Signal Processing 93 (2013) 1531–1546

Contents lists available at SciVerse ScienceDirect

Signal Processing

journal homepage: www.elsevier.com/locate/sigpro

Simultaneous Bayesian clustering and feature selection using RJMCMC-based learning of finite generalized Dirichlet mixture models

Tarek Elguebaly[a], Nizar Bouguila[b,*]

[a] Electrical and Computer Engineering Department, Faculty of Engineering and Computer Science, Concordia University, Montreal, Qc, Canada H3G 2W1

[b] Concordia Institute for Information Systems Engineering, Faculty of Engineering and Computer Science, Concordia University, Montreal, Qc, Canada H3G 2W1

ARTICLE INFO

Article history:
Received 2 March 2012
Received in revised form
19 July 2012
Accepted 21 July 2012
Available online 18 August 2012

Keywords:
Clustering
Finite mixture models
Generalized Dirichlet
Feature selection
Bayesian analysis
RJMCMC
Gibbs sampling
Metropolis–Hastings
Video categorization
Pedestrian detection
Face recognition

ABSTRACT

Selecting relevant features in multidimensional data is important in several pattern analysis and image processing applications. The goal of this paper is to propose a Bayesian approach for identifying clusters of proportional data based on the selection of relevant features. More specifically, we consider the problem of selecting relevant features in unsupervised settings when generalized Dirichlet mixture models are considered to model and cluster proportional data. The learning of the proposed statistical model, to formulate the unsupervised feature selection problem, is carried out using a powerful reversible jump Markov chain Monte Carlo (RJMCMC) technique. Experiments involving the challenging problems of human action videos categorization, pedestrian detection and face recognition indicate that the proposed approach is efficient.

1. Introduction

Larger and larger multimedia data are collected and stored everyday presenting enormous opportunities and challenges by increasing the need for efficient modeling, knowledge extraction, categorization and clustering approaches [1,2]. A fundamental task in many of these approaches is the learning of appropriate statistical models. Mixture models are now among the most widely used statistical approaches in many areas of application and allow a formal approach for unsupervised learning [3]. In such context, classic interest is often in the determination of the number of clusters (i.e. model selection) and estimation of the mixture's parameters (see, for instance, [4,5]). Another essential issue in the case of statistical modeling in general and finite mixtures in particular is feature selection (i.e. identification of the relevant or discriminative features describing the data) especially in the case of high-dimensional data which analysis has been the topic of extensive research in the past [6–8]. Indeed, feature selection has been shown to be a crucial step in several image processing, computer vision and pattern recognition applications such as object detection [9,10], handwriting separation [11], image retrieval, categorization and recognition [12,13]. Feature selection is a major concern not only because it speeds up learning but also because it improves model accuracy and generalization. Moreover, the learning of the

* Corresponding author.
E-mail addresses: t_elgue@encs.concordia.ca (T. Elguebaly), bouguila@ciise.concordia.ca, nizar.bouguila@concordia.ca (N. Bouguila).

0165-1684/$ - see front matter © 2012 Elsevier B.V. All rights reserved.
http://dx.doi.org/10.1016/j.sigpro.2012.07.037

mixture parameters (i.e. both model selection and parameters estimation) is greatly affected by the quality of the features used as shown for instance in [14] which has given renewed attention to the feature selection problem especially in unsupervised settings. Like many other model-based feature selection approaches (see, for instance, [15,16]) this work has been based on the Gaussian assumption by assuming diagonal covariance matrices [16] for all clusters (i.e. all the features are assumed independent). Although the normality assumption has been taken for granted in general, several works have shown that this assumption is not sound and not realistic in several applications [17,18] since per-class distributions of real data often deviate from the Gaussian distribution. This is especially true in the case of proportional data which arise in many fields of application and for which the generalized Dirichlet (GD) mixture has been shown to be an efficient modeling choice as clearly shown, for instance, in [19,5,13]. In particular, the authors in [13] have proposed a mixture-based feature selection approach relying on GD distribution and benefiting from its interesting mathematical properties and flexibility.

The unsupervised feature selection model in [13] has been trained using a minimum message length (MML) [20] objective function with the expectation–maximization (EM) [21] algorithm. This algorithm is, however, prone to initialization errors and converges either to a local maximum or to a saddle point solution which may compromise the modeling capabilities. More recently, with the increase on power of computing resources, a number of more advanced approaches based on Bayesian inference have been proposed, along with associated learning algorithms to design them in order to overcome drawbacks related with the EM framework [22]. Indeed, Bayesian inference is enjoying increasing attention in the statistical learning literature and allows to avoid over-fitting and suboptimal generalization performance. The main idea is to consider an ensemble of models described by a probability distribution over all possible parameter values, rather than considering a single model as in EM-based learning, in order to take into account model uncertainty [23]. The implementation of Bayesian inference-based approaches is based on MCMC approximations (e.g. Gibbs sampler, Metropolis–Hastings) which are important tools for statistical inference in signal and image processing applications especially in non-Gaussian settings [24,25].

