A Particular Gaussian Mixture Model for Clustering and its Application to Image Retrieval

Hichem Sahbi

Machine Intelligence Laboratory, Department of Engineering, Cambridge University, UK

and Certis Laboratory, Ecole Nationale des Ponts et Chaussees, France

hs385@cam.ac.uk, hichem.sahbi@enpc.fr

Abstract.
We introduce a new approach for data clustering based on Gaussian Mixture Models (GMMs). Each cluster of data, modeled as a GMM in an input space, is interpreted as a hyperplane in a high dimensional mapping space where the underlying coefficients are found by solving a quadratic programming problem (QP). The main contribution is an approach for GMM estimation which requires only finding the mixture parameters in contrast to other density estimation methods which are very sensitive to initialization. Furthermore, the introduced QP is solved easily using a new decomposition algorithm which involves trivial linear programming sub-problems. The validity of the method is demonstrated for clustering $2D$ toy examples as well as image databases.

Keywords: Gaussian Mixture Models, Clustering, Kernel Methods and Image Retrieval.

1 Introduction

Consider a training set $\mathcal{S}_N = \{x_1, ..., x_N\}$ generated i.i.d. according to an unknown but a fixed probability distribution $P(X)$. Here X is a random variable standing for training data in an *input space* denoted $\mathcal{X}$. Our goal is to define a partition of this training set, i.e., assign each training example to its actual cluster y_k, $k = 1, ..., C$. Existing clustering methods can be categorized depending on the "crispness" of data memberships, i.e., whether each training example belongs to only one cluster or not. The first category includes hierarchical approaches [1, 2], EM (Expectation Maximization) [3], C (or K) means [4], self organizing maps (SOMs) [5] and recently original approaches based on kernel methods [6, 7, 8]. In the second family the membership of a training example to one or another cluster is fuzzy and a family of algorithms dealing with fuzziness exists in the literature for instance [9, 10, 11, 12]. All these methods have been used in different domains including Gene expression [13], image segmentation [14] and database categorization [15].

One of the main issues in the existing clustering methods remains setting the appropriate number of classes for a given problem. Many methods for instance hierarchical clustering [16, 17, 2] has proven to perform well when the application allows us to know a priori the number of clusters or when the user sets it manually. Of course, the estimation of this number is application-dependent, for instance in image segmentation it can be set a priori to the number of targeted regions. Unfortunately, for some applications such as database categorization, it is not always possible to predict automatically and even manually the appropriate number of classes.

Given a training set $\mathcal{S}_N$ in the input space $\mathcal{X}$, in our method each cluster in $\mathcal{X}$ is modeled as a GMM and interpreted as a hyperplane in a high dimensional space referred to as the mapping space, denoted $\mathcal{H}$. Clustering consists in maximizing a likelihood function of data memberships and is equivalent to finding the parameters of the cluster hyperplanes by solving a QP problem. We will show that this QP can be tackled efficiently by solving trivial linear programming sub-problems. Notice that when solving this QP, the number of clusters (denoted C), fixed initially, might be overestimated and this leads to several overlapping clusters; therefore the actual clusters are found as constellations of highly correlated hyperplanes in the mapping space $\mathcal{H}$.

In the remainder of this paper, we refer to a cluster as a set of data gathered using an algorithm while a class (or a category) is the actual membership of this data according to a well defined ground truth. Among notations, i and j stand for data indices while k, c and l stand for cluster indices. Other notations will be introduced as we go along through different sections of this paper which is organized as following: In Section 2 we provide a short reminder on GMMs followed by our formulation in Section 3. In Section 4 we show that the clustering minimization problem can be solved as a succession of simple linear programming subproblems which are handled efficiently. We present in Section 5 experiments on simple as well as challenging problems in content based image retrieval. Finally, we conclude in Section 6 and we provide some directions for a future work.

2 A Short Reminder on GMMs

Given a training set $\mathcal{S}_N$ of size N, we are interested in a particular Gaussian Mixture Model (denoted $\mathcal{M}_k$) [18] where the number of its components is equal to the size of the training set. The parameters of $\mathcal{M}_k$ are denoted $\Theta_k = \{\Theta_{i|k},\ i = 1, ..., N\}$. Here $\Theta_{i|k}$ stands for the i^{th} component parameter of $\mathcal{M}_k$, i.e., the mean and the covariance matrices of a Gaussian density function. In this case, the output of the likelihood GMM function related to a cluster y_k is a weighted sum of N component densities:

$$p(x|y_k) \ = \ \sum_{i}^{N} \ P(\Theta_{i|k}|y_k)\, p(x|y_k, \Theta_{i|k}), \quad (1)$$

here $P(\Theta_{i|k}|y_k)$ (also denoted μ_{ik}) is the prior probability of the i^{th} component parameter $\Theta_{i|k}$ given the cluster y_k and $p(x|y_k, \Theta_{i|k}) \ = \ \mathcal{N}_k(x; \Theta_{i|k})$ is the normal density function of the i^{th} component. In order to guarantee that $p(x|y_k)$ is a density function, the mixture parameters $\{\mu_{ik}\}$ are chosen such that $\sum_i \mu_{ik} \ = \ 1$. Usually the training of GMMs can be formulated as a maximum likelihood problem where the parameters Θ_k and the mixture parameters $\{\mu_{ik}\}$ are estimated using expectation maximization [3, 18].

