Levels of details for Gaussian mixture models

Vincent Garcia , Frank Nielsen, and Richard Nock

Introduction

Mixture models

A mixture model is a powerful framework to estimate probability density functions. A mixture model f is given by

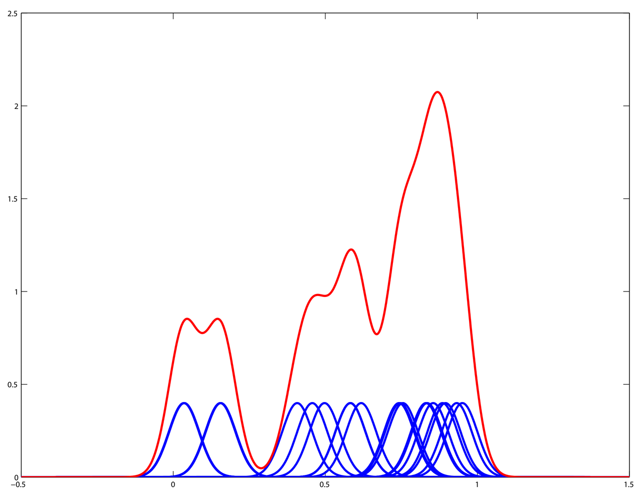
$$f(x) = \sum_{i=1}^{n} \alpha_i f_i(x) \tag{1}$$

where $\alpha_i \geq 0$ denotes a weight with $\sum_{i=1}^{n} \alpha_i = 1$. If f is a Gaussian mixture model (GMM),

$$f_i(x) = \frac{1}{(2\pi)^{d/2}|\Sigma_i|^{1/2}} \exp\left(-\frac{(x-\mu_i)^T\Sigma_i^{-1}(x-\mu_i)}{2}\right) \tag{2}$$

with μ_i mean and Σ_i covariance matrix.

Problem of using mixture models



Density estimation using kernel-based Parzen estimator

Mixture models usually contain a lot of components. The estimation of statistical measures is computationally expensive. We need to reduce the number of components in f:

- Re-lear a simpler mixture model from dataset (too long)
- Simplify the mixture model f (most appropriated solution)

Mixture model simplification

Given a mixture model f of n components (see Eq. (1)), the problem of mixture model simplification consits in computing a mixture model g of m components

$$g(x) = \sum_{j=1}^{m} \alpha'_j g_j(x)$$

such as g is the *best* approximation of f. The problem is how to simplify f and what is a good value for m? In this paper, we propose a new algorithm who allow to

1. simplifies a mixture of exponential family f,
2. learns the *optimal* number of components m in the simplified mixture,
3. provides a progressive representation of the mixture.

Mixture of exponential families

Kullback-Leibler divergence and Bregman divergence

The fundamental measure between statistical distributions is the Kullback-Leibler divergence (KLD). Given f_i and f_j two distributions, the KLD is a **sided similarity measure** given by

$$\mathrm{KLD}(f_i||f_j) = \int f_i(x) \log \frac{f_i(x)}{f_j(x)} \mathrm{d}x \tag{3}$$

In the case of normal distriubtions

$$\mathrm{KLD}(f_i||f_j) = \frac{1}{2}\log\left(\frac{\det \Sigma_j}{\det \Sigma_i}\right) + \frac{1}{2}\mathrm{tr}\left(\Sigma_j^{-1}\Sigma_i\right) + \frac{1}{2}(\mu_j-\mu_i)^T\Sigma_j^{-1}(\mu_j-\mu_i) - \frac{d}{2} \tag{4}$$

Nomral distributions belong to the class of exponential families. The canonical form is

$$f(x) = \exp\left\{\langle\tilde{\Theta}, t(x)\rangle - F(\tilde{\Theta}) + C(x)\right\} \tag{5}$$