The aim of this paper is to extend the unsupervised feature selection approach previously proposed in [13] by reformulating it within a fully Bayesian framework. We present a learning algorithm based on RJMCMC technique [26] which has been widely studied and applied since its introduction by offering an alternative to MCMC that includes the automatic determination of the appropriate number of mixture components and the quantification of the uncertainty (see, for instance, [27–31]). We are mainly motivated by the good results obtained recently using Bayesian learning techniques in machine learning applications in general and for the unsupervised feature selection problem in particular [15]. The remainder of the paper is structured as follows: In Section 2 we review the feature selection model based on the generalized Dirichlet mixture. In Section 3 we propose a Bayesian extension to this model and we present the complete learning algorithm where inference on the parameters is made by constructing a Gibbs sampling technique. In Section 4 we provide experimental results and we conclude with a discussion and a summary of the work in Section 5.

2. The unsupervised feature selection model

In this section, we briefly describe the unsupervised feature selection approach based on the finite GD mixture model. Although this paper is self-contained, we refer the interested reader to [13] for detailed discussions and analysis.

2.1. The mixture model

Mixture models have recently drawn a great deal of interest, being recognized as a powerful framework for probabilistic inference. One of the cited advantages of finite mixture models is that they allow for principled methods for reasoning with incomplete data. Let $\{\vec{X}_1,\ldots,\vec{X}_N\}$ be an unlabeled dataset where each vector $\vec{X}_i$ is composed of a set of continuous features representing a given object (e.g. image, video, document, etc.). Here we assume that each vector follows a mixture of generalized Dirichlet distributions of which each generalized Dirichlet has parameters θ_j, $j=1,\ldots,M$ and the mixing weights, which are positive and sum to one, of the different components are $\vec{P}=(p_1,\ldots,p_M)$:

$$p(\vec{X}_i|\Theta_M)=\sum_{j=1}^{M}p_j p(\vec{X}_i|\theta_j) \tag{1}$$

where M is the number of components which determines the structure of the model, $\Theta_M=(\vec{P},\vec{\theta})$, $\vec{\theta}=(\theta_1,\ldots,\theta_M)$ and $p(\vec{X}_i|\theta_j)$ are the components distributions which we take as generalized Dirichlet. In dimension D, the generalized Dirichlet density is defined by

$$p(\vec{X}_i|\theta_j)=\prod_{d=1}^{D}\frac{\Gamma(\alpha_{jd}+\beta_{jd})}{\Gamma(\alpha_{jd})\Gamma(\beta_{jd})}X_{id}^{\alpha_{jd}-1}\left(1-\sum_{l=1}^{d}X_{il}\right)^{\gamma_{jd}} \tag{2}$$

where $\theta_j=(\vec{\alpha}_j,\vec{\beta}_j)$, $\vec{\alpha}_j=(\alpha_{j1},\ldots,\alpha_{jD})$ and $\vec{\beta}_j=(\beta_{j1},\ldots,\beta_{jD})$; $\sum_{d=1}^{D}X_{id}<1$ and $0<X_{id}<1$ for $d=1,\ldots,D$; $\gamma_{jd}=\beta_{jd}-\alpha_{jd+1}-\beta_{jd+1}$ for $d=1,\ldots,D-1$ and $\gamma_{jD}=\beta_{jD}-1$.

The Generalized Dirichlet distribution has an interesting property, previously shown in [13]. Indeed, if a vector $\vec{X}_i$ has a generalized Dirichlet distribution with parameters $(\vec{\alpha}_j,\vec{\beta}_j)$, then we can construct a vector $\vec{Y}_i$ using the following geometric transformation: $Y_{i1}=X_{i1}$ and $Y_{id}=X_{id}/(1-X_{i1}-\cdots-X_{id-1})$ for $d=2,3,\ldots,D$ such that each Y_{id} has a Beta distribution with parameters α_{jd} and β_{jd}. This transformation means that the generalized Dirichlet mixture model can be transformed to a multidimensional Beta mixture model with conditionally independent features:

$$p(\vec{Y}_i|\Theta_M)=\sum_{j=1}^{M}p_j\prod_{d=1}^{D}p_b(Y_{id}|\alpha_{jd},\beta_{jd}) \tag{3}$$

where $p_b(Y_{id}|\alpha_{jd},\beta_{jd}) = (\Gamma(\alpha_{jd}+\beta_{jd})/\Gamma(\alpha_{jd})\Gamma(\beta_{jd}))Y_{id}^{\alpha_{jd}-1}(1-Y_{id})^{\beta_{jd}-1}$, in which $\Gamma(\cdot)$ denotes the Gamma function. Note that the mean and variance of a Beta distribution with parameters (α_{jd},β_{jd}) are given by

$$m_{jd} = \frac{\alpha_{jd}}{\alpha_{jd}+\beta_{jd}} \tag{4}$$

$$v_{jd} = \frac{\alpha_{jd}\beta_{jd}}{(\alpha_{jd}+\beta_{jd})^2(\alpha_{jd}+\beta_{jd}+1)} \tag{5}$$

