our algorithms. The resulting algorithms do well in uncovering multiple clusterings, and are much improved over existing methods. We evaluate our methods on two real-world datasets—a music dataset from the text-mining domain, and a portrait dataset from the computer-vision domain. Our methods achieve a substantially higher accuracy than existing factorial learning as well as traditional clustering algorithms. © 2008 Wiley Periodicals, Inc. Statistical Analysis and Data Mining 1: 195–210, 2008

Keywords: disparate clustering; unsupervised learning; k-means; expectation maximization

1. INTRODUCTION

Clustering data into groups based on similarity is often one of the most important steps in any data-analysis application. Currently, most clustering algorithms partition the data into groups that are disjoint, while other algorithms extend this approach to probabilistic or overlapping clustering. However, in many important applications, it is necessary to uncover disparate or alternative clusterings[1] in order to reflect the different groupings inherent in the data. As an example, consider a set of pictures of different persons in different poses (see Fig. 1). These images can be clustered by the identity of the person in the picture or by the pose of the person. Given such a dataset, it would be desirable to recover two disparate clusterings of the data—one based on the identity of the person and the other based on their pose.

The above problem arises naturally for many other widely used datasets, for instance: news articles (can be clustered by the main topic, or by the news source), reviews of various musical albums (can be clustered by composers, or by other characteristics like genre of the album), and movies (can be clustered based on actors/actresses or genre).

Most existing methods to recover alternative clusterings use semisupervision or side-information about one or more of the clusterings. Since clustering is generally the first step in data analysis, such information might not be available beforehand. For example, news articles change dynamically and it is infeasible to manually label them by topic and the source. Thus, completely unsupervised techniques to find disparate clusterings are immensely useful.

In this paper, we present two novel unsupervised approaches for discovering disparate clusterings in a given dataset. In the first approach we aim to find multiple clusterings of the data which satisfy two criteria: (i) the clustering error of each individual clustering is small and (ii) different clusterings have small *correlation* between them. To this end, we present a new and computationally tractable characterization of *correlation* (or decorrelation) between different clusterings. We use this characterization to formulate a k-means type objective function which contains error terms for each individual clustering along with a regularization

Correspondence to: Prateek Jain (pjain@cs.utexas.edu)

[1] Throughout this paper, a *clustering* will refer to a set of disjoint clusters of the data.

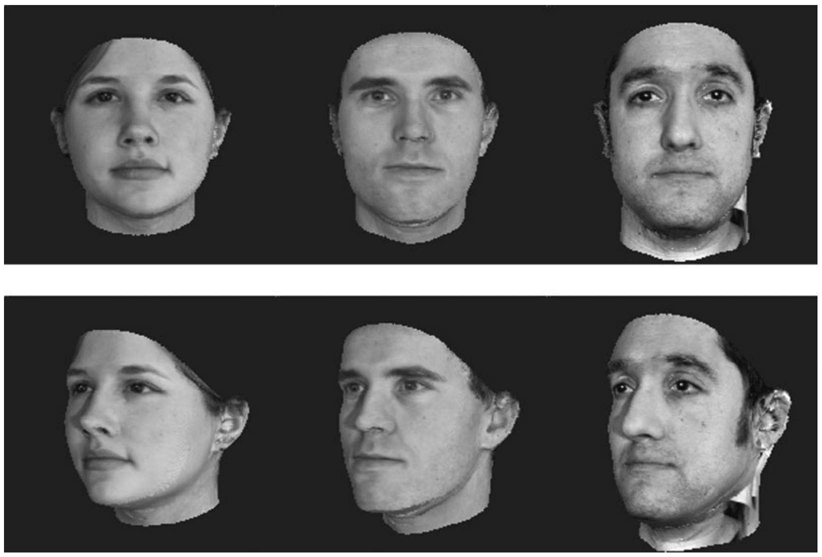


Fig. 1 Images of different persons in different poses. Each row has different persons in the same pose. Each column has the same person in different poses.

term corresponding to the correlation between clusterings. We provide a computationally efficient k-means type algorithm for minimizing this objective function.

In the second approach, we model the problem of finding disparate clusterings as one of learning the component distributions when the given data is sampled from a convolution of mixture distributions. This formulation is appropriate when the different clusterings come from independent additive components of the data. The problem of learning a convolution of mixture distributions is closely related to factorial learning [8,13,16]. However, the methods of [8,13,16] are not suited for recovering multiple clusterings. The problem with applying factorial learning directly is that there are multiple solutions to the problem of learning a convolution of mixture distributions. Out of all such possible solutions, the desirable solutions are the ones that give maximally disparate clusterings. To address this problem, we propose a regularized factorial-learning model that intuitively captures the notion of decorrelation between clusterings and aims to estimate the parameters of the decorrelated model.

An important aspect of both our approaches is the notion of decorrelation between clusterings. The decorrelation measures that we propose quantify the “orthogonality” between the mean vectors corresponding to different clusterings. We show that the characterization of disparity between different clusterings by the “orthogonality” between the mean vectors of the respective cluster centers has a well-founded theoretical basis (see Section 5.1).

Both of our algorithms assume linear separability of the clusters, which does not hold for various real-life problems. To overcome this limitation, we provide kernelized versions of both our algorithms.

We evaluate our methods on synthetic and real-world datasets that have multiple disparate clusterings. We consider real-world datasets from two different domains—a music dataset from the text-mining domain and a portrait dataset from the computer-vision domain. We compare our methods to two factorial-learning algorithms, cooperative vector quantization (CVQ) [8] and multiple cause vector quantization (MCVQ) [16]. We also compare against traditional single-clustering algorithms like k-means and nonnegative matrix approximation (NNMA) [15]. On all the datasets, both of our algorithms significantly outperform the factorial learning as well as the single-clustering algorithms. The factorial-learning methods work reasonably well on a few synthetic datasets that exactly satisfy their respective model assumptions. But they are not robust in the case where model assumptions are even slightly violated. Because of this, their performance is poor on real-world datasets and other synthetic datasets. In comparison, our algorithms are more robust and perform significantly better on all the datasets. For the music dataset both our algorithms achieve around 20% improvement in accuracy over the factorial-learning and single-clustering algorithms (k-means and NNMA). Similarly, for the portrait dataset we achieve an improvement of 30% over the baseline algorithms.

2. RELATED WORK

Most of the existing work for finding disparate clusterings has been in the semisupervised setting. The semisupervised clustering problem of finding a clustering consistent with a given set of constraints has been extensively studied [21, 23, 2]. This approach has been applied to the problem of recovering multiple clusterings by providing appropriate constraints. Must-link and cannot-link constraints have been extensively used for semisupervised clustering [21, 22, 2]. Recently, Davidson *et al.* [4] proposed an efficient incremental algorithm for must-link and cannot-link constraints. An alternative approach to the problem is taken by [1,10,11] where it is assumed that a clustering of the data is given and the objective is to find a clustering different from the given one. Our work differs from the above approaches in that our methods for discovering the disparate clusterings are completely unsupervised.

A supervised approach to the related problem of discovering two-factor structures from the observed data was suggested in [20]. Their method, named separable mixture model (SMM), models the data using a bilinear function of the factors and can also be used for obtaining two clusterings of the data. An advantage of our methods over SMM is that our methods are unsupervised compared to the supervised approach of SMM. Also, our model can be extended to more than two factors, whereas it is unclear how SMM could be extended to a data generated from more than two factors.

Our second approach ('sum of parts' approach) is closely related to the factorial-learning problem where each data point is assumed to be generated by combining multiple factors. Ghahramani [8] introduced a novel architecture named CVQ, in which a set of multiple vector quantizers (VQ) combine linearly to generate the input data. However, a drawback of CVQ is that it can have multiple solutions. Many of these solutions give poor results for the problem of discovering disparate clusterings, especially on our real-world applications. Also, CVQ can be seen as a special case of our model. Another recent model related to factorial learning is MCVQ (Ross and Zemel [16]). In MCVQ, it is assumed that the dimensions of the data can be separated into several disjoint factors, which take on values independently of each other. The factors are then modeled using a VQ as in CVQ. However, MCVQ also faces the same drawbacks of CVQ—existence of multiple solutions—which leads to poor performance for our application of discovering disparate clusterings.

The problem of learning convolutions of distributions that forms the basis of our second approach has been considered in the statistics community—see for instance [6], [18], [17]. However, these methods deal with learning convolutions of simple distributions like binomial, Gaussian and Poisson, and do not consider mixtures of distributions. A fundamental problem with learning a convolution of Gaussians, as mentioned in [18], is that the problem is not well-defined—there exist many solutions to the learning problem. We face a similar problem in the M-step of our algorithm for learning the convolution of mixtures of Gaussians, where the maximum likelihood estimation has multiple solutions. We deal with this issue by regularizing the solution space in a way suitable for the purpose of recovering disparate clusterings so that the problem becomes well-posed.

We emphasize that though we state the problem of recovering disparate clusterings as one of learning independent components from the data, the problem we address is completely different from that of independent component analysis (ICA) [14]. ICA tries to separate a multivariate signal into independent additive univariate signals, whereas in our problem we try to decompose the signal into independent multivariate signals, each of which may have high correlation between its different dimensions.

For our experiments, we also evaluated various simple extensions of k-means such as the removal of important features of the first clustering to uncover the second clustering, and projection of the data onto the space orthogonal to the means of the first clustering. The later heuristic was motivated by principal gene shaving [12]. But, these approaches are *ad hoc* and do not perform well in our experiments.