The expressions of $\tilde{\Theta}$, $t(x)$, $F(\tilde{\Theta})$, and $C(x)$ for normal distributions are given in the article. The KLD between two Gaussians is equal to the Bregman divergence on natural parameters (inversion of the parameter order) and defined for the log normalizer F:

$$\mathrm{KLD}(f_i||f_j) = D_F(\tilde{\Theta}_j||\tilde{\Theta}_i) = F(\tilde{\Theta}_j) - F(\tilde{\Theta}_i) - \langle\tilde{\Theta}_j - \tilde{\Theta}_i, \nabla F(\tilde{\Theta}_i)\rangle \tag{6}$$

Using this formalism, we are able to propose algorithms working on the wide class of **mixtures of exponential families**. In particular, this class include the mixture of Gaussians.

Sided-centroids

Given a set of Gaussians (or exponential family member), it is possible to compute a centroid. This centroid can be sided (right-sided or left-sided) or symmetric.

Bregman G-means clustering

G-means clustering algorithm

Given a set of points, the G-Means algorithm (Hamerly and Elkan, NIPS 2003) recursively splits the initial set into subsets of points until each subset follows a normal distribution according to the Anderson-Darling test. The G-means algorithm first provides a hierarchical clustering of the input data, and second automatically learns the *optimal* number of clusters (parameter k). The proposed GMM simplification algorithm is based on the G-means algorithm.

Bregman G-means clustering algorithm

BGMC(N, f, c, α)

1: Store the centroid c and the weight α in the node N.
2: Draw a set of l points $X = \{x_1, \cdots, x_l\}$ from f.
3: Split the centroid c into two centroids c_1 and c_2.
4: Perform a Bregman k-means on c_1 and c_2. Let f_1 (resp. f_2) be the set containing the weighted Gaussians of f closer to c_1 (resp. c_2) than c_2 (resp. c_1). Let α_1 (resp. α_2) be the sum of all the weights of the Gaussians contained in f_1 (resp. f_2).
5: Compute the projection vector $v = \mu_1 - \mu_2$ where μ_1 and μ_2 are resp. the mean of c_1 and c_2.
6: Given X and v, use the Anderson-Darling statistical test to detect if f is a normal distribution (at confidence level $\beta = 0.95$).
7: **if** f is a normal distribution **then**
8: Stop the process; the current node N is a leaf (N_{left} and N_{right} are null).
9: **else**
10: Compute BGMC($N_{left}, f_1, c_1, \alpha_1$).
11: Compute BGMC($N_{right}, f_2, c_2, \alpha_2$).
12: **end if**

The algorithm starts with BGMC(N, f, c, α) where N is the root of an empty binary tree, f is a GMM, c is the centroid (right-sided, left-sided, or symmetric) of f, and $\alpha = 1$. N_{left} and N_{right} respectively denote the left-child and the right-child of the node N.