Using Eq. (4), it is easy to obtain the following parametrization of our mixture model:

$$p(\vec{Y}_i|\Theta_M) = \sum_{j=1}^{M} p_j \prod_{d=1}^{D} p_b(Y_{id}|\theta_{jd}) \tag{6}$$

where the symbol Θ_M refers now to $(\{\theta_{jd}\},\vec{P})$, $p_b(Y_{id}|\theta_{jd}) = (\Gamma(s_{jd})/\Gamma(s_{jd}m_{jd})\Gamma(s_{jd}(1-m_{jd})))Y_{id}^{s_{jd}m_{jd}-1}(1-Y_{id})^{s_{jd}(1-m_{jd})-1}$ such that $\theta_{jd} = (m_{jd},s_{jd})$ and $s_{jd} = \alpha_{jd}+\beta_{jd}$. It is noteworthy that this kind of parametrization of the Beta distribution has been adopted for instance in [32,31] because it provides interpretable parameters. Indeed, m_{jd} and s_{jd} represent the mean and a measure of the sharpness of the distribution, respectively [33]. A large value of s_{jd} produces a sharply peaked distribution around the mean m_{jd}. And when s_{jd} decreases, the distribution becomes broader.

In mixture modeling, generally membership vectors $\vec{Z}_i$ are introduced for variables $\vec{Y}_i$. Let $\vec{Z}_i$ be an M-dimensional membership vector (known also as the unobserved or missing vector) that indicates to which component $\vec{Y}_i$ belongs, such that: Z_{ij} will be equal to 1 if $\vec{Y}_i$ belongs to class j or 0 otherwise. Thus,

$$p(Z_{ij}=1) = p_j, \quad j = 1,\dots,M \tag{7}$$

and the distribution of $\vec{Y}_i$ given the class label $\vec{Z}_i$ is

$$p(\vec{Y}_i|\vec{Z}_i,\Theta_M) = \prod_{j=1}^{M}\left(\prod_{d=1}^{D} p_b(Y_{id}|\theta_{jd})\right)^{Z_{ij}} \tag{8}$$

2.2. Unsupervised feature selection

It is noteworthy that the previous model in Eq. (8) supposes actually that the D features have the same importance and carry pertinent information which is not generally the case, since many of which can be irrelevant for the targeted application. In order to take into account the potential presence of irrelevant features, it is possible to represent irrelevant features by background Beta distributions with parameter ψ_d independent from class labels $\vec{Z}_i$. Indeed, we can introduce a hidden binary variable, ϕ_{id}, set to zero when Y_{id} comes from the irrelevant beta distribution with parameters ψ_d and to 1 otherwise, then the distribution of each Y_{id} is approximated as

$$p(Y_{id}|\theta_{jd},\psi_d,\phi_{id}) = (p_b(Y_{id}|\theta_{jd}))^{\phi_{id}}(p_b(Y_{id}|\psi_d))^{1-\phi_{id}} \tag{9}$$

A better and more flexible approach has been proposed in [13] by observing that in practice, irrelevant features may be distributed according to a mixture of overlapped Beta distributions common to all clusters rather than a single univariate Beta distribution. Indeed, it is well-known that a Beta mixture is flexible enough to approximate efficiently a wide variety of distributions [32,31]. Thus, let us consider the distribution of an irrelevant feature as a mixture of K_d Beta distributions $\vec{\psi}_d = (\psi_{1d},\psi_{2d},\dots,\psi_{K_d d})$ that are independent of the labels $\vec{Z}_i$, where K_d is determined for each feature d from the data. Also we introduce $\vec{W}_i$ as the component label of an irrelevant feature where $\vec{W}_i = (\vec{W}_{i1},\vec{W}_{i2},\dots,\vec{W}_{iD})$, $\vec{W}_{id} = (W_{id1},W_{id2},\dots,W_{idK_d}) \in (0,1)^{K_d}$: $\sum_k W_{idk} = 1$. In other words, $W_{idk}=1$ if Y_{id} comes from the kth component ψ_{kd} given $\phi_{id}=0$. Then,

$$p(\vec{Y}_i|\vec{Z}_i,\Theta_M,\{\psi_{kd}\},\vec{\phi}_i,\vec{W}_i) = \prod_{j=1}^{M}\left[\prod_{d=1}^{D} p_b(Y_{id}|\theta_{jd})^{\phi_{id}} \times \left(\prod_{k=1}^{K_d} p_b(Y_{id}|\psi_{kd})^{W_{idk}}\right)^{1-\phi_{id}}\right]^{Z_{ij}} \tag{10}$$