We consider in our work, $\mathcal{N}_k(x; \Theta_{i|k})$ with a *fixed* mean $x_i \ \in \ \mathcal{S}_N$ and covariance matrix $\sigma\ \Sigma_i$ ($\Sigma_i \ \in \ \mathbb{R}^{p\times p}$, $\sigma \ \in \ \mathbb{R}$). Now, each cluster y_k is modeled as a GMM where the *only* free parameters are the mixture coefficients $\{\mu_{ik}\}$; the means and the covariances are assumed constant but of course dependent on a priori knowledge of the training set (see; Figure 1 and Section 5).

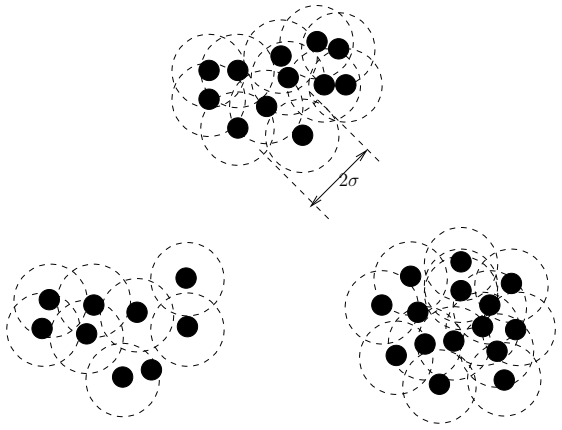

Figure 1: *This figure shows a particular GMM model where the centers and the variances are fixed. The only free parameters are the GMM mixture coefficients and the scale* σ.

3 Clustering

The goal of a clustering algorithm is to make clusters, containing data from different classes, as different as possible while keeping data from the same classes as close as possible to their *actual* clusters. Usually this implies the optimization of an objective function (see for instance [19]) involving a fidelity term which measures the fitness of each training sample to its model and a regularizer which reduces the number of clusters, i.e., the complexity of the model.

3.1 Our Formulation

Following notations and definitions in Section 2, a given $x \ \in \ \mathbb{R}^p$ is assigned to a cluster y_k if:

$$y_k \ = \ \arg\max_{y_l} \ p(x|y_l), \quad (2)$$

here the weights $\mu \ = \ \{\mu_{il}\}$, of the GMM functions $p(x|y_l)\ l \ = \ 1, ..., C$, are found by solving the following constrained minimization problem (see. motivation below):

$$\begin{aligned} \min_{\mu} \quad & \sum_{k,i} \mu_{ik} \left(\sum_{l\neq k} p(x_i|y_l) \right) \\ \text{s.t.} \quad & \sum_{i} \mu_{ic} \ = \ 1, \quad \mu_{ic} \in [0, 1], \quad i = 1, ..., N, \\ & c = 1, ..., C \end{aligned} \quad (3)$$

In the above objective function, the training data belonging to a cluster y_l are assumed drawn from a GMM with a likelihood function $p(x|y_l)$. Each mixture

parameter μ_{il} stands for the degree of membership (or the contribution) of the training example x_i to the cluster y_l. The overall objective is to maximize the membership of each training example to its actual cluster while keeping the memberships to the remaining clusters relatively low. Usually, existing clustering methods (see for instance [9]) find the parameters $\{\mu_{il}\}$ as those which maximize the membership of each training example to its actual cluster. In contrast to these methods, our formulation proceeds using a dual principle; *the purpose is to minimize the memberships of training examples to their* **non-actual** *clusters*.

Using (1), we can expand the objective function (3) as:

$$\min_{\mu} \sum_{k \neq l} \sum_{i,j} \mu_{ik}\, \mu_{jl}\, \mathcal{N}_l(x_i; \Theta_{j|l}), \tag{4}$$

here $\mathcal{N}_l(x_i; \Theta_{j|k})$ is the response of a Gaussian density function (also referred to as kernel), with fixed parameters $\Theta_{j|k} = \{x_j, \sigma\ \Sigma_j\}$; x_j is the mean, Σ_j is the covariance and σ is the scale. We will denote this kernel simply as $K_\sigma(\|x_i - x_j\|)$. It is known that the Gaussian kernel is positive definite [20] so this function corresponds to a scalar product in the mapping space $\mathcal{H}$, i.e., there is a mapping Φ_σ from the input space into an infinite dimensional space such that $K_\sigma(\|x_i - x_j\|) = \langle \Phi_\sigma(x_i), \Phi_\sigma(x_j) \rangle$ where $\langle\rangle$ stands for the inner product in $\mathcal{H}$. At this stage, the response of a GMM function $p(x|y_k)$ is equal to the scalar product $\langle \omega_k, \Phi_\sigma(x) \rangle$ where $\omega_k = \sum_i \mu_{ik}\, \Phi_\sigma(x_i)$ is the normal of a hyperplane in $\mathcal{H}$ (see; Figure 2). Now, the objective function (4) can be rewritten:

$$\min_{\mu} \sum_{k \neq l} \langle \omega_k, \omega_l \rangle \tag{5}$$

The above objective function minimizes the sum of hyperplane *correlations* taken pairwise among all different clusters. Now, we can derive the new form of the constrained minimization problem (3):

$$\begin{aligned} \min_{\mu} \quad & \sum_{k,i} \sum_{l \neq k,j} \mu_{ik}\, \mu_{jl}\, K_\sigma(\|x_i - x_j\|) \\ \text{s.t.} \quad & \sum_i \mu_{ic} = 1, \quad \mu_{ic} \in [0,1], \quad i = 1, ..., N, \\ & c = 1, ..., C \end{aligned} \tag{6}$$

This defines a constrained QP which can be solved using standard QP libraries (see for instance [21]). When solving this problem, training examples $\{x_i\}$ for which the mixing parameter $\{\mu_{ik}\}$ are positive will be referred to as the *GMM vectors* of the cluster y_k (see; Figure 2).

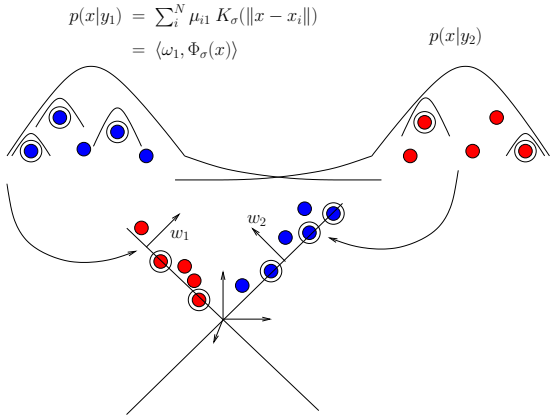

Figure 2: *This figure shows the mapping of training samples into a high dimensional space. Data in the original space characterizes Gaussian blobs while in the mapping space they correspond to hyperplanes. The GMM vectors are surrounded with circles and correspond to the centers of the Gaussian kernels for which the mixture parameters do not vanish.*

4 Training

The number of parameters intervening in (6) is $N \times C$, so for clustering problems of reasonable size, for instance $N = 1.000$ and $C = 20$, solving this QP, using standard packages, can quickly get out of hand. Chunking methods have been successfully used to solve QP for large scale training problems such as SVM [22]. The idea consists in solving a QP problem using an active subset of parameters, referred to as a *chunk*, which is updated iteratively. When the QP is convex, by checking that the Gram matrix is positive definite, the process is guaranteed to converge to the global optimum after a sufficient number of iterations [22].

Using the same principle as [22, 23], we will show in this section that for a particular choice of the active chunk, the QP (6) can be decomposed into linear programming subproblems each one can be solved trivially.

4.1 Decomposition

Let us fix one cluster index $p \in \{1, ..., C\}$ and rewrite the objective function (6) as:

$$\begin{aligned} \min_{\mu} \quad & 2 \sum_i \mu_{\mathbf{ip}}\, c_{ip} \\ & + \sum_{i,k \neq p} \sum_{j,l \neq k,p} \mu_{ik}\, \mu_{jl}\, K_\sigma(\|x_i - x_j\|), \end{aligned} \tag{7}$$

here:

$$c_{ip} = \sum_{j,l \neq p} \mu_{jl}\, K_\sigma(\|x_i - x_j\|) \qquad (8)$$

Consider a chunk $\{\mu_{ip}\}_{i=1}^N$ and fix $\{\mu_{jk}, k \neq p\}_{j=1}^N$, the objective function (7) is linear in terms of $\{\mu_{ip}\}$ and can be solved, with respect to this chunk, using linear programming. Only one equality constraint $\sum_i \mu_{ip} = 1$ is taken into account in the linear programming problem as the other equality constraints in (6) are independent from the chunk $\{\mu_{ip}\}$.

We can further reduce the size of the chunk $\{\mu_{ip}\}_{i=1}^N$ to only two parameters. Given $i_1, i_2 \in \{1, ..., N\}$, we can rewrite (7) as:

$$\begin{array}{ll} \min\limits_{\mu_{i_1 p},\ \mu_{i_2 p}} & 2\,(\boldsymbol{\mu_{i_1 p}}\, c_{i_1 p} + \boldsymbol{\mu_{i_2 p}}\, c_{i_2 p}) + b \\ \mathbf{s.t} & \mu_{i_1 p} + \mu_{i_2 p} = 1 - \sum\limits_{i \neq i_1, i_2} \mu_{ip} \\ & 0 \leq \mu_{i_1 p} \leq 1 \\ & 0 \leq \mu_{i_2 p} \leq 1, \end{array} \qquad (9)$$

here b is a constant independent from the chunk $\{\mu_{i_1 p}, \mu_{i_2 p}\}$ (see; 7). When $c_{i_1 p} = c_{i_2 p}$ the above objective function and the equality constraints are linearly dependent, so we have an infinite set of optimal solutions; any *one* which satisfies the constraints in (9) can be considered. Let us assume $c_{i_1 p} \neq c_{i_2 p}$ and denote $d = 1 - \sum_{i \neq i_1, i_2} \mu_{ip}$, since $\mu_{i_2 p} = d - \mu_{i_1 p}$, the above linear programming problem can be written as:

$$\begin{array}{lll} \min\limits_{\mu_{i_1 p}} & & \boldsymbol{\mu_{i_1 p}}\, (c_{i_1 p} - c_{i_2 p}) \\ \mathbf{s.t} & \max(d-1, 0) & \leq \mu_{i_1 p} \leq \min(d, 1) \end{array} \qquad (10)$$