3. METHODOLOGY

For simplicity, we present our methods for uncovering two disparate clusterings from the data; our techniques can be generalized to uncover more than two clusterings. We propose the following approaches:

- Decorrelated k-means approach: In this approach, we try to fit each clustering to the entire data, while requiring that different clusterings be decorrelated with each other. To this end, we introduce a novel measure for correlation between clusterings. This measure is motivated by the fact that if the representative vectors of two clusterings are orthogonal to one another, then the labelings generated by nearest-neighbor assignments for these representative vectors are independent under some mild conditions (see Section 5.1).
- Sum of parts approach: In this approach, we model the data as a sum of independent components, each of which is a mixture model. We then associate each component with a clustering. Further, as the distribution of the sum of two independent random variables is the convolution of the distributions (see [5]), we model the observed data as being sampled from a convolution of two mixtures. Thus, our approach leads us to the problem of learning

a convolution of mixtures. Note that the individual components uncovered by this approach may not be good approximations to the data by themselves, but their sum is. This is in complete contrast to the first approach where we try to approximate the data individually by each component.

3.1. First Approach: Decorrelated k-means

Given a set of data points $Z = \{z_1, z_2, \ldots, z_n\} \subseteq \mathbb{R}^m$, we aim to uncover two clusterings C^1 and C^2. Specifically, we wish to partition the set Z into k_1 groups for the first clustering C^1 and k_2 groups for the second clustering C^2. To achieve this task, we try to find *decorrelated* clusterings each of which approximates the data as a whole. We propose the following objective function:

$$\begin{aligned} G(\mu_{1\ldots k_1}, \mu_{1\ldots k_2}) &= \sum_i \sum_{z \in C_i^1} \|z - \mu_i\|^2 \\ &+ \sum_j \sum_{z \in C_j^2} \|z - \nu_j\|^2 \\ &+ \lambda \sum_{i,j} (\beta_j^T \mu_i)^2 \\ &+ \lambda \sum_{i,j} (\alpha_i^T \nu_j)^2, \end{aligned} \tag{1}$$

where C_i^1 is cluster i of the first clustering, C_j^2 is cluster j of the second clustering, and $\lambda > 0$ is a regularization parameter. The vector μ_i is the *representative* vector of C_i^1, ν_j is the *representative* vector of C_j^2, α_i is the mean vector of C_i^1 and β_j is the mean vector of C_j^2.

The first two terms of Eq. (1) correspond to a k-means type error term for the clusterings, with a crucial difference being that the “representative” vector of a cluster may not be its mean vector. The last two terms are regularization terms that measure the decorrelation between the two clusterings. In order to extend this formulation for $T \geq 2$ clusterings, we add k-means type error terms for each of the T clusterings. Furthermore, we add $T \times (T-1)/2$ terms corresponding to the decorrelation between pairs of clusterings.

The decorrelation measure given above is motivated by the intuition that if the “representative” vectors of two clusterings are orthogonal to one another, then the labelings generated by nearest-neighbor assignments for these vectors are independent. We provide a theoretical basis for the above intuition in Section 5.1 Also, an important advantage of the proposed decorrelation measure is that the objective function remains strictly and jointly convex in the μ_is and ν_js (assuming fixed C_i^1s and C_j^2s).

To minimize the objective function in Eq. (1), we present an iterative algorithm, which we call decorrelated k-means (Algorithm 1). We fix C^1 and C^2 to obtain μ_is and ν_js that minimize Eq. (1) and then assign each point z to C_i^1 such that $i = \text{argmin}_l \|z - \mu_l\|^2$ and to C_j^2 such that $j = \text{argmin}_l \|z - \nu_l\|^2$. We initialize one of the clusterings using k-means with $k = k_1$ and the other clustering randomly.

For computing the μ_is and ν_js, we need to minimize Eq. (1). The gradient of the objective function in Eq. (1) w.r.t μ_i is given by:

$$\frac{\partial G}{\partial \mu_i} = -2\left(\sum_{z \in C_i^1} z\right) + 2\left(\sum_j n_{ij}\right)\mu_i + 2\lambda \sum_j (\beta_j^T \mu_i)\beta_j,$$

where n_{ij} is the number of points that belong to C_i^1 and C_j^2.

Now, $(\beta_j^T \mu_i)\beta_j = (\beta_j \beta_j^T)\mu_i$ and $\alpha_i = \frac{\left(\sum_{z \in C_i^1} z\right)}{\sum_j n_{ij}}$. Thus,

$$\frac{\partial G}{\partial \mu_i} = -2\sum_j n_{ij}\alpha_i + 2\sum_j n_{ij}\mu_i + 2\lambda\left(\sum_j \beta_j \beta_j^T\right)\mu_i.$$

Similarly,

$$\frac{\partial G}{\partial \nu_j} = -2\sum_i n_{ij}\beta_j + 2\sum_i n_{ij}\nu_j + 2\lambda\left(\sum_i \alpha_i \alpha_i^T\right)\nu_j.$$

Setting the gradients to zero gives us the following equations:

$$\mu_i = \left(I + \frac{\lambda}{\sum_j n_{ij}} \sum_j \beta_j \beta_j^T\right)^{-1} \alpha_i, \tag{2}$$

$$\nu_j = \left(I + \frac{\lambda}{\sum_i n_{ij}} \sum_i \alpha_i \alpha_i^T\right)^{-1} \beta_j. \tag{3}$$

Since the objective function Eq. (1) is strictly and jointly convex in both μ_is and ν_js, the above updates lead to a global minima of the objective function Eq. (1) for *fixed* C^1 and C^2.

3.1.1. *Computing the updates efficiently*

Computing the updates given by Eqs. (2) and (3) requires computing the inverse of an $m \times m$ matrix, where m is the dimensionality of the data. Thus updating all the $\boldsymbol{\mu}_i$s and $\boldsymbol{\nu}_j$s directly would seem to require $O(k_1 m^3 + k_2 m^3)$ operations, which is cubic in the dimensionality of the data. We now give a substantially faster way to compute the updates in time linear in the dimensionality. Using the Sherman-Morrison-Woodbury formula (see [9]) for the inverse in Eq. (2), we get

$$\left(I + \xi_i V V^T\right)^{-1} = I - \xi_i V \left(I + \xi_i V^T V\right)^{-1} V^T,$$

where $\xi_i = \frac{\lambda}{\sum_j n_{ij}}$ and $V = [\boldsymbol{\beta}_1, \ldots, \boldsymbol{\beta}_{k_2}]$. Using the eigenvalue decomposition $V^T V = Q \Sigma Q^T$ we see that

$$\left(I + \xi_i V^T V\right)^{-1} = Q \left(I + \xi_i \Sigma\right)^{-1} Q^T.$$

Since $V^T V$ is a $k_2 \times k_2$ matrix its eigenvalue decomposition can be computed in $O(k_2^3)$ time. Also, as $(I + \xi_i \Sigma)^{-1}$ is a diagonal matrix, calculating its inverse requires just $O(k_2)$ operations. The updates for $\boldsymbol{\mu}_i$s can now be rewritten as,

$$\boldsymbol{\mu}_i = \left(I - \xi_i V Q \left(I + \xi_i \Sigma\right)^{-1} Q^T V^T\right) \boldsymbol{\alpha}_i. \tag{4}$$

Similarly, the updates for $\boldsymbol{\nu}_j$s can now be written as,

$$\boldsymbol{\nu}_j = \left(I - \zeta_j M U \left(I + \zeta_j \Lambda\right)^{-1} U^T M^T\right) \boldsymbol{\beta}_j, \tag{5}$$

where $\zeta_j = \frac{\lambda}{\sum_i n_{ij}}$, $M = [\boldsymbol{\alpha}_1, \ldots, \boldsymbol{\alpha}_{k_1}]$, and $U \Lambda U^T$ is the eigenvalue decomposition of $M^T M$.

Using these updates reduces the computational complexity of computing all the $\boldsymbol{\mu}_i$s and $\boldsymbol{\nu}_j$s to $O(mk_1^2 + mk_2^2 + k_1^3 + k_2^3)$. If $m > k = \max(k_1, k_2)$, which is typically the case, the above bound becomes $O(mk^2)$.

3.1.2. *Determining* $\boldsymbol{\lambda}$

The regularization parameter λ plays an important role in the decorrelated k-means algorithm. It determines the trade-off between minimizing the individual clustering error of each clustering (first two terms in Eq. (1)) and finding decorrelated cluster centers for the different clusterings (last two terms in Eq. (1)). Empirically, we observe that the clustering accuracies are good when $\lambda \in [100, 10000]$, which is a large range. But, a different scaling of the data can change this range for λ. Hence, we determine λ using a simple heuristic. Note that for small values of λ, the decorrelated k-means algorithm finds approximately the same clusters for both the clusterings. While for a high value of λ, it tries to find clusterings that are orthogonal to each other, even though both the clusterings may not fit the data well. Thus, a suitable λ balances out both the objectives and hence generally there is a large change in objective function value when λ is perturbed slightly. On the basis of this intuition, we form a heuristic to determine λ: start with a large λ and find different clusterings of the data while decreasing λ, and select a λ for which the drop in the objective function is the highest. Note that different variants of the heuristic can be used depending on the data and domain knowledge. For example, if the data is large, then a subset of the data can be used for finding clusterings or if the data is noisy then a more robust measure like average change in objective function should be preferred over the maximum change measure for selecting λ.

3.2. Second Approach: Sum of Parts

In this section, we describe our "sum of parts" approach. Let $\mathcal{Z} = \{z_1, \ldots, z_n\}$ be the observed m-dimensional data sampled from a random variable Z. We model Z as a sum $X + Y$, where X, Y are independent random variables and are drawn from mixtures of distributions. Specifically,

$$p_X = \sum_{i=1}^{k_1} a_i p_{X_i}, \qquad p_Y = \sum_{j=1}^{k_2} b_j p_{Y_j}.$$

The problem of learning independent components can now be stated as: Given data sampled according to Z, recover the parameters of the probability distributions p_{X_i}, p_{Y_j} along with the mixing weights a_i, b_j.