In order to obtain $p(\vec{Y}_i|\Theta)$, we start by setting the following priors: $p(\vec{Z}_i) = \prod_{j=1}^{M} p_j^{Z_{ij}}$, $p(\vec{\phi}_i) = \prod_{d=1}^{D} \epsilon_{d1}^{\phi_{id}}\epsilon_{d2}^{1-\phi_{id}}$ and $p(\vec{W}_i|\vec{\phi}_i) = \prod_{d=1}^{D}\prod_{k=1}^{K_d}(\eta_{kd}^{W_{idk}})^{1-\phi_{id}}$, where each ϕ_{id} is a Bernoulli variable with parameters $p(\phi_{id}=1)=\epsilon_{d1}$ and $p(\phi_{id}=0)=\epsilon_{d2}$ such that $\epsilon_{d1}+\epsilon_{d2}=1$. Also, η_{kd} denotes the prior probability that Y_{id} is generated from the kth common component ψ_{kd} (i.e. $\sum_{k=1}^{K}\psi_{kd}=1$) given the irrelevance of the dth feature, i.e. $\phi_{id}=0$. Then we marginalize the complete likelihood $p(\vec{Y}_i,\vec{Z}_i,\vec{\phi}_i,\vec{W}_i|\Theta_M)$ over the hidden variables $\vec{Z}_i$, $\vec{\phi}_i$, and $\vec{W}_i$ which give us [13]

$$p(\vec{Y}_i|\Theta) = \sum_{j=1}^{M} p_j \prod_{d=1}^{D}\left(\epsilon_{d1}p_b(Y_{id}|\theta_{jd}) + \epsilon_{d2}\sum_{k=1}^{K_d}\eta_{kd}p_b(Y_{id}|\psi_{kd})\right) \tag{11}$$

where $\Theta = (\Theta_M,\{\epsilon_{d1},\epsilon_{d2}\},\{\eta_{kd}\},\{\psi_{kd}\})$ is the complete set of parameters to be estimated.

3. Bayesian learning using RJMCMC

The unsupervised feature selection model presented by Eq. (11) has been learned in [13] using a deterministic approach based on the well-known EM algorithm for parameter estimation by developing and MML objective function that allows simultaneously the selection of the number of mixture components and relevant features. According to many previous theoretical and empirical studies, however, it is clear that Bayesian approaches provide better generalization capabilities than deterministic ones by allowing a better way of incorporating prior knowledge into the learning process [34]. Thus, the goal here is to develop a Bayesian alternative to the approach proposed in [13]. Bayesian inference starts by specifying a prior distribution for model parameters that is meant to capture prior beliefs. When data is obtained, the prior distribution is updated to a posterior parameter distribution.

3.1. The Bayesian model

First, let us focus on the learning of the first part of the model presented in Eq. (11) namely $\sum_{j=1}^{M} p_j \prod_{d=1}^{D} p_b(Y_{id}|\theta_{jd})$ in case all the features are supposed relevant. An important issue when learning finite mixtures is the model selection problem [35] (i.e. determination of the number of components M in our case) since we must balance the complexity of the model with its goodness of fit where the ultimate goal is to minimize the resulting generalization error. Many Bayesian procedures have been proposed for inferring the optimal number of components. Usual Bayesian approaches for model selection, first prepare a set of models and calculate the value of a given criterion for each model (e.g. Bayes factors [36]). The best model that gives the highest value is then selected. In contrast to these approaches, the approach that we shall adopt here, namely RJMCMC [26,29], allows to select the number of components simultaneously with the estimation process. Indeed, the number of components M is considered here as a parameter in the model for which a conditional distribution should be found. Thus, our unknowns are M, $\vec{\theta} = (\vec{\theta}_1, \dots, \vec{\theta}_M)$, such that $\vec{\theta}_j = (\theta_{j1}, \dots, \theta_{jD})$, and $\vec{P}$ and are regarded as random variables drawn from some prior distributions that we have to specify. The joint distribution of all these variables is

$$p(M,\vec{P},Z,\vec{\theta},\mathcal{Y}) = p(M)p(\vec{P}|M)P(Z|\vec{P},M)p(\vec{\theta}|Z,\vec{P},M)$$
$$p(\mathcal{Y}|\vec{\theta},Z,\vec{P},M) \tag{12}$$

where $\mathcal{Y} = \{\vec{Y}_1, \dots, \vec{Y}_N\}$, $Z = \{\vec{Z}_1, \dots, \vec{Z}_N\}$. Then, by imposing common conditional independencies (see, for instance, [26,29]), we get the following joint distribution:

$$p(M,\vec{P},Z,\vec{\theta},\mathcal{Y}) = p(M)p(\vec{P}|M)P(Z|\vec{P},M)p(\vec{\theta}|M)p(\mathcal{Y}|\vec{\theta},Z) \tag{13}$$

The goal of Bayesian inference is to generate realizations from the conditional joint density $p(M,\vec{P},Z,\vec{\theta}|\mathcal{Y})$.