Depending on the sign of $c_{i_1 p} - c_{i_2 p}$, the solution of (10) is simply taken as one of the bounds of the inequality constraint in (10). At this stage, the parameters $\{\mu_{1p}, ..., \mu_{Np}\}$ which solve (7) are found by solving iteratively the trivial minimization problem (10) for different chunks, as shown in **Algorithm1**,

Algorithm1(p, μ)

Do the following steps **ITERMAX1** iterations
Select randomly $(i_1, i_2) \in \{1, ..., N\}^2$.
$(c_{i_1 p}, c_{i_2 p}) \longleftarrow$ (8).
if $(c_{i_1 p} - c_{i_2 p} < 0)$ $\mu_{i_1 p} \longleftarrow \min(d, 1)$ (see; 10)
else $\mu_{i_1 p} \longleftarrow \max(d-1, 0)$
$\mu_{i_2 p} \longleftarrow d - \mu_{i_1 p}$ (see; 9)
endDo
return $(\mu_{i_1 p}, ..., \mu_{i_N p})$

and the whole QP (6) is solved by looping several times through different cluster indices as shown in **Algorithm2**.

Algorithm2

Set $\{\mu\}$ to random values.
Do the following steps **ITERMAX2** iterations
Fix p in $\{1, ..., C\}$, update μ_{1p},...,μ_{Np} using:
$(\mu_{1p}, ..., \mu_{Np}) \longleftarrow$ Algorithm1(p, μ)
endDo

4.2 Agglomeration

Initially, the algorithm described above considers an over-estimated number of clusters (denoted C). This will result into several overlapping cluster GMMs which might be detected using a similarity measure for instance the Kullback Leibler divergence [18].

Given two cluster GMMs $p(x|y_k)$, $p(x|y_l)$, for a fixed $\epsilon > 0$ we declare $p(x|y_k)$, $p(x|y_l)$ as similar if:

$$\int_{\mathcal{X}} \|p(x|y_k) - p(x|y_l)\|^2\, dx < \epsilon \qquad (11)$$

As $p(x|y_k) = \langle \omega_k, \Phi_\sigma(x) \rangle$ and $p(x|y_l) = \langle \omega_l, \Phi_\sigma(x) \rangle$, we can simply rewrite (11) as:

$$\int_{\mathcal{X}} \langle \omega_k - \omega_l, \Phi_\sigma(x) \rangle^2\, dx < \epsilon \qquad (12)$$

We can set the memberships in ω_k, ω_l such that $\|\omega_k\| = \|\omega_l\| = 1$. Now, from (12) it is clear that two overlapping GMMs correspond to two highly correlated hyperplanes in the mapping space, i.e., $\langle \omega_k, \omega_l \rangle \geq \tau$ (for a fixed threshold $\tau \in [0, 1]$).

Regularization: notice that,

$$\frac{\partial \langle\, \omega_k, \omega_l \rangle}{\partial \sigma} = 2 \sum_{i,j} \mu_{ik}\, \mu_{jl}\, \frac{K_\sigma(\|x_i - x_j\|)}{\sigma^3} \geq 0, \qquad (13)$$

so the correlation is a convex increasing function of the scale σ. Assuming $\|w_k\| = \|w_l\| = 1$, it results that $\forall\, \tau \in [0, 1]$, $\exists\, \sigma$ such that $\langle \omega_k, \omega_l \rangle \geq \tau$.
Let us define the following adjacency matrix:

$$A_{k,l} = \begin{cases} 1 & \text{if } \langle \omega_k, \omega_l \rangle \geq \tau \\ 0 & \text{otherwise} \end{cases} \qquad (14)$$

This matrix characterizes a graph where a node is a hyperplane and an arc corresponds to two highly correlated hyperplanes (i.e., $\langle \omega_k, \omega_l \rangle \geq \tau$). Now, the actual

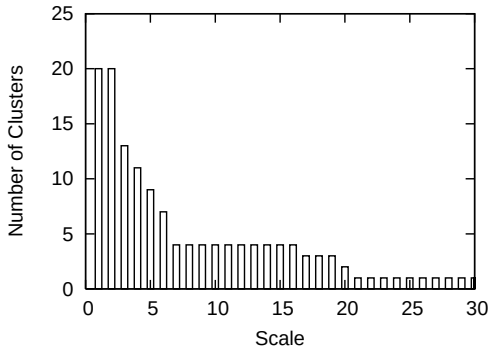

Figure 3: *This figure shows the decrease of the number of clusters with respect to the scale parameter σ ($N = 100$, $C = 20$). These experiments are performed on the training samples shown in Figure (4).*

number of clusters is found as the number of connected components in this graph. For a fixed threshold τ, the scale parameter σ acts as a *regularizer*. When σ is overestimated, the graph $A = \{A_{k,l}\}$ will be connected resulting into only one big cluster whereas a small σ results into lots of disconnected subgraphs, therefore the total number of clusters will be large (see; Figure 3).