As $Z = X + Y$, the probability density function of Z is the convolution of p_X and p_Y [5, Section A.4.11]. Thus,

$$p_Z(z) = (p_X * p_Y)(z) = \sum_{i=1}^{k_1} \sum_{j=1}^{k_2} (a_i b_j) \cdot (p_{X_i} * p_{Y_j})(z)\ , \tag{6}$$

where $f_1 * f_2(z) = \int_{\mathbb{R}^m} f_1(\boldsymbol{x}) \cdot f_2(z - \boldsymbol{x}) d\boldsymbol{x}$ denotes the convolution of f_1 and f_2.

From Eq. (6) it follows that when the distributions p_{X_i} and p_{Y_j} belong to a family of distributions closed under convolution, Z can be viewed as a mixture of $k_1 \times k_2$ distributions. However, the problem of learning the components X and Y from Z is harder than that of simply learning the parameters of a mixture model, as along with learning the $k_1 \times k_2$ component distributions one must also be able to factor them out. In the following section, we give a generalized expectation maximization (EM) algorithm for learning the parameters of the component mixtures when the base distributions are spherical multivariate Gaussians. Our techniques can be extended to more general distributions like nonspherical Gaussians and potentially to other families closed under convolution.

3.2.1. *Learning the convolution of a mixture of Gaussians*

Let the components X and Y be mixtures of spherical Gaussians, i.e., $p_X = \sum_{i=1}^{k_1} a_i \mathcal{N}(\boldsymbol{\mu}_i, \sigma^2)$ and $p_Y = \sum_{i=1}^{k_2} b_i \mathcal{N}(\boldsymbol{\nu}_i, \sigma^2)$. As in our first approach, we initialize the EM algorithm (Algorithm 2) by k-means for the first clustering and a random assignment for the second clustering. We initialize $\boldsymbol{\mu}_i^0$s and $\boldsymbol{\nu}_j^0$s to be the means of the first and second clusterings respectively. To initialize σ we use a heuristic presented in [3],

$$\sigma = \frac{1}{\sqrt{2m}} \min\left(\min_{i \neq j} \|\boldsymbol{\mu}_i^0 - \boldsymbol{\mu}_j^0\|, \min_{i \neq j} \|\boldsymbol{\nu}_i^0 - \boldsymbol{\nu}_j^0\| \right).$$

E-step:

Let $p_{ij}^t(z)$ denote the conditional probability that z comes from the Gaussian $p_{X_i} * p_{Y_j}$ given the current parameters. As our main objective is to cluster the data, we use hard assignments in the E-step to ease the computations involved. The E-step in this case will be:

$$p_{ij}^{t+1}(z) = \begin{cases} 1, & \text{if } (i,j) = \\ & \operatorname{argmax}_{(r,s)} \{a_r^t b_s^t \cdot \mathcal{N}\left(\boldsymbol{\mu}_r^t + \right. \\ & \left.\boldsymbol{\nu}_s^t, 2(\sigma^t)^2\right)(z)\} \\ 0, & \text{otherwise.} \end{cases} \tag{7}$$

Note that, to uncover T different clusterings from the data, $O(k^T)$ computational operations are required for each data point in the E-step. Ghahramani [8] suggested various approximation methods to reduce the time complexity of this estimation, and the same can be applied to our setting as well. In our implementation, we use Gibbs sampling for approximating the distribution of labels, $p_{ij}^{t+1}(z)$, when the parameters of the base distributions are fixed.

M-step:

In the M-step, we use the clusterings updated in the E-step (specified by p_{ij}^{t+1}'s) to estimate the parameters of the distributions. Formally, we maximize the log-likelihood:

$$\left(\boldsymbol{\mu}_{1\ldots k_1}^{t+1}, \boldsymbol{\nu}_{1\ldots k_2}^{t+1}, \sigma^{t+1}, a_{1\ldots k_1}^{t+1}, b_{1\ldots k_2}^{t+1}\right) = \operatorname*{argmax}_{\substack{\boldsymbol{\mu}_{1\ldots k_1}, \boldsymbol{\nu}_{1\ldots k_2}, \sigma, \\ a_{1\ldots k_1}, b_{1\ldots k_2}}} \sum_{i,j,z} p_{ij}^{t+1}(z)\ \log\left(a_i b_j \mathcal{N}(\boldsymbol{\mu}_i + \boldsymbol{\nu}_j, \sigma)(z)\right).$$

The mixture weights and variance σ can be easily computed by differentiating w.r.t. a_is, b_js, σ and setting the derivatives to zero. This gives us the following expressions:

$$a_i^{t+1} = \frac{1}{n} \sum_j \sum_z p_{ij}^{t+1}(z), \tag{8}$$

$$b_j^{t+1} = \frac{1}{n} \sum_i \sum_z p_{ij}^{t+1}(z), \tag{9}$$

$$(\sigma^{t+1})^2 = \frac{1}{2mn} \sum_{i,j,z} p_{ij}^{t+1}(z)\ \|z - \boldsymbol{\mu}_i^t - \boldsymbol{\nu}_j^t\|^2. \tag{10}$$

Computing the means to maximize the log-likelihood is more involved and it reduces to minimizing the following objective function:

$$\min_{\boldsymbol{\mu}_{1\ldots k_1}, \boldsymbol{\nu}_{1\ldots k_2}} F(\boldsymbol{\mu}_{1\ldots k_1}, \boldsymbol{\nu}_{1\ldots k_2}) = \sum_{i,j,z} p_{ij}^{t+1}(z) \|z - \boldsymbol{\mu}_i - \boldsymbol{\nu}_j\|^2. \tag{11}$$

Note that multiple solutions exist for the above equation; since we can translate the means $\boldsymbol{\mu}_i$s by a fixed vector $\boldsymbol{w}$ and the means $\boldsymbol{\nu}_j$s by $-\boldsymbol{w}$ to get another set of solutions. As discussed in Section 2, the CVQ [8] algorithm also suffers from the same problem of multiple solutions. Out of all the solutions to Eq. (11), the solutions which give maximally disparate clusterings are more desirable. To obtain such solutions, we regularize the $\boldsymbol{\mu}_i$s and $\boldsymbol{\nu}_j$s to have small correlation with each other. To this end, we introduce a regularization term to make the $\boldsymbol{\mu}_i$s and $\boldsymbol{\nu}_j$s orthogonal to one another. This correlation measure is similar to the measure discussed in the previous decorrelated k-means approach (Section 3.1). Formally, we minimize the following objective function:

$$\tilde{F}(\boldsymbol{\mu}_{1\ldots k_1}, \boldsymbol{\nu}_{1\ldots k_2}) = \textstyle\sum_{i,j,z} p_{ij}^{t+1}(z) \|z - \boldsymbol{\mu}_i - \boldsymbol{\nu}_j\|^2 + \lambda \sum_{i,j} (\boldsymbol{\mu}_i^T \boldsymbol{\nu}_j)^2, \tag{12}$$

where $\lambda > 0$ is a regularization parameter and can be selected using a heuristic similar to the one described in Section 3.1.2.

Observe that the above objective is not jointly convex in $\boldsymbol{\mu}_i$ and $\boldsymbol{\nu}_j$ but is strictly-convex in $\boldsymbol{\mu}_i$ for fixed $\boldsymbol{\nu}_j$'s and vice versa. To minimize $\tilde{F}$, we use the block coordinate descent algorithm [24] where we fix $\boldsymbol{\nu}_j$'s to minimize $\boldsymbol{\mu}_i$ and vice versa. By differentiating Eq. (12) w.r.t. $\boldsymbol{\mu}_i$ and $\boldsymbol{\nu}_j$ and setting the derivatives to zero we get,

$$\left(I + \frac{\lambda \sum_j \boldsymbol{\nu}_j \boldsymbol{\nu}_j^T}{\sum_j n_{ij}}\right) \boldsymbol{\mu}_i + \frac{\sum_j n_{ij} \boldsymbol{\nu}_j}{\sum_j n_{ij}} = \boldsymbol{\alpha}_i,$$

$$\left(I + \frac{\lambda \sum_i \boldsymbol{\mu}_i \boldsymbol{\mu}_i^T}{\sum_i n_{ij}}\right) \boldsymbol{\nu}_j + \frac{\sum_i n_{ij} \boldsymbol{\mu}_i}{\sum_i n_{ij}} = \boldsymbol{\beta}_j,$$

where $n_{ij} = \sum_z p_{ij}^{t+1}(z)$ is the number of data points that belong to cluster i of the first clustering and cluster j of the second clustering, $\boldsymbol{\alpha}_i$ denotes the mean of all points that are assigned to cluster i in the first clustering and $\boldsymbol{\beta}_j$ denotes the mean of points assigned to cluster j in the second clustering, i.e. ,

$$\boldsymbol{\alpha}_i = \frac{1}{na_i} \sum_j \sum_z z\, p_{ij}^{t+1}(z), \tag{13}$$

$$\boldsymbol{\beta}_j = \frac{1}{nb_j} \sum_i \sum_z p_{ij}^{t+1}(z). \tag{14}$$

To solve for $\boldsymbol{\mu}_i$ and $\boldsymbol{\nu}_j$ in the above equations, we use an alternative minimization scheme—we iteratively update the $\boldsymbol{\mu}_i$ and $\boldsymbol{\nu}_j$ as follows:

$$\boldsymbol{\mu}_i = \left(I + \frac{\lambda \sum_j \boldsymbol{\nu}_j \boldsymbol{\nu}_j^T}{\sum_j n_{ij}}\right)^{-1} \left(\boldsymbol{\alpha}_i - \frac{\sum_j n_{ij} \boldsymbol{\nu}_j}{\sum_j n_{ij}}\right) \tag{15}$$

$$\boldsymbol{\nu}_j = \left(I + \frac{\lambda \sum_i \boldsymbol{\mu}_i \boldsymbol{\mu}_i^T}{\sum_i n_{ij}}\right)^{-1} \left(\boldsymbol{\beta}_j - \frac{\sum_i n_{ij} \boldsymbol{\mu}_i}{\sum_i n_{ij}}\right). \tag{16}$$

For initialization, we set $\boldsymbol{\nu}_j$ to be $\boldsymbol{\beta}_j$ for each j. Below we prove that this scheme converges to a local minima of Eq. (12).