3.2. Priors and posteriors

Let us now define the priors, which we suppose that are all drawn independently of the different parameters in our hierarchical Bayesian model. We know that each location m_{jd} is defined in the compact support [0,1], then an appealing flexible prior choice is a Beta distribution, with location ε and scale ζ common to all components, which was found appropriate in our applications. Thus, $\vec{m}_j = (m_{j1}, \dots, m_{jD})$ for each component is given the following prior:

$$p(\vec{m}_j|\varepsilon,\zeta) \sim \prod_{d=1}^{D} \frac{\Gamma(\zeta)}{\Gamma(\zeta\varepsilon)\Gamma(\zeta(1-\varepsilon))} m_{jd}^{\zeta\varepsilon-1}(1-m_{jd})^{\zeta(1-\varepsilon)-1} \tag{14}$$

Since $\vec{s}_j = (s_{j1}, \dots, s_{jD})$ control the dispersion of the distributions, a common choice as a prior is an inverse gamma with shape ϑ and scale ϖ common to all components [37], then

$$p(\vec{s}_j|\vartheta,\varpi) \sim \prod_{d=1}^{D} \frac{\varpi^{\vartheta}\exp(-\varpi/s_{jd})}{\Gamma(\vartheta)s_{jd}^{\vartheta+1}} \tag{15}$$

using the two previous equations, we have

$$p(\vec{\theta}|M,\tau) = \prod_{j=1}^{M} p(\vec{m}_j|\varepsilon,\zeta)p(\vec{s}_j|\vartheta,\varpi) \tag{16}$$

where $\tau = (\varepsilon,\zeta,\vartheta,\varpi)$ are the hyperparameters of $\vec{\theta}$. Thus, the full conditional posterior distributions for $\vec{m}_j$ and $\vec{s}_j$ are

$$p(\vec{m}_j|\dots) \propto \prod_{j=1}^{M} p(\vec{m}_j|\varepsilon,\zeta)p(\vec{s}_j|\vartheta,\varpi) \prod_{i=1}^{N} p(\vec{Y}_i|\vec{\theta}_{Z_i})$$
$$\propto p(\vec{m}_j|\varepsilon,\zeta) \prod_{i=1}^{N} p(\vec{Y}_i|\vec{\theta}_{Z_i}) \tag{17}$$

$$p(\vec{s}_j|\dots) \propto \prod_{j=1}^{M} p(\vec{m}_j|\varepsilon,\zeta)p(\vec{s}_j|\vartheta,\varpi) \prod_{i=1}^{N} p(\vec{Y}_i|\vec{\theta}_{Z_i})$$
$$\propto p(\vec{s}_j|\vartheta,\varpi) \prod_{i=1}^{N} p(\vec{Y}_i|\vec{\theta}_{Z_i}) \tag{18}$$

where $n_j = \sum_{i=1}^{N} \mathbb{I}_{Z_i=j}$ and represents the number of vectors belonging to cluster j. Moreover, we know that the vector $\vec{P}$ is defined on the simplex $\{(p_1, \dots, p_M) : \sum_{j=1}^{M-1} p_j < 1\}$, then the typical choice, as a prior, for this vector is a Dirichlet distribution with parameters $\delta = (\delta_1, \dots, \delta_M)$ [22]:

$$p(\vec{P}|M,\delta) = \frac{\Gamma\left(\sum_{j=1}^{M}\delta_j\right)}{\prod_{j=1}^{M}\Gamma(\delta_j)} \prod_{j=1}^{M} p_j^{\delta_j-1} \tag{19}$$

According to Eq. (7), we have also

$$p(Z|\vec{P},M) = \prod_{j=1}^{M} p_j^{n_j} \tag{20}$$

Using the two previous equations we obtain

$$p(\vec{P}|\dots) \propto p(Z|\vec{P},M)p(\vec{P}|M,\delta)$$
$$\propto \prod_{j=1}^{M} p_j^{n_j} \frac{\Gamma\left(\sum_{j=1}^{M}\delta_j\right)}{\prod_{j=1}^{M}\Gamma(\delta_j)} \prod_{j=1}^{M} p_j^{\delta_j-1} \propto \prod_{j=1}^{M} p_j^{n_j+\delta_j-1} \tag{21}$$

which is actually proportional to a Dirichlet distribution with parameters $(\delta_1+n_1, \dots, \delta_M+n_M)$. Now the posterior for the membership variables can be

$$p(Z_i = j|\dots) \propto p_j \prod_{d=1}^{D} p_b(Y_{id}|\theta_{jd}) \tag{22}$$