4.3 Mixing Different Scales

We can extend the formulation presented in (3) to handle clusters with large variation in scale. Let us denote $\sigma(y_l)$ the scale of the GMM related to the cluster y_l. We can rewrite the objective function (6) as:

$$\begin{aligned} \min_{\mu} \quad & \sum_{k,i} \sum_{l \neq k,j} \mu_{ik}\, \mu_{jl}\, K_{\sigma(y_l)}(\|x_i - x_j\|) \\ \text{s.t.} \quad & \sum_{i} \mu_{ic} = 1, \quad \mu_{ic} \in [0,1], \quad i = 1, ..., N, \\ & c = 1, ..., C \end{aligned} \tag{15}$$

Following the same steps (see. Section 4.1), we can solve (15). However, the above objective function cannot be interpreted as the minimization of pairwise correlations between hyperplanes since the GMMs, with different scale parameters, correspond to hyperplanes living in different mapping spaces. It follows that, the criteria (in 14) used to detect and merge overlapping GMMs cannot be used. Instead, other criteria, for instance the Kullback Leibler divergence, might be used.

4.4 Toy Examples

These experiments are targeted to show the performance brought by this decomposition algorithm both in term of precision and speed. We consider a two dimensional training set, of size N, generated using four

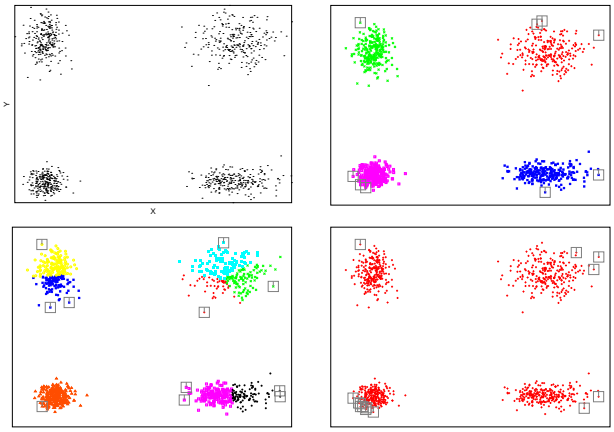

Figure 4: *(Top-left) Data randomly generated from four Gaussian densities centered at four different locations ($N = 1000$). When running Algorithm2, C is fixed to 20, so the total number of parameters in the training problem is 20×1000. (Other Figures) Different clusters are shown with different colors and the underlying GMM vectors are shown in gray. In these experiments σ is set respectively from top to bottom-right to 10, 4 and 50.*

Gaussian densities centered at four different locations (see; Figure 4, top-left). We cluster these data using **Algorithm2**, for different values of N (40, 80, 100, 500, 1000). For $N = 100$, table (1) shows all the pairwise correlations between the hyperplanes found (i) when running **Algorithm2** and (ii) when solving the whole QP (6) using a standard package LOQO [21]. We can see that each hyperplane found using **Algorithm2** is highly correlated with *only* one hyperplane found using LOQO. Table (2) shows a comparison of the underlying runtime performances using a 1Ghz Pentium III.

Table 1: *This table shows the correlations (normalized inner products) between the hyperplanes found, when (i) running* **Algorithm2** *and (ii) solving the QP (6) using the LOQO package, on the 2D toy data of Figure (4). ($\sigma = 10$).*

LOQO pack vs. Algorithm2	Cluster 1	Cluster 2	Cluster 3	Cluster 4
Cluster 1	**0.994**	0.043	0.331	0.187
Cluster 2	0.036	**0.999**	0.238	0.133
Cluster 3	0.284	0.313	**0.985**	0.058
Cluster 4	0.152	0.147	0.057	**0.998**

Table 2: *Comparison of the run-time performances on the training samples in Figure (4) for different values of* N. $(C = 20, \sigma = 10)$.

$\#\mathcal{S}$ (N)	40	80	100	500
# of parameters in the QP $(N \times C)$	280	480	700	3500
LOQO pack	24.6 (s)	87.9 (s)	317.8(s)	> 1 (H)
Algorithm2	0.09 (s)	0.32 (s)	0.56 (s)	24.99 (s)

5 Database Categorization

Experiments have been conducted on both the Olivetti and Columbia databases in order to show the good performance of our clustering method. The Olivetti subset contains 20 persons each one represented by 10 faces. The Columbia subset contains 15 categories of objects each one represented by 72 images. Each image from both the Olivetti and Columbia datasets is processed using histogram equalization and encoded using 20 coefficients of projection into a subspace learned using ISOMAP [24]. Notice that ISOMAP embeds a training database into a subspace such that non-linearly separable classes become more separable, homogeneous in terms of scale, and easier to cluster.