LEMMA 1: The updates for $\boldsymbol{\mu}_i$ and $\boldsymbol{\nu}_j$ given by Eq. (15) converge to a local minimum of the regularized objective function given by Eq. (12).

PROOF 1: As the updates Eq. (15) minimize the objective function at each iteration, the updates converge to a fixed point [24]. Also, the objective function Eq. (12) is strictly convex in $\boldsymbol{\mu}_i$ for fixed $\boldsymbol{\nu}_j$s and vice versa. Thus, any fixed point of Eq. (12) is also a local minimum. It now follows that our updates converge to a local minimum of the objective function. □

THEOREM 1: Algorithm 2 monotonically decreases the objective function:

$$F = \sum_{i,j,z} p_{ij}^{t+1}(z) \frac{\|z - \boldsymbol{\mu}_i - \boldsymbol{\nu}_j\|^2}{2\sigma^2} + \lambda \sum_{i,j} (\boldsymbol{\mu}_i^T \boldsymbol{\nu}_j)^2 \tag{17}$$

PROOF 2: Let F_t be the objective function value at the start of t-th iteration, F_t^E be the objective function value after the E-step of t-th iteration and $F_t^M = F_{t+1}$ be the objective function after M-step of t-th iteration. The E-step assigns new labels according to Eq. (7), which is equivalent to minimizing:

$$\sum_{i,j,z} p_{ij}^{t+1}(z) \frac{\|z - \boldsymbol{\mu}_i - \boldsymbol{\nu}_j\|^2}{2\sigma^2},$$

with μ_i and ν_j being fixed.

Thus, the first term of the objective function (17) is decreased by the E-step while the second term remains fixed. Hence, $F_t \geq F_t^E$. Using Lemma 1, $F_t^E \geq F_t^M$, as only μ_is and ν_js are variables with p_{ij} fixed (σ can be absorbed in λ). Thus, $F_t \geq F_{t+1}$. □

3.2.2. *Computing the updates efficiently*

Using techniques similar to Section 3.1.1, the update for $\boldsymbol{\mu}_i$ can be written as:

$$\boldsymbol{\mu}_i = \left(I - \xi_i V Q \left(I + \xi_i \Sigma\right)^{-1} Q^T V^T\right) \left(\boldsymbol{\alpha}_i - \frac{\sum_j n_{ij} \boldsymbol{\nu}_j}{\sum_j n_{ij}}\right), \tag{18}$$

where, $\xi_i = \frac{\lambda}{\sum_j n_{ij}}$ and $V = [\boldsymbol{\nu}_1, \ldots, \boldsymbol{\nu}_{k_2}]$ and $Q \Sigma Q^T$ is the eigenvalue decomposition of $V^T V$.

Similarly, the update for $\boldsymbol{\nu}_j$ can be written as,

$$\boldsymbol{\nu}_j = \left(I - zeta_j MU \left(I + zeta_j \Lambda\right)^{-1} U^T M^T\right)$$
$$(\boldsymbol{\beta}_j - \tfrac{\sum_i n_{ij} \boldsymbol{\mu}_i}{\sum_i n_{ij}}), \tag{19}$$

where, $\zeta_j = \frac{\lambda}{\sum_i n_{ij}}$, $M = [\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_{k_1}]$ and $M^T M = U \Lambda U^T$.

As in Section 3.1.1, the above updates reduce the computational complexity of computing all the $\boldsymbol{\mu}_i$s and $\boldsymbol{\nu}_j$s from O($k_1 m^3 + k_2 m^3$) to $O(m(k_1^2 + k_2^2))$.

4. KERNELIZATION

In Section 3, we proposed two algorithms for learning disparate clusterings from the given data. Both of the proposed methods assume that the given data is linearly separable. This might not hold in general (Fig. 2)—more so for real-world applications. Over the past decade, kernel methods have been shown to successfully overcome the

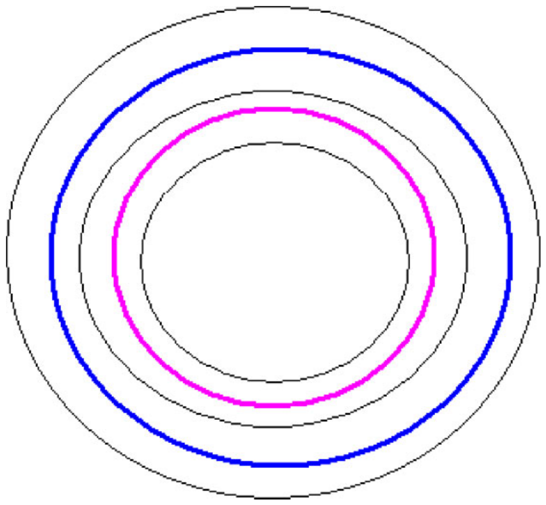


Fig. 2 A case where multiple clusterings exists in the data but the data is not linearly separable. Thin and black circles show the data points, while bold and colored circles show the separating surfaces for generating two sets of disparate clusters.

limitation of linearly separability in feature space. Typically, kernel methods map the data into a high-dimensional hilbert space with a hope that the data will be linearly separable in the new high-dimensional feature space. Furthermore, it is assumed that the inner product in the new feature space can be computed efficiently. Thus kernelization of a method involves expressing the algorithm in terms of inner products between the input data points. In this section, we provide kernel versions of our methods, Decorrelated k-means and Convolutional-EM.

4.1. Kernelized Decorrelated k-means

Given a kernel matrix K over the input data points, s.t.

$$K(i, j) = \phi(z_i)^T \phi(z_j),$$

where $\phi(z)$ is the (high-dimensional) feature map associated with the input data point z. We first provide a simple result regarding kernelization. Let $u = \sum_i a_i \phi(z_i)$ and $v = \sum_j b_j \phi(z_j)$ with $\boldsymbol{a}, \boldsymbol{b} \in \mathbb{R}^n$ be two linear combinations of the feature mappings of the input data points. Then

$$K(u, v) = u^T v = \sum_{ij} a_i b_j K(i, j), \tag{20}$$

that is, if the coefficients of linear combination of feature mappings are known, then the inner product or kernel value between the linear combinations can be computed efficiently. Now, we give step by step kernelization of the Decorrleated k-means algorithm.

Algorithm 1 Decorrelated k-means (Dec. k-means)

```
Input: Data $\mathcal{Z} = \{z_1, z_2, \ldots, z_n\}$
    $k_1$: Number of clusters in first clustering ($C^1$)
    $k_2$: Number of clusters in second clustering($C^2$)
    $\lambda$: regularization parameter
Output: $C^1, C^2$: Two different clusterings
1. $C^1 \leftarrow k$-means($\mathcal{Z}$), $C^2 \leftarrow$ Random assignment
2. repeat
  2.1. $\alpha_i \leftarrow$ ComputeMean($C^1_i$), for all $1 \le i \le k_1$
  2.2. $\beta_j \leftarrow$ ComputeMean($C^2_j$), for all $1 \le j \le k_2$
  2.3. Update $\boldsymbol{\mu}_i$ and $\boldsymbol{\nu}_j$ for all $i, j$ using Eqs. (4) and (5)
  2.4. $\forall z$, $C^1_i \leftarrow C^1_i \cup \{z\}$,
       if $i = \arg\min_l \|z - \boldsymbol{\mu}_l\|^2$.
  2.5. $\forall z$, $C^2_j \leftarrow C^2_j \cup \{z\}$,
       if $j = \arg\min_l \|z - \boldsymbol{\nu}_l\|^2$.
3. until convergence
return $C^1$, $C^2$
```

Step 1. in Algorithm 1 can be kernelized using Kernel K-means with K as the input kernel. Random assignment of C^2 clustering can be carried out as before.

Step 2.1, 2.2 compute means of the clusters for clusterings C^1 and C^2. We do not compute the explicit mappings for the mean vectors ($\boldsymbol{\alpha}_i$, $\boldsymbol{\beta}_j$), but just store the contributing weight of each input point to each mean vector.

Similarly, we do not compute explicit mappings for representative vectors ($\boldsymbol{\mu}_i$, $\boldsymbol{\nu}_j$), rather we show that the representative vectors are linear combinations of the feature mapping of input points and find out the corresponding weight of each input point in the linear combination. Recall that the update for $\boldsymbol{\mu}_i$ is given by:

$$\boldsymbol{\mu}_i = \left(I - \xi_i V Q \,(I + \xi_i \Sigma)^{-1} Q^T V^T\right) \boldsymbol{\alpha}_i, \tag{21}$$

where $V = [\boldsymbol{\beta}_1, \ldots, \boldsymbol{\beta}_{k_2}]$, $V^T V = Q \Sigma Q^T$. Now $(V^T V)_{ij} = \boldsymbol{\beta}_i^T \boldsymbol{\beta}_j$ can be computed in the new feature space or kernel space using Eq. (20), as the mean of a cluster can be written as a linear combination of the input data points. Thus $V^T V$ matrix can be computed in the kernel space. Similarly $V^T \boldsymbol{\alpha}_i$ can be computed in the kernel space using Eq. (20). Thus $\boldsymbol{\mu}_i$ can be written as a linear combination of feature mappings of the input points (z_i) and we can compute the weights of each input point efficiently.