It is possible to introduce an additional hierarchical level by allowing the hyperparameters to follow some selected distributions. In our model, the hyperparameters, ε and ζ, associated with the $\vec{m}_j$ are given uniform and inverse Gamma priors, respectively:

$$p(\varepsilon) \sim \mathcal{U}_{[0,1]} \tag{23}$$

$$p(\zeta|\varphi,\varrho) \sim \frac{\varrho^{\varphi}\exp(-\varrho/\zeta)}{\Gamma(\varphi)\zeta^{\varphi+1}} \tag{24}$$

Thus, according to these two previous equations we have

$$p(\varepsilon|\ldots) \propto p(\varepsilon)\prod_{j=1}^{M} p(\vec{m}_j|\varepsilon,\zeta) \tag{25}$$

$$p(\zeta|\ldots) \propto p(\zeta|\varphi,\varrho)\prod_{j=1}^{M} p(\vec{m}_j|\varepsilon,\zeta) \tag{26}$$

The hyperparameters, ϑ and ϖ, associated with the $\vec{s_j}$ are given inverse Gamma and exponential priors, respectively:

$$p(\vartheta|\lambda,\mu) \sim \frac{\mu^{\lambda}\exp(-\mu/\vartheta)}{\Gamma(\lambda)\vartheta^{\lambda+1}} \tag{27}$$

$$p(\varpi|\phi) \sim \phi\exp(-\phi\varpi) \tag{28}$$

Thus, according to these two previous equations we have

$$p(\vartheta|\ldots) \propto p(\vartheta|\lambda,\mu)\prod_{j=1}^{M} p(\vec{s}_j|\vartheta,\varpi) \tag{29}$$

$$p(\varpi|\ldots) \propto p(\varpi|\phi)\prod_{j=1}^{M} p(\vec{s}_j|\vartheta,\varpi) \tag{30}$$

For M we take as a prior a common choice which is a Uniform $\{1,\ldots,\sigma\}$ distribution, where σ is a constant representing the maximum value allowed for M. Our complete Bayesian hierarchical model can be displayed as a directed acyclic graph (DAG) as shown in Fig. 1.

3.3. Main procedure for RJMCMC methodology

The RJMCMC methodology developed in [26] consists of six types of moves:

1. Update the mixing parameters $\vec{P}$.
2. Update the parameters $\vec{m_j}$ and $\vec{s_j}$.
3. Update the allocation Z.
4. Update the hyperparameters $\vartheta,\varpi,\varepsilon,\zeta$.
5. Split one component into two, or combine two into one.
6. The birth or death of an empty component.

Note that (1), (2), (3) and (4) are usual parameter update moves via the Gibbs sampling. On the other hand, (5) and (6) are trans-dimensional moves (involve changing the number of components by one), they constitute the

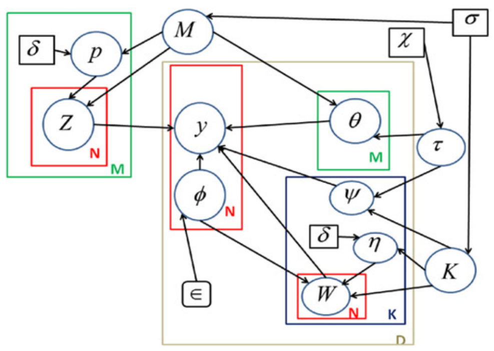

Fig. 1. Graphical model representation of our model. Nodes in this graph represent random variables, rounded boxes are fixed hyperparameters, boxes indicate repetition (with the number of repetitions in the lower right) and arcs describe conditional dependencies between variables.

reversible jump and are used for model selection via the Metropolis–Hastings (M–H) algorithm. Details and discussions about the Gibbs sampling and M–H algorithms can be found in [38,22].

Assume that a move of type m, from a state s to a state s' in a higher dimensional space, is under consideration. Usually, it is implemented by drawing a vector of continuous random variables u, independent of s, and obtaining s' with an invertible deterministic function $f(s, u)$. Then the acceptance probabilities from s to s' and vice versa are $\min\{1; R\}$ and $\min\{1; R^{-1}\}$, respectively, where

$$R = \frac{p(s'|\mathcal{Y})r_m(s')}{p(s|\mathcal{Y})r_m(s)q(u)}\left|\frac{\partial s'}{\partial(s,u)}\right| \tag{31}$$