5.1 Scale

It might be easier for database categorization to predict the variance of data rather than the number of clusters, mainly for large databases living in high dimensional spaces. As the number of clusters is initially set to an overestimated value (see; Section 4.2), our method relies mainly on one parameter, i.e., the scale σ, which makes it possible to find automatically the actual number of clusters. In our experiments, we predict σ by sampling manually few images from some categories, estimating the scales of the underlying classes, then setting σ to the expectation of the scale through these classes. While this setting is not automatic, it has at least the advantage of introducing a priori knowledge on the variance of the data at the expense of few interactions and reasonable effort from the user.

Figure (5) shows that this heuristic provides "good guess" of the scale on the Olivetti and Columbia sets as it is close to the optimal scale shown in Figure (6). Indeed, the estimated scale in Figure (5) is approximately 24 and 0.4 for respectively Olivetti and Columbia sets. For these scales, the number of clusters found by our algorithm on Olivetti and Columbia (resp. 19 and 15, see; Figure 6, top) is close to the actual numbers, resp. 20 and 15. Furthermore, the probability of error (17)

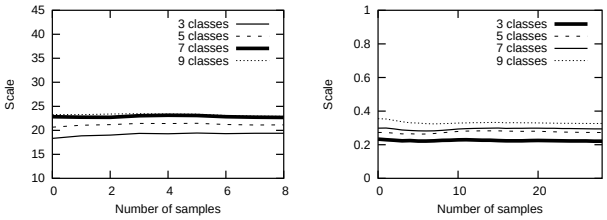

Figure 5: *These diagrams show the variation of the scale with respect to the number of samples per class on (left) the Olivetti and (right) Columbia sets. The scale is given as the expectation of the distance between pairs of images taken from* 3, 5, 7 *and* 9 *classes.*

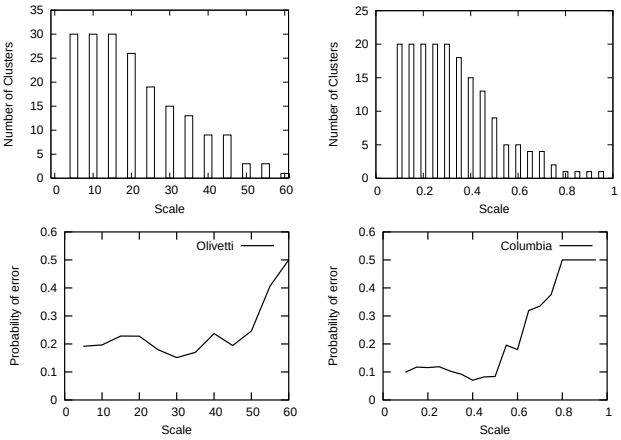

Figure 6: *(Top) Decrease of the number of clusters with respect to the scale parameter. (Bottom) Probability of error with respect to* σ. *These observations are given using (left) the Olivetti and (right) the Columbia sets.*

is close to its minimum for both Olivetti and Columbia (see; Figure 6, bottom).

5.2 Precision

A Clustering method can be *objectively* evaluated when the ground truth is available, otherwise the meaning of clustering can differ from one intelligent observer to another. The validity criteria are introduced in order to measure the quality of a clustering algorithm, i.e., its capacity to assign data to their actual classes. For a survey on these methods, see for example [25].

In the presence of a ground truth, we consider in our work a simple validity criteria based on the probability of misclassification. The latter occurs when either two examples belonging to two different classes are assigned to the same cluster, or when two elements belonging to the same class are assigned different clusters. We denote X and Y as two random variables stand-

Figure 7: *This figure shows* 18 *face prototypes, from different clusters, found after the application of our method. Each prototype corresponds to a GMM vector.*

ing respectively for the training examples and their different possible classes $\{y_1, ..., y_C\}$ and X', Y' respectively similar to X, Y. We denote by $f(X)$ the index of the cluster of X in $\{y_1, ..., y_C\}$. Formally, we define the misclassification error as:

$$P(\mathbb{1}_{\{(f(X) = f(X')\}} \neq \mathbb{1}_{\{Y = Y'\}}) = (*), \qquad (16)$$

here:

$$\begin{aligned}(*) \quad &= \quad P(f(X) \neq f(X') \mid Y = Y')\, P(Y = Y') \\ &+ \quad P(f(X) = f(X') \mid Y \neq Y')\, P(Y \neq Y') \end{aligned} \qquad (17)$$

Again, Figure (6, bottom) shows the misclassification error (17) with respect to the scale parameter σ. Tables (3, 4) are the confusion matrices which show the distribution of 20 and 15 categories from respectively the Olivetti and the Columbia sets through the clusters after the application of our clustering method. We can see that this distribution is concentrated in the diagonal and this clearly shows that most of the training data are assigned to their actual categories (see; Figures 7,8).