Now, to assign a cluster from clustering C^1 for an input point z we need to solve the following problem:

$$i = \arg\min_l \|z - \boldsymbol{\mu}_l\|^2 = K(z, z) + K(\mu, \mu) - 2K(z, \mu).$$

Thus, label assignment for clustering C^1 (Step 2.4, Algorithm 1) can be performed in the kernel space. A similar method can be used to perform label assignment for clustering C^2 as well (Step 2.5, Algorithm 1). Thus, Algorithm 1 can be kernelized.

4.2. Kernelized Convolutional-EM

As for decorrelated k-means, we give step by step kernelization of the Convolution-EM algorithm (Algorithm 2). We assume an input kernel matrix K over the input data points, s.t.

$$K(i, j) = \phi(z_i)^T \phi(z_j).$$

Note that the update for σ (Eq. (10)) uses dimensionality of the feature mapping (m) which might not be available in general. Thus, we give kernelized Convolutional-EM algorithm for the case of fixed $\sigma^2 = c$, where c is a constant.

As in Kernelized Decorrelated-k-means, we show that μ_i and ν_j computed by Step 4.5 (Algorithm 2) are linear combination of feature mappings of the input data points.

CLAIM 1: Let μ_i and ν_j be computed using Step 4.5 (Algorithm 2), then:

$$\mu_i = \sum_l q_l \phi(z_l), \quad \nu_j = \sum_l g_l \phi(z_l),$$

where $\boldsymbol{q} \in \mathbb{R}^n$ and $\boldsymbol{g} \in \mathbb{R}^n$.

Algorithm 2 Convolutional-EM (Conv-EM)

```
Input: Data $\mathcal{Z} = \{z_1, z_2, \ldots, z_n\}$
    $k_1$: Number of clusters in first clustering ($C^1$)
    $k_2$: Number of clusters in second clustering($C^2$)
    $\lambda$: regularization parameter
Output: $C^1, C^2$: Two different clusterings
1. $C^1$ ←k-means($\mathcal{Z}$), $C^2$ ←Random assignment
2. $\mu_i$ ←ComputeMean($C^1_i$), $\nu_j$ ←ComputeMean($C^2_j$)
3. $a_i=\frac{1}{k_1}$, $b_j=\frac{1}{k_2}$
4. repeat
  E Step:
   4.1 For each $z$, assign $p_{ij}(z)$ using Eq. (7).
  M Step:
   4.2 Assign $a_i$, $b_j$ and $\sigma$ using (8), (9), (10).
   4.3 Assign $\alpha_i$ and $\beta_j$ using (13), (14).
   4.4 $\nu_j \leftarrow \beta_j$
   4.5 repeat until convergence
        • Update $\mu_i$ using (18).
        • Update $\nu_j$ using (19).
5. until convergence
6. $C^1_i = \{z | p_{ij}(z) = 1, \forall j\}$, $C^2_j = \{z | p_{ij}(z) = 1, \forall j\}$
return $C^1$, $C^2$
```

PROOF 3: We prove this claim using principle of mathematical induction. μ_i is initialized to 0, hence μ_i is a linear combination of $\phi(z_l)$. ν_j is intialized using β_j and since mean of a cluster is a linear combination of input points within the cluster, ν_j is also a linear combination of input points. Thus the base case holds.

Let hypothesis hold for iteration iter $= t$, then updates at iteration $t + 1$ are given by:

$$\mu_i^{(t+1)} = \left(I - \xi_i V^T Q (I + \xi_i \Sigma)^{-1} Q^T V^T\right) \left(\alpha_i - \frac{\sum_j n_{ij} \nu_j^{(t)}}{\sum_j n_{ij}}\right), \qquad (22)$$

where, $\xi_i = \frac{\lambda}{\sum_j n_{ij}}$ and $V = [\nu_1^t, \ldots, \nu_{k_2}^t]$ and $Q \Sigma Q^T$ is the eigenvalue decomposition of $V^T V$.

Using induction hypothesis, ν_j^t is a linear combination of the input points. And, $\left(\alpha_i - \frac{\sum_j n_{ij} \nu_j^{(t)}}{\sum_j n_{ij}}\right)$ is again a linear combination of the input points. Thus, $V^T \left(\alpha_i - \frac{\sum_j n_{ij} \nu_j^{(t)}}{\sum_j n_{ij}}\right)$ can be calculated efficiently using Eq. (20). Hence, μ_i^{t+1} is a linear combination of the input points. Similarly it can be shown that ν_j^{t+1} is a linear combination of the input points.

Now for given $\mu_r^t + \nu_s^t$ and $\sigma^2 = c$, consider the following probability density function:

$$\mathcal{N}\left(\mu_r^t + \nu_s^t, 1\right)(z) = \frac{1}{\sqrt{2\pi}} \exp\left(-\|z - \mu_r - \nu_s\|^2/\sigma^2\right) \qquad (23)$$

Note that $\|z - \mu_r - \nu_s\|^2 = K(z, z) + K(\mu_r + \nu_s, \mu_r + \nu_s) - 2K(z, \mu_r + \nu_s)$. Now using Claim 1, $\mu_r^t + \nu_s^t$ is a linear combination of the input points. Thus using Eq. (20), $\mathcal{N}\left(\mu_r^t + \nu_s^t, 1\right)(z)$ can be computed efficiently. This in turn implies that the E-step of Algorithm 2 can be computed efficiently in the kernel space. Thus Algorithm 2 can be kernelized efficiently.

5. DISCUSSION

5.1. Decorrelation Measure

Now we motivate the decorrelation measures used in Eqs. (1) and (12). For this, we will need the following two lemmas about uniqueness of projection and multivariate Gaussians. In the following lemmas, for a subspace S of $\mathbb{R}^m$ let $P_S : \mathbb{R}^m \to \mathbb{R}^m$ be the orthogonal projection (OP) operator onto the subspace S.

LEMMA 2: Let S_1, S_2 be subspaces of $\mathbb{R}^m$ such that $S_1 \cap S_2 = \{\mathbf{0}\}$. Then, for all $x \in S_1$, and $y \in S_2$, there exists a unique $u \in S_1 + S_2$ such that $P_{S_1}(u) = x$ and $P_{S_2}(u) = y$.

PROOF 4: Let $x \in S_1$ and $y \in S_2$. Also, let P_1, P_2 be the projection matrices for the projection operators P_{S_1} and P_{S_2} respectively. We first formulate the hypothesis that $S_1 \cap S_2 = \{\mathbf{0}\}$ in terms of the matrices P_1, P_2 by showing that $I - P_1P_2$ and $I - P_2P_1$ are invertible. Suppose, on the contrary that $I - P_1P_2$ is not invertible. Then, for some nonzero z we must have, $(I - P_1P_2)z = 0$, i.e., $z = P_1P_2z$. Recall that for a projection matrix P into a subspace S we always have $\|Pu\| \le \|u\|$ with equality if and only if $u \in S$. Thus, we have

$$\|z\| = \|P_1P_2z\| \le \|P_2z\| \le \|z\|.$$

Therefore, $z = P_1P_2z = P_2z$, which is possible only if $z \in S_1$ and $z \in S_2$. This contradicts the assumption that $S_1 \cap S_2 = \{\mathbf{0}\}$, $I - P_1P_2$ must be invertible. Similarly, we can also show that $I - P_2P_1$ is invertible.

Now, to prove the lemma we need to show that there exists a unique $u \in S_1 + S_2$ such that "$x = P_1u$ and $y = P_2u$". Since, $S_1 \cap S_2 = \{\mathbf{0}\}$, solving the above system of equations is equivalent to solving for $v \in S_1$, and $w \in S_2$ such that

$$x = P_1(v + w), \quad y = P_2(v + w).$$

Manipulating the above equations, we get:

$$(I - P_1P_2)v = x - P_1y, \quad (I - P_2P_1)w = y - P_2x.$$

The existence and uniqueness of v, w follow from the fact that $I - P_1P_2$ and $I - P_2P_1$ are invertible. ■

LEMMA 3: Let $Z \in \mathbb{R}^m$ denote a random variable with spherical Gaussian distribution. Let $S_1, S_2 \subseteq \mathbb{R}^m$ be two subspaces such that $S_1 \cap S_2 = \{0\}$ and let $Z_1 = P_{S_1}(Z)$, $Z_2 = P_{S_2}(Z)$ be the random variables obtained by projecting Z onto S_1, S_2 respectively. Then, the random variables Z_1 and Z_2 are independent if and only if the subspaces S_1 and S_2 are orthogonal.

PROOF 5: $\Longleftarrow$ If S_1 and S_2 are orthogonal, then for $u_1 \in S_1$ and $u_2 \in S_2$, $Pr[Z = u_1 + u_2] = Pr[Z_1 = u_1, Z_2 = u_2]$. Further, since Z has a spherical Gaussian distribution so do Z_1 and Z_2. The independence of Z_1 and Z_2 follows easily from the above observations.

$\Longrightarrow$ Let the random variables Z_1 and Z_2 be independent. Note that without loss of generality we can assume that Z has mean $\mathbf{0}$ (as else we can translate Z). Furthermore, we can also assume that the support of Z is contained in $S_1 + S_2$. This is because, $P_{S_1} = P_{S_1} \circ P_{S_1+S_2}$ and $P_{S_1+S_2}(Z)$ is also distributed as a spherical multivariate Gaussian. For the rest of the proof, we will suppose that $S_1 + S_2 = support(Z) = \mathbb{R}^m$ and that Z has mean $\mathbf{0}$.