$r_m(s)$ is the probability of choosing move m in state s and $q(u)$ is the density function of u. Not that $|\partial s'/\partial(s,u)|$ is the Jacobian function arising from the variable change from (s,u) to s'. In general, the birth-and-death moves are used for empty components contained in the mixture, while the split and combine are used for non-empty components [26,29]. As mentioned above the first four moves are classic Gibbs sampling moves. For the first move the mixing parameters are generated from Eq. (21). The second move is based on the generation of $\vec{m_j}$ and $\vec{s_j}$. It is noteworthy that the sampling of $\vec{m}_j$ and $\vec{s}_j$ is more complex, since the conditional posteriors given by Eqs. (17) and (18) are not in a well known forms. Thus, we have used the Metropolis–Hastings algorithm with log-normal and Beta proposals for $\vec{s}_j$ and $\vec{m}_j$, respectively. The third move is based on the generation of the missing data $\vec{Z}_i$ from standard uniform random variables r_n, where $Z_{ij} = 1$ if $(p(Z_{i1} = 1|\ldots) + \cdots,$ $p(Z_{ij-1} = 1|\ldots)) < r_n \leq \quad (p(Z_{i1} = 1|\ldots) + \cdots + p(Z_{ij} = 1|\ldots))$ (see Eq. (22)) [39]. The fourth move consists of updating the hyperparameters $\vartheta,\varpi,\varepsilon,\zeta$. The posterior distribution of ε, ζ, ϑ and ϖ, given by Eqs. (25), (26), (29) and (30), respectively, are not of standard forms. However, it is possible to show that they are log-concave, then the samples generation is based on the adaptive rejection sampling [40]. Now, let us consider the split and merge moves. As proposed by [26] we need to preserve the first two moments before and after the combine and split moves. In our case, the merging proposal works as follows: choose two components j_1 and j_2, where $\vec{m}_{j_1} < \vec{m}_{j_2}$ with no other $\vec{m}_j \in [\vec{m}_{j_1}, \vec{m}_{j_2}]$ (i.e adjacency condition). If these components are merged, we reduce M by 1, which forms a new components $j*$ containing all the observation previously allocated to j_1 and j_2 and then creates values for $p_{j*}, \vec{s}_{j*}, \vec{m}_{j*}$, by preserving the first two moments, as follows [31]:

$$p_{j*} = p_{j_1} + p_{j_2} \tag{32}$$

$$m_{jd*} = \frac{p_{j_1}m_{jd_1} + p_{j_2}m_{jd_2}}{p_{j_1} + p_{j_2}}, \quad d = 1,\ldots,D \tag{33}$$

$$s_{jd*} = \frac{p_{j*}m_{jd*}(1-m_{jd*})}{p_{j_1}\left(m_{jd_1}^2 + \dfrac{m_{jd_1}(1-m_{jd_1})}{s_{jd_1}+1}\right) + p_{j_2}\left(m_{jd_2}^2 + \dfrac{m_{jd_2}(1-m_{jd_2})}{s_{jd_2}+1}\right) - p_{j*}m_{jd*}^2} - 1 \tag{34}$$

When the decision is to split, we choose a component $j*$ at random to define two new components j_1 and j_2 having weights and parameters $(p_{j_1},\vec{m}_{j_1},\vec{s}_{j_1})$ and $(p_{j_2},\vec{m}_{j_2},\vec{s}_{j_2})$, respectively, conforming to Eqs. (32)–(34). Clearly, resolving these equations is an ill-posed problem for which we adopt the same solution as in [31]:

$$p_{j_1}=u_1p_{j*},\quad p_{j_2}=(1-u_1)p_{j*} \tag{35}$$

$$m_{jd_1}=m_{jd*}-u_2\sqrt{\frac{m_{jd*}(1-m_{jd*})p_{j_2}}{(s_{jd*}+1)p_{j_1}}} \tag{36}$$

$$m_{jd_2}=m_{jd*}+u_2\sqrt{\frac{m_{jd*}(1-m_{jd*})p_{j_1}}{(s_{jd*}+1)p_{j_2}}} \tag{37}$$

$$s_{jd_1}=\frac{m_{jd_1}(1-m_{jd_1})}{u_3(1-u_2^2)\frac{m_{jd*}(1-m_{jd*})}{s_{jd*}+1}\frac{p_{j*}}{p_{j_1}}}-1 \tag{38}$$

$$s_{jd_2}=\frac{m_{jd_2}(1-m_{jd_2})}{(1-u_3)(1-u_2^2)\frac{m_{jd*}(1-m_{jd*})}{s_{jd*}+1}\frac{p_{j*}}{p_{j_2}}}-1 \tag{39}$$

where u_1, u_2, u_3 are drawn from Beta distributions with parameters (2,2), (2,2) and (1,1), respectively [26]. Then, we assign the different $\vec{Y}_i$ previously in $j*$ to j_1 or j_2 using Eq. (22) (i.e. Bayes rule). Now, we calculate the acceptance probabilities of split and combine moves: $\min\{1;R\}$ and $\min\{1;R^{-1}\}$, where we have the following according to Eq. (31):