Figure 8: *This figure shows five clusters from the Olivetti database found after the application of our algorithm.*

Table 3: *This table shows the distribution of the categories through the clusters on the Olivetti set using our clustering algorithm. We can see that this distribution is concentrated near the diagonal. The scale parameter* σ *is set to* 24 *and* $C = 30$*. Errors in the cardinality of the clusters are mentioned in red.*

categories / clusters	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	total
1:	9	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	9
2:	.	10	.	.	1	3	.	.	.	1	.	.	.	.	.	.	.	.	3	.	18
3:	.	.	10	.	.	.	.	.	.	.	.	.	.	.	.	.	.	5	.	.	15
4:	.	.	.	10	.	.	.	.	3	.	.	.	.	.	.	.	2	.	.	.	15
5:	.	.	.	.	9	.	.	.	.	.	.	.	.	.	.	.	.	.	.	1	10
6:	.	.	.	.	.	.	5	.	.	.	.	.	.	.	.	.	.	.	.	.	5
7:	.	.	.	.	.	.	.	.	5	.	2	.	.	.	.	.	.	.	.	.	7
8:	.	.	.	.	.	1	.	.	.	9	.	.	.	.	.	.	.	.	.	2	12
9:	.	.	.	.	.	.	.	.	.	.	3	.	.	.	.	.	.	.	.	.	3
10:	1	.	.	.	.	.	.	.	.	.	.	4	5	.	2	.	.	.	.	.	12
11:	.	.	.	.	.	.	.	.	.	.	5	.	.	.	.	.	.	.	.	.	5
12:	.	.	.	.	.	.	.	.	.	.	.	6	3	.	.	.	.	.	.	.	9
13:	.	.	.	.	.	.	.	.	.	.	.	.	2	10	.	1	.	.	.	.	13
14:	.	.	.	.	.	1	.	.	.	.	.	.	.	.	8	.	.	.	.	.	9
15:	.	.	.	.	.	1	.	.	.	.	.	.	.	.	.	9	.	.	.	.	10
16:	.	.	.	.	.	4	1	2	2	.	.	.	.	.	.	.	8	2	.	.	19
17:	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	3	.	5	8
18:	.	.	.	.	.	.	4	.	.	.	.	.	.	.	.	.	.	.	7	.	11
19:	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	.	2	2
total	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	10	

Table 4: *This table shows the distribution of the categories through the clusters on the Columbia set using our clustering algorithm. We can see that this distribution is concentrated near the diagonal. The scale* σ *is set to* 0.4 *and* $C = 20$*. Errors in the cardinality of the clusters are mentioned in red.*

categories / clusters	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	total
1:	72	.	.	.	.	.	.	.	.	.	.	.	.	.	.	72
2:	.	72	.	.	.	.	.	.	.	.	.	.	.	.	.	72
3:	.	.	72	.	.	.	.	.	.	.	.	.	.	.	.	72
4:	.	.	.	72	.	.	.	.	.	.	.	.	.	.	.	72
5:	.	.	.	.	72	.	.	.	.	.	.	.	.	.	.	72
6:	.	.	.	.	.	37	.	34	.	.	.	.	.	.	.	71
7:	.	.	.	.	.	35	.	38	.	.	.	.	.	.	.	73
8:	.	.	.	.	.	.	72	.	.	.	.	.	.	.	.	72
9:	.	.	.	.	.	.	.	.	72	.	.	.	.	.	.	72
10:	.	.	.	.	.	.	.	.	.	72	.	.	.	.	.	72
11:	.	.	.	.	.	.	.	.	.	.	72	.	.	.	.	72
12:	.	.	.	.	.	.	.	.	.	.	.	72	.	.	.	72
13:	.	.	.	.	.	.	.	.	.	.	.	.	30	.	.	30
14:	.	.	.	.	.	.	.	.	.	.	.	.	42	72	72	186
total	72	72	72	72	72	72	72	72	72	72	72	72	72	72	72	

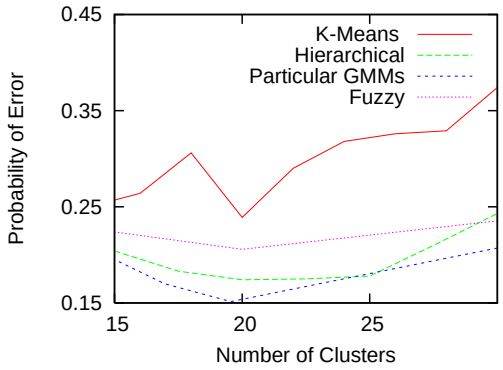

Figure 9: *This figure shows a comparison of the generalization error on the Olivetti set of our method and other existing state of the art methods including fuzzy, kmeans and hierarchical clustering.*

5.3 Comparison

Figure (9) shows a comparison of our kernel clustering methods with respect to existing state of the art approaches including fuzzy clustering [12], Kmeans [4] and hierarchical clustering [2]. These results are shown for the Olivetti database. The error (17) is measured with respect to the number of clusters. Notice that the latter is fixed for hierarchical clustering and Kmeans while it corresponds to the resulting number of clusters after setting σ (see; Section 4.2) for fuzzy clustering and our method. We see clearly that the generalization error takes its smallest value when the number of clusters is close to the actual one (i.e., 20).