Using Lemma 2, for $u \in \mathbb{R}^m$, we have

$$Pr[Z = u] = Pr[P_{S_1}(Z) = P_{S_1}(u), P_{S_2}(Z) = P_{S_2}(u)].$$

As Z_1 and Z_2 are independent, the above can be rewritten as

$$\begin{aligned} Pr[Z = u] = Pr[P_{S_1}(Z) = P_{S_1}(u)] \\ \cdot Pr[P_{S_2}(Z) = P_{S_2}(u)]. \end{aligned}$$

Now, since the projection of a spherical multivariate Gaussian is also a spherical multivariate Guassian, substituting probability density formulae in the above equation we get the following:

$$\frac{1}{(2\pi)^{m/2}} e^{-\frac{1}{2}\|u\|^2} = \frac{1}{(2\pi)^{m_1/2}} e^{-\frac{1}{2}\|u_1\|^2} \cdot \frac{1}{(2\pi)^{m_2/2}} e^{-\frac{1}{2}\|u_2\|^2},$$

where, m_1, m_2 denote the dimensions of S_1, S_2 respectively and $u_1 = P_{S_1}(u), u_2 = P_{S_2}(u)$. Noting that $m = m_1 + m_2$ (since $S_1 \cap S_2 = \{\mathbf{0}\}$) the above equation can be simplified to

$$\|u\|^2 = \|P_{S_1}(u)\|^2 + \|P_{S_2}(u)\|^2.$$

As the above equation holds for all u it also holds in particular for $u \in S_1$. Now, for $u \in S_1$ we have $P_{S_1}(u) = u$, thus we get

$$\forall u \in S_1, P_{S_2}(u) = 0.$$

The above condition can easily be shown to be equivalent to S_1 and S_2 being orthogonal. ■

We now give the motivation for our decorrelation measures. Let $\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_{k_1}$ and $\mathbf{v}_1, \ldots, \mathbf{v}_{k_2}$ be vectors in $\mathbb{R}^m$ such that $\boldsymbol{\mu}_i$ and $\mathbf{v}_j$ are orthogonal for all i, j. Let S_1 be the space spanned by $\boldsymbol{\mu}_i$s and S_2 be the space spanned by $\mathbf{v}_j$s. Define the "nearest-neighbor" random variables, $NN_1(Z), NN_2(Z)$ as follows:

$$\begin{aligned} NN_1(Z) &= \operatorname{argmin}\{\|Z - \boldsymbol{\mu}_i\| : 1 \le i \le k_1\}, \\ NN_2(Z) &= \operatorname{argmin}\{\|Z - \mathbf{v}_j\| : 1 \le j \le k_2\}. \end{aligned} \quad (24)$$

Then as S_1 and S_2 are orthogonal to each other, it follows from Lemma 3 that when Z is a spherical multivariate Gaussian, the random variables $NN_1(Z)$ and $NN_2(Z)$ are independent. Similarly, it can be shown that when Z is a spherical multivariate Gaussian, the random variables $\overline{NN}_1$, and $\overline{NN}_2$ defined by,

$$(\overline{NN}_1(Z), \overline{NN}_2(Z)) = \underset{(i,j)}{\operatorname{argmin}}\{\|Z - \boldsymbol{\mu}_i - \boldsymbol{\nu}_j\|\}, \qquad (25)$$

are independent. Note that in Eqs. (1) and (12) we use inner products involving the mean vectors of different clusterings as the correlation measure. Thus, minimizing the correlation measure ideally leads to the mean vectors of different clusterings being orthogonal. Also, observe that we use nearest-neighbor assignments of the form Eqs. (24) and (25) in our algorithms in Decorrelated k-means and Convolutional-EM. Thus, the decorrelation measures specified in Eqs. (1) and (12) intuitively correspond to the labelings of the clusterings being independent.

5.2. Decorrelated k-means versus Convolutional-EM

Decorrelated k-means (Algorithm 1) has a three-fold advantage over the "sum of the parts" approach (Algorithm 2):

- Computing the E-step exactly in the "sum of the parts" approach requires O(k^T) computation for each data point, where T is the number of alternative clusterings. On the other hand, in Decorrelated k-means, each label assignment step requires just O(kT) computations as the error terms for different clusterings are independent in Eq. (1). Thus, Decorrelated k-means is more scalable than Convolutional-EM with respect to the number of alternative clusterings.
- The M-step in the "sum of the parts" approach solves a nonconvex problem and requires an iterative procedure to reach a local minimum. In the decorrelated k-means approach, computing the representative vectors (the equivalent of M-step) requires solving a convex problem and the optimal solution can be written in closed form. Hence, estimation of representative vectors is more accurate and efficient for decorrelated k-means.
- Decorrelated k-means is a discriminative approach, while Convolutional-EM is a generative model based approach. Thus, the model assumptions are more stringent for the latter approach. This is observed empirically also, where Decorrelated k-means works well for all the datasets, but Convolutional-EM suffers on one of the real-life datasets.

On the flip side, there is no natural interpretation of the "representative" vectors given by decorrelated k-means. On the other hand, the means given by Convolutional-EM can naturally be interpreted as giving a part-based representation of the data. This argument is illustrated by Fig. 3. The representative vectors obtained from decorrelated k-means

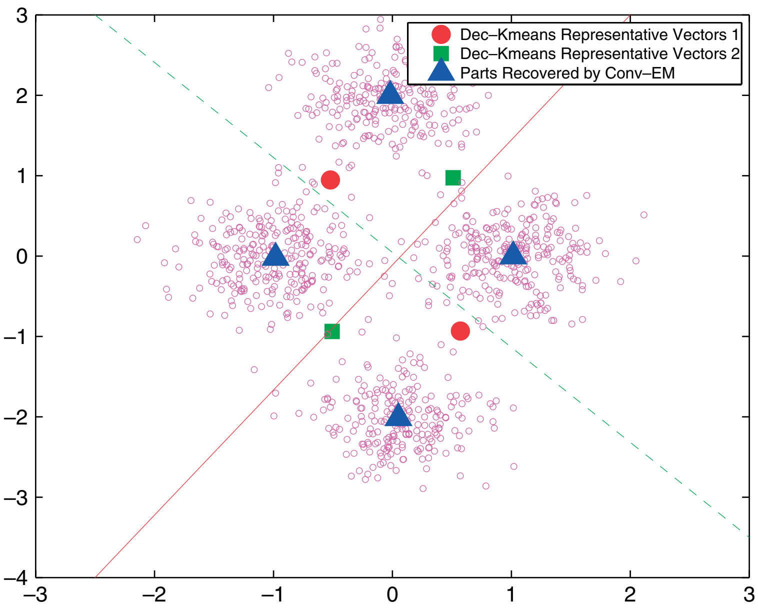


Fig. 3 Representative vectors obtained by Dec. k-means and the parts obtained by Conv-EM. The bold line represents the separating hyperplane for the first clustering, while the dotted line represents the separating hyperplane for the second clustering. Conv-EM produced mean vectors $\{\mu_1, \mu_2, \nu_1, \nu_2\}$ and subsequently each of the four parts are obtained by $\mu_i + \nu_j$ ($i \in \{1, 2\}, j \in \{1, 2\}$).

partition the data into two clusters accurately. But, they do not give any intuitive characterization of the data. In contrast, Convolutional-EM is able to recover the four clusters in the data generated by the addition of two mixtures of Gaussians.

6. EXPERIMENTS

We now provide experimental results on synthetic as well as real-world datasets to show the applicability of our methods. For real-world datasets, we consider a music dataset from the text-mining domain and a portrait dataset from the computer-vision domain. We compare our methods against the factorial-learning algorithms CVQ [8] and MCVQ [16]. We also compare against single-clustering algorithms such as k-means and NNMA. We will refer to the methods of Sections 3.1 and 3.2 as Dec. k-means (for Decorrelated k-means) and Conv-EM (for Convolutional-EM) respectively.

We also compare our methods against two simple heuristics:

1. *Feature Removal (FR)*: In this approach, we first cluster the data using k-means. Then, we remove the coordinates that have the most *correlation* with the labels in the obtained clustering. Next, we cluster the data again using the remaining features to obtain the alternative clustering. The correlation between a feature and the labels is taken to be proportional to the total weight of the mean vectors for the feature and inversely proportional to the entropy of the particular feature in the mean vectors. Formally:

$$C(i) = \frac{\sum_j \mu_j^i}{-\sum_j \left(\frac{\mu_j^i}{\sum_l \mu_l^i} \log \frac{\mu_j^i}{\sum_l \mu_l^i} \right)},$$

   where μ_j^i is the i-th dimension of the j-th cluster.
2. *Orthogonal Projection (OP)*: This heuristic is motivated by principal gene shaving [12]. The heuristic proceeds by projecting the data onto the subspace orthogonal to the means of the first clustering and uses the projected data for computing the second clustering.

   (a) Cluster the data using a suitable method of clustering.

   (b) Let the means of the obtained clustering be $\boldsymbol{m}_1, \ldots, \boldsymbol{m}_k$. Project the input matrix X onto the space orthogonal to the one spanned by the means $\boldsymbol{m}_1, \ldots, \boldsymbol{m}_k$ to get X'.

   (c) Cluster the columns of X' to obtain a new set of labels, and compute the cluster means $\tilde{\boldsymbol{m}}_1, \ldots, \tilde{\boldsymbol{m}}_k$.

   (d) Repeat steps (b),(c) with means $\tilde{\boldsymbol{m}}_1, \ldots, \tilde{\boldsymbol{m}}_k$.

   (e) Until convergence, repeat steps (a)-(d).