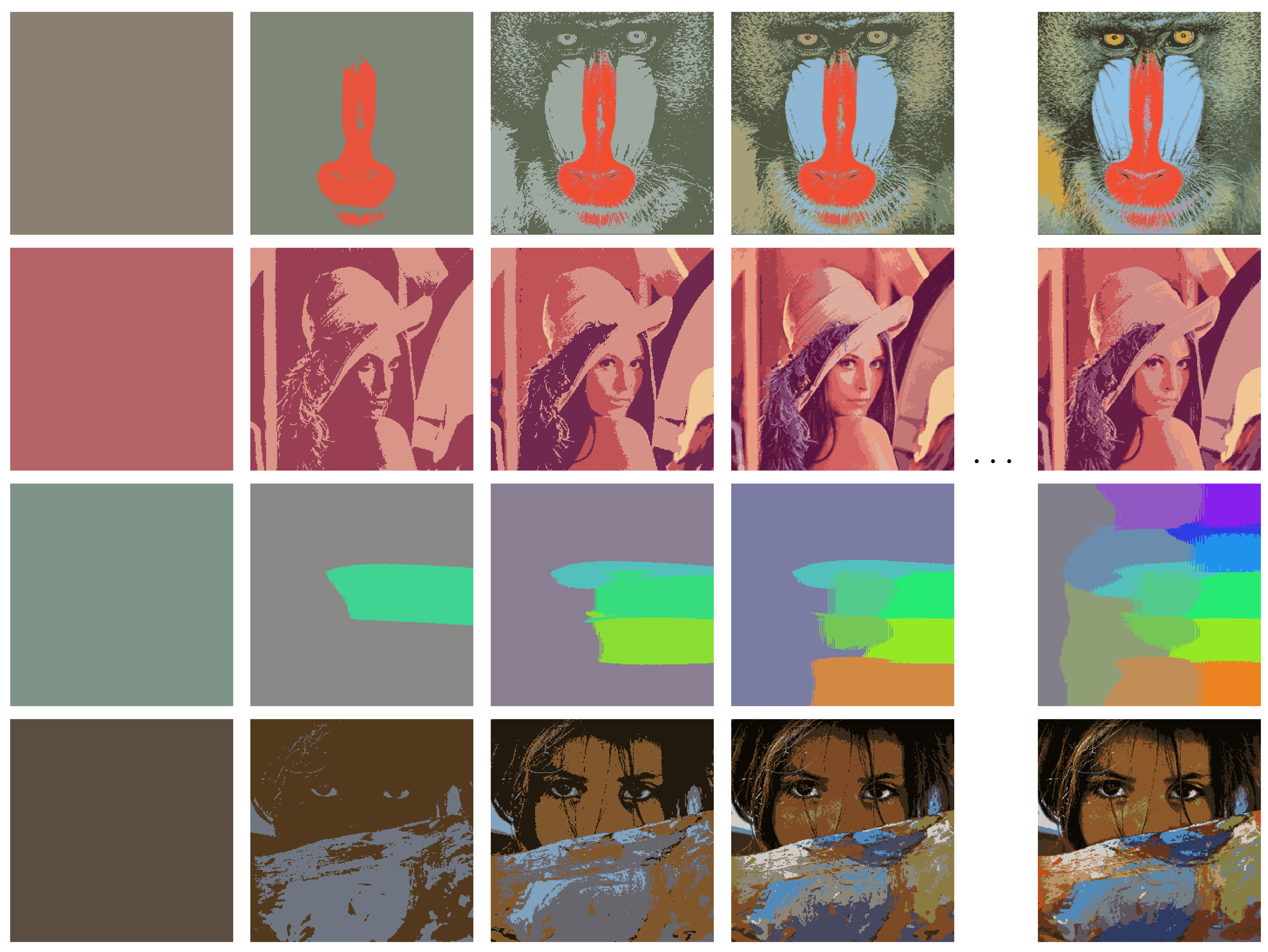
Resolution and automatic learning of m

G-means provides a hierarchical structure (binary tree) of the simplified GMM.

- Each node of the tree contains a weighted Gaussian.
- The resolution r corresponds to all the weighted Gaussians contained in nodes of depth r.
- The *optimal* value of m is given by the GMM size at the maximal resolution.

Experiments

- Application to clustering-based image segmentation.
- Simplification quality (estimated using Monte-Carlo based Kullback-Leibler divergence) and the visual quality increase with the rsolution.
- Automatic learning of *optimal* value of m:
  - Baboon: $m = 14$, max. res.=8, KLD=0.18
  - Lena: $m = 14$, max. res.=7, KLD=0.13
  - Colormap: $m = 14$, max. res.=9, KLD=0.59
  - Shantytown: $m = 13$, max. res.=5, KLD=0.28



Res. = 1 | Res. = 2 | Res. = 3 | Res. = 4 | Max. res.

Conclusion

In this paper, we have proposed an algorithm algorithm nammed BGMC:

- Simplification of Gaussian mixture model and mixture of exponential families.
- Automatic learning of the *optimal* of components m in the simplified mixture.
- Hierarchical representation of the mixture.

jMEF: Java library for Mixture of Exponential Families available on-line at: *www.lix.polytechnique.fr/~nielsen/MEF*