6 Conclusion and Future Work

We introduced in this work an original approach for clustering based on a particular Gaussian Mixture Models (GMMs). The method considers an objective function which acts as a regularizer and minimizes the overlap between the cluster GMMs. The GMM parameters are found by solving a quadratic programming problem using a new decomposition algorithm which considers trivial linear programming subproblems. The actual number of clusters is found by controlling the scale parameter of these GMMs; in practice, it turns out that predicting this parameter is easier than predicting the actual number of clusters mainly for large databases living in high dimensional spaces.

The concept presented in this paper is different from kernel regression which might be assimilated to density estimation while our approach performs this estimation for each cluster. Obviously, the proposed approach performs clustering and density estimation at the same time.

The validity of the method is demonstrated on toy data as well as database categorization problems. As a future work, we will investigate the application of the method to handle noisy data.

References

[1] A.K. Jain and R. C. Dubes, "Algorithms for clustering data. prentice hall," 1988.

[2] C. Posse, "Hierarchical model-based clustering for large datasets," *Journal of Computational and Graphical Statistics*, vol. 10, no. 3, pp. 464–486, 2001.

[3] A. Dempster, N. Laird, and D Rubin, "Maximum likelihood from incomplete data via the em algorithm," *Journal of Royal Statistical Society. Ser. B*, vol. 39, no. 1, pp. 1–38, 1977.

[4] J. MacQueen, "Some methods for classi cation and analysis of multivariate observations," *in Proceedings of the 5th Berkeley Symposium on Mathematical Statistics and Probability*, vol. 1, 1965.

[5] E.K. Tsao, J.C. Bezdek, and N.R. Pal, "Fuzzy kohonen clustering networks," *PR*, vol. 27, no. 5, pp. 757–764, May 1994.

[6] A. Ben-Hur, D. Horn, H. T. Siegelmann, and V. Vapnik, "Support vector clustering," *In Neural Information Processing Systems*, pp. 367–373, 2000.

[7] F.R. Bach and M.I. Jordan, "Learning spectral clustering," *Neural Information Processing Systems*, 2003.

[8] M. Wu and B. Schoelkopf, "A local learning approach for clustering," *In Avances Neural Information Processing Systems*, 2006.

[9] J.C. Bezdek, "Pattern recognition with fuzzy objective function algorithms. new york," 1981.

[10] H. Frigui and R. Krishnapuram, "A robust competitive clustering algorithm with applications in computer vision," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 21, no. 5, pp. 450–465, 1999.

[11] R.N. Dave, "Characterization and detection of noise in clustering," *In Pattern Recognition*, vol. 12, no. 11, pp. 657–664, 1991.

[12] H. Sahbi and N. Boujemaa, "Validity of fuzzy clustering using entropy regularization," *In proceedings of the IEEE conference on Fuzzy Systems*, 2005.

[13] C.A. Orengo, D.T. Jones, and J.M. Thornton, "Bioinformatics - genes, protein and computers. bios," *ISBN: 1-85996-054-5*, 2003.

[14] C. Carson, S. Belongie, H. Greenspan, and J. Malik, "Blobworld: Image segmentation using expectation-maximization and its application to image querying," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 8, pp. 1026 – 1038, 2002.

[15] B. Le-Saux and N. Boujemaa, "Unsupervised robust clustering for image database categorization," *IEEE-IAPR International Conference on Pattern Recognition*, pp. 259–262, 2002.

[16] P.H. Sneath and R.R. Sokal, "Numerical taxonomy- the principles and practice of numerical classification," *W. H. Freeman, San Francisco*, 1973.

[17] C. Fraley, "Algorithms for model-based gaussian hierarchical clustering," *SIAM Journal on Scientific Computing*, vol. 20, no. 1, pp. 270–281, 1998.

[18] C.M. Bishop, *Neural networks for pattern recognition*, Clarendon Press, Oxford, 1995.

[19] H. Frigui and R. Krishnapuram, "Clustering by competitive agglomeration," *In Pattern Recognition*, vol. 30, no. 7, 1997.

[20] N. Cristianini and J. Shawe-Taylor, "An introduction to support vector machines," *Cambridge University Press*, 2000.

[21] R. J. Vanderbei, "LOQO: An interior point code for quadratic programming," *Optimization Methods and Software*, vol. 11, pp. 451–484, 1999.

[22] E. Osuna, R. Freund, and F. Girosi, "Training support vector machines: an application to face detection," *In Proceedings of the International Conference on Computer Vision and Pattern Recognition*, pp. 130–136, 1997.

[23] J. Platt, "Fast training of support vector machines using sequential minimal optimization," *In B. Schlkopf and C. Burges and A. J. Smola editors. Advances in Kernel Methods — Support Vector Learning. Cambridge. MA. MIT Press*, pp. 185–208, 1999.

[24] J Tanenbaum, V de Silva, and J Langford, "A global geometric framework for non-linear dimensionality reduction," *Science*, vol. 290, no. 5500, pp. 2319–2323, 2000.

[25] M. Halkidi, Y. Batistakis, and M. Vazirgiannis, "Cluster validity methods: Part i and ii," *SIGMOD Record*, June 2002.