6.1. Implementation Details

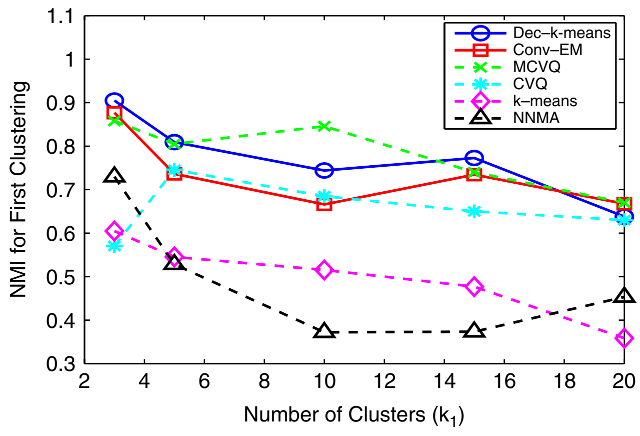
All the methods have been implemented in MATLAB. The implementation of MCVQ was obtained from the authors of [16]. Lee and Seung's algorithm [15] is used for NNMA. Experiments were performed on a Linux machine with a 2.4 GHz Pentium IV processor and 1 GB main memory. For the real-world datasets, we report results in terms of accuracy with the true labels. As the number of clusters can be high in the synthetic datasets, we report results in terms of normalized mutual information (NMI) [19]. For all the experiments, accuracy/NMI is averaged over 100 runs.

6.2. Synthetic Datasets

For our experiments we generate synthetic datasets as a sum of independent components. Let $\mathcal{X}$ and $\mathcal{Y}$ be samples drawn from two independent mixtures of multivariate Gaussians. To evaluate our methods in various settings, we generate the final dataset $\mathcal{Z}$ by combining $\mathcal{X}$ and $\mathcal{Y}$ in three different ways. By viewing $\mathcal{X}$ and $\mathcal{Y}$ as the *components* of the datasets, and clustering based on these components we get two different clusterings of the data.

1. Concatenated dataset: This dataset is produced by simply concatenating the features of $\mathcal{X}$ and $\mathcal{Y}$, i.e., $\mathcal{Z} = \begin{bmatrix} \mathcal{X} \\ \mathcal{Y} \end{bmatrix}$.
2. Partial overlap dataset: In this dataset we allow a few of the features of $\mathcal{X}$ and $\mathcal{Y}$ to overlap. Specifically, let $\mathcal{X} = \begin{bmatrix} \mathcal{X}_1 \\ \mathcal{X}_2 \end{bmatrix}$ and $\mathcal{Y} = \begin{bmatrix} \mathcal{Y}_1 \\ \mathcal{Y}_2 \end{bmatrix}$, where $\mathcal{X}_1$, $\mathcal{X}_2$, $\mathcal{Y}_1$ and $\mathcal{Y}_2$ all have the same dimensionality. Then, we form $\mathcal{Z} = \begin{bmatrix} \mathcal{X}_1 \\ \mathcal{X}_2 + \mathcal{Y}_1 \\ \mathcal{Y}_2 \end{bmatrix}$.
3. Sum dataset: In this dataset, all of the features of $\mathcal{X}$ and $\mathcal{Y}$ overlap, i.e., $\mathcal{Z} = \mathcal{X} + \mathcal{Y}$.

In our experiments, the dimensionality of $\mathcal{X}$ and $\mathcal{Y}$ was set to 30 and there were 3000 data points. We label each x_i and y_j according to the Gaussian from which they were sampled. Thus, each z is associated with two true labels. Both our methods produce two disparate clusterings, and we associate each clustering with a unique true-labeling and report NMI with respect to that true-labeling. We use



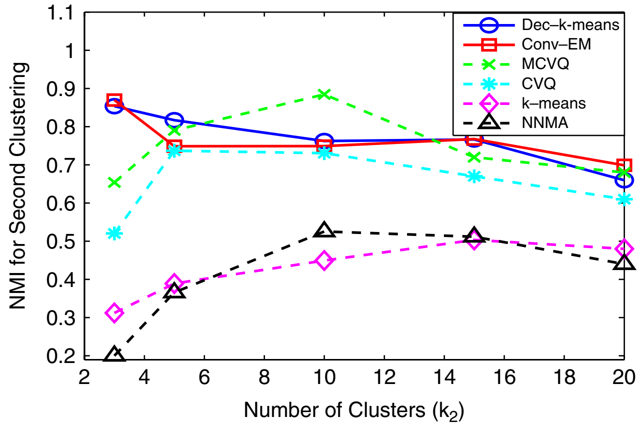


Fig. 4 NMI achieved by various methods on the concatenated dataset. Top figure shows NMI for the first clustering and bottom figure shows NMI for the second clustering. Overall, Dec. k-means achieves the highest NMI, while MCVQ also performs well on this dataset.

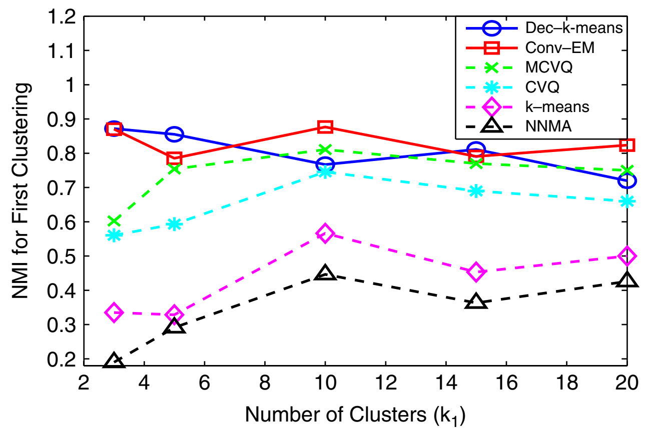
the same procedure for CVQ and MCVQ. For *k*-means and NNMA[2], which produce just one clustering, we report the NMI of the clustering with respect to the true-labelings.

Figure 4 compares the NMI achieved by various methods on the Concatenated dataset. It can be seen from the figure that Conv-EM and Dec. k-means outperform *k*-means and NNMA for both the clusterings, achieving an average improvement of $50-60\%$ in NMI. Similarly, both Conv-EM and Dec. k-means achieve significantly higher NMI than CVQ. Note that the Concatenated dataset satisfies MCVQ's assumption that each dimension of the data is generated from one of the two factors. This is empirically confirmed by the results, as MCVQ not only outperforms CVQ but also performs competitively with Conv-EM and Dec. k-means.

[2] As NNMA is useful for the nonnegative data only, we made the data nonnegative by choosing the means sufficiently far away from origin.

Figure 5 compares the NMI for various methods on the Overlap dataset. Clearly, Conv-EM and Dec. k-means perform better than both CVQ and MCVQ. Note that when the number of clusters is small, NMI of MCVQ with respect to both the clusterings drops to around 0.6. This is probably because the Overlap dataset does not satisfy the model assumptions of MCVQ.

Figure 6 shows the NMI achieved by various methods on the Sum dataset. For this dataset also, both Conv-EM and Dec. k-means perform comparably and both the methods achieve significantly higher NMI than other methods. Interestingly, NMI for MCVQ is even lower than the single-clustering algorithms (*k*-means and NNMA). This could be because the modeling assumption of MCVQ—each dimension in the data is generated by exactly one factor—is completely violated in the Sum dataset. Note that, although CVQ is designed to model the Sum datasets, it performs poorly compared to Conv-EM and Dec. k-means. This trend



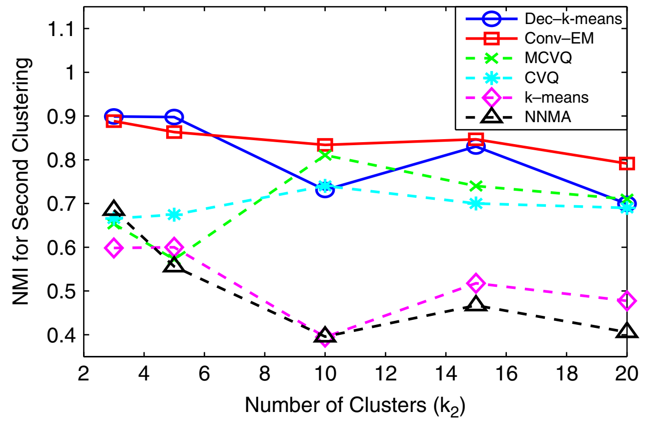


Fig. 5 NMI achieved by various methods on the overlap dataset. Top figure shows NMI for the first clustering and bottom figure shows NMI for the second clustering. Dec.k-means and Conv-EM achieves similar NMI. Both achieve higher NMI than MCVQ or CVQ.

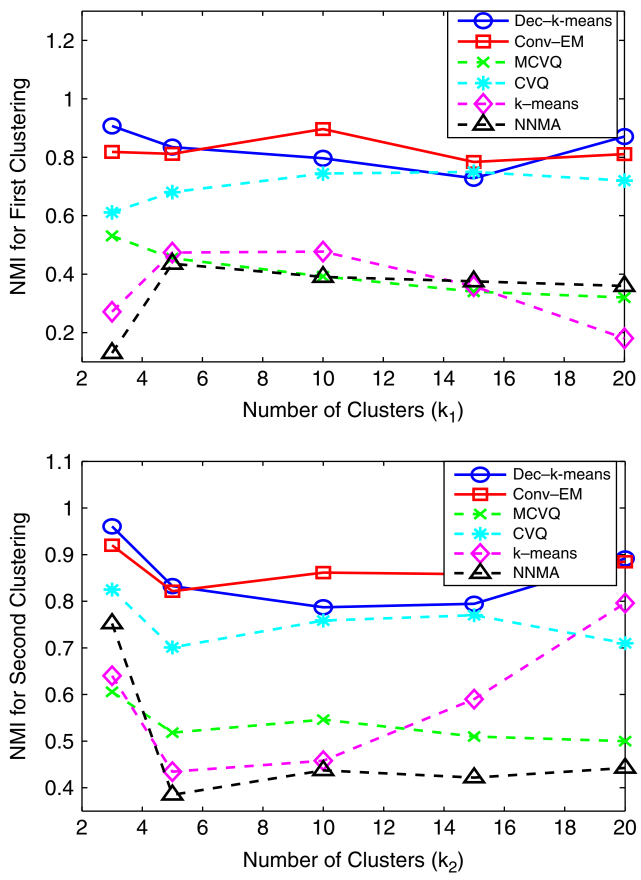


Fig. 6 NMI achieved by various methods on the sum dataset. Top figure shows NMI for the first clustering and bottom figure shows NMI for the second clustering. Dec.k-means and Conv-EM achieve similar NMI. NMI achieved by MCVQ is very low.

can be attributed to the fact that due to the lack of regularization CVQ selects one of the many possible solutions to its optimization problem, which may or may not correspond to good disparate clusterings.

Also note that Conv-EM does not perform significantly better than Dec. k-means, even though the datasets fit the Conv-EM model well. This is probably because of the nonconvex nature of the optimization problem for the M-step in Conv-EM, due to which the maximum likelihood estimation gets stuck in a local minimum.

6.3. Real-world Datasets

6.3.1. Music dataset

The music dataset is a collection of 270 documents, with each document being a review of a classical music piece taken from *amazon.com*. Each music piece is composed by one of Beethoven, Mozart or Mendelssohn and is in one of symphony, sonata or concerto forms. Thus, the documents can be clustered based on the composer or the genre of the musical piece. For the experiments, a term-document matrix was formed with dimensionality 258 after stop word removal and stemming.

Table 1 shows that although all the methods are able to recover the true clustering for composer, most of the algorithms perform poorly for the genre-based clustering. In particular, k-means and NNMA perform very poorly for the clustering based on genre as they produce just one clustering, which has high NMI with the clustering based on composers. Note that in this dataset, the sets of features (words) that determine clustering with respect to composer and genre respectively are almost disjoint. Hence, methods like FR and OP, which try to identify disjoint sets of features for the clusterings work fairly well. But, both MCVQ and CVQ algorithms achieve very low accuracy as they do not try to find *decorrelated* clusterings. Both our methods outperform the baseline algorithms.

6.3.2. Portrait dataset

The portrait dataset consists of 324 images obtained from Yale face dataset B [7]. Each image in the dataset is a portrait of one of three people, in one of three poses in different backgrounds. The dimensionality of each image is 64 × 64. As a first step, we reduce the dimensionality of the data to 300 by using principal component analysis. As in the music dataset, the current dataset can be clustered in two natural ways—by the person in the picture or the pose. Table 2 shows that both k-means and NNMA perform poorly with respect to both the clusterings. This shows that in the datasets where there is more than one natural clustering, traditional clustering algorithms could fail to find even one good clustering. Hence, it can be beneficial to use alternative clustering methods even if one is interested in obtaining a single clustering.

Our hypothesis is that unlike the music dataset, there are no dominant features for any of the clusterings in this dataset. This hypothesis can be justified by observing the poor accuracies of methods like FR and OP. Conv-EM

Table 1. Accuracy achieved by various methods on the music dataset, which is a collection of text documents. Dec. k-means performs the best on this dataset. CVQ and MCVQ perform very poorly compared to Conv-EM.

Method\Type	Composer	Genre
NNMA	1.00	0.40
k-Means	0.89	0.41
Feature removal	0.97	0.64
Orthogonal projection	0.99	0.66
CVQ	0.97	0.57
MCVQ	0.91	0.53
Conv-EM	1.00	0.65
Dec. k-means	**1.00**	**0.69**

Table 2. Accuracy achieved by various methods on the portrait dataset, which is a collection of images. Dec.k-means outperforms all other methods by a significant margin. Conv-EM achieves better accuracy than all other methods, especially, CVQ and MCVQ.

Method\Type	Person	Pose
NNMA	0.51	0.49
k-means	0.66	0.56
Feature Removal	0.56	0.48
Orthogonal projection	0.66	0.70
CVQ	0.53	0.51
MCVQ	0.64	0.51
Conv-EM	0.69	0.72
Dec.k-means	**0.84**	**0.78**

outperforms baseline algorithms CVQ and MCVQ significantly, but interestingly Dec. k-means achieves an even higher accuracy of 84 and 78% for the two clusterings.

7. CONCLUSIONS AND FUTURE WORK

We address the difficult problem of uncovering disparate clusterings from the data in a totally unsupervised setting. We present two novel approaches for the problem—a decorrelated k-means approach and a sum of parts approach. In the first approach, we introduce a new regularization for k-means to uncover decorrelated clusterings. We provide theoretical justification for using the proposed decorrelation measure. The sum of parts approach leads us to the interesting problem of learning a convolution of mixture models and we present a regularized EM algorithm for learning a convolution of mixtures of spherical Gaussians. We address the problem of identifiability for learning a convolution of mixtures of Gaussians by using a regularization geared for providing disparate clusterings. We demonstrate the effectiveness and robustness of our algorithms on synthetic and real-world datasets. On each of these datasets, we significantly improve upon the accuracy achieved by existing factorial-learning methods such as CVQ and MCVQ. Our methods also outperform the traditional clustering algorithms like k-means and NNMA.

For future work, it would be of interest to study the problem of learning a convolution of mixtures for a more general class of distributions and look for other settings where learning convolutions of mixtures could be useful. We also plan to further investigate the properties of the decorrelation measure, especially for more general distributions of the data. A problem that we do not address in this paper is model selection—choosing the number of clusters and the number of clusterings. Good heuristics for choosing these parameters would be very useful. Also, a more detailed comparison of Conv-EM and Dec. k-means would be useful; it would be interesting to understand the comparable performance of Conv-EM and Dec.k-means for the synthetic datasets that fit the convolution model well.

ACKNOWLEDGEMENTS

This research was supported by NSF grant CCF-0431257, NSF Career Award ACI-0093404, and NSF-ITR award IIS-0325116.

REFERENCES

[1] E. Bae and J. Bailey, COALA: A novel approach for the extraction of an alternate clustering of high quality and high dissimilarity, in ICDM, 2006, 53–62.

[2] M. Bilenko, S. Basu, and R. J. Mooney, Integrating constraints and metric learning in semi-supervised clustering, in ICML, 2004, 81–88.

[3] C. M. Bishop, Neural Networks for Pattern Recognition, Oxford, Oxford University Press, 1996.

[4] I. Davidson, S. S. Ravi, and M. Ester, Efficient incremental constrained clustering, in KDD, 2007, 240–249.

[5] R. O. Duda, P. E. Hart, and D. G. Stork, Pattern Classification, Wiley-Interscience, New York, 2000.

[6] W. R. Gaffey, A consistent estimator of a component of a convolution, Ann Math Stat 30 (1959), 198–205.

[7] A. Georghiades, P. Belhumeur, and D. Kriegman, From few to many: illumination cone models for face recognition under variable lighting and pose, IEEE Trans Pattern Anal Mach Intell 23 (2001), 643–660.

[8] Z. Ghahramani, Factorial learning and the EM algorithm, in NIPS, 1994, 617–624.

[9] G. H. Golub and C. F. van Loan, Matrix Computations, (2nd ed.), Johns Hopkins University Press, Baltimore, 1989.

[10] D. Gondek and T. Hofmann, Non-redundant data clustering, in ICDM, 2004, 75–82.

[11] D. Gondek, S. Vaithyanathan, and A. Garg, Clustering with model-level constraints, in SDM, 2005, 126–138.

[12] T. Hastie, R. Tibshirani, A. Eisen, R. Levy, L. Staudt, et al., Gene shaving as a method for identifying distinct sets of genes with similar expression patterns, Genome Biol (2000).

[13] G. E. Hinton and R. S. Zemel, Autoencoders, minimum description length and Helmholtz free energy, in NIPS, 1993, 3–10.

[14] A. Hyvärinen and E. Oja, Independent component analysis: algorithms and applications, Neural Netw 13 (2000), 411–430.

[15] D. D. Lee and H. S. Seung, Algorithms for non-negative matrix factorization, in NIPS, 2000, 556–562.

[16] D. A. Ross and R. S. Zemel, Learning parts-based representations of data, J Mach Learn Res 7 (2006), 2369–2397.

[17] F. J. Samaniego and L. E. Jones, Maximum likelihood estimation for a class of multinomial distributions arising in realibility, J R Stat Soc, Ser B (Methodological) 43 (1981), 46–52.

[18] S. L. Sclove and J. van Ryzin, Estimating the parameters of a convolution, J R Stat Soc, Ser B (Methodological) 31 (1969), 181–191.

[19] A. Strehl and J. Ghosh, Cluster ensembles—a knowledge reuse framework for combining multiple partitions, J Mach Learn Res 3 (2002), 583–617.
[20] J. B. Tenenbaum and W. T. Freeman, Separating style and content with bilinear models, Neural Comput 12 (2000), 1247–1283.
[21] K. Wagstaff and C. Cardie, Clustering with instance-level constraints, in ICML, 2000, 1103–1110.
[22] K. Wagstaff, C. Cardie, S. Rogers, and S. Schrödl, Constrained k-means clustering with background knowledge, in ICML, 2001, 577–584.
[23] E. P. Xing, A. Y. Ng, M. I. Jordan, and S. Russell, Distance metric learning, with application to clustering with side-information, in NIPS, vol. 15, 2003, 521–528.
[24] W. I. Zangwill, Nonlinear Programming: A Unified Approach, Prentice-Hall, Englewood Cliffs, 1969.