$$R=\frac{p(Z,\vec{P},M+1,\vec{\theta},\vartheta,\varpi,\varepsilon,\zeta|\mathcal{Y})b_{M+1}}{p(Z,\vec{P},M,\vec{\theta},\vartheta,\varpi,\varepsilon,\zeta|\mathcal{Y})a_MP_{alloc}q(u)}\left|\frac{\partial s'}{\partial(s,u)}\right| \tag{40}$$

where P_{alloc} is the probability of making this particular allocation, $q(u)=p(u_1)p(u_2)p(u_3)$, b_M and d_M are the probabilities of choosing split and combine moves respectively. Knowing that

$$\frac{p(Z,\vec{P},M+1,\vec{\theta},\vartheta,\varpi,\varepsilon,\zeta|\mathcal{Y})}{p(Z,\vec{P},M,\vec{\theta},\vartheta,\varpi,\varepsilon,\zeta|\mathcal{Y})}=(likelihood\ ratio)$$
$$\times\frac{p(M+1)}{p(M)}(M+1)\frac{p(\vec{P}|M+1,\delta)}{p(\vec{P}|M,\delta)}\frac{p(Z|\vec{P},M+1)}{p(Z|\vec{P},M)}$$
$$\times\frac{p(\vec{\theta}|M+1,\tau)}{p(\vec{\theta}|M,\tau)} \tag{41}$$

where

$$\frac{p(\vec{P}|M+1,\delta)}{p(\vec{P}|M,\delta)}=\frac{\Gamma\left(\sum_{j=1}^{M-1}\delta_j+\delta_{j_1}+\delta_{j_2}\right)\Gamma(\delta_{j*})p_{j_1}^{\delta_{j_1}-1}p_{j_2}^{\delta_{j_1}-1}}{\Gamma(\delta_{j_1})\Gamma(\delta_{j_2})\Gamma(\sum_{j=1}^{M-1}\delta_j+\delta_{j*})p_{j*}^{\delta_{j*}-1}}$$
$$\frac{p(Z|\vec{P},M+1)}{p(Z|\vec{P},M)}=\frac{p_{j_1}^{n_{j_1}}p_{j_2}^{n_{j_2}}\prod_{j=1}^{M-1}p_j^{n_j}}{p_{j*}^{n_{j*}}\prod_{j=1}^{M-1}p_j^{n_j}}=\frac{p_{j_1}^{n_{j_1}}p_{j_2}^{n_{j_2}}}{p_{j*}^{n_{j*}}} \tag{42}$$

and n_{j_1} and n_{j_2} are the numbers of observations to be assigned to components j_1 and j_2.

$$\frac{p(\vec{\theta}|M+1,\tau)}{p(\vec{\theta}|M,\tau)}=\frac{\varpi^{D\vartheta}\Gamma(\zeta)^D}{\Gamma(\vartheta)^D[\Gamma(\zeta\varepsilon)\Gamma(\zeta(1-\varepsilon))]^D}$$
$$\times\prod_{d=1}^{D}\frac{\exp\left(-\varpi\left(\frac{s_{jd1}+s_{jd2}}{s_{jd1}s_{jd2}}-\frac{1}{s_{jd*}}\right)\right)\left(\frac{m_{jd1}m_{jd2}}{m_{jd*}}\right)^{\zeta\varepsilon-1}\left(\frac{(1-m_{jd1})(1-m_{jd2})}{1-m_{jd*}}\right)^{\zeta(1-\varepsilon)-1}}{\left(\frac{s_{jd1}s_{jd2}}{s_{jd*}}\right)^{\vartheta+1}} \tag{43}$$

and finally the absolute value for the determinant of the Jacobian matrix can be straightforwardly calculated as follows [31]:

$$\left|\frac{\partial s'}{\partial(s,u)}\right|=\left|\frac{\partial(p_{j_1},p_{j_2},\vec{m}_{j_1},\vec{m}_{j_2},\vec{s}_{j_1},\vec{s}_{j_2})}{\partial(u_1,p_{j*},\vec{m}_{j*},u_2,\vec{s}_{j*},u_3)}\right|$$
$$=p_{j*}\prod_{d=1}^{D}\frac{(m_{jd2}-m_{jd1})(s_{jd1}+1)(s_{jd2}+1)}{u_2(1-u_2^2)u_3(1-u_3)(s_{jd*}+1)} \tag{44}$$

Our birth and death moves can be straightforwardly be obtained from the ones in one dimensional setting [31]. We first make a random choice between birth and death with probabilities a_M and b_M as above. For a birth, the parameters of the new component proposed are drawn from the associated prior distributions given by Eqs. (14) and (15), respectively. The weight of the new component, p_{j*}, is generated from the marginal distribution of p_{j*} derived from the distribution of $\vec{P}=(p_1,\ldots,p_M,p_{j*})$. The vector $\vec{P}$ follows a Dirichlet with parameters $(\delta_1,\ldots,\delta_M,\delta_{j*})$ (see Eq. (19)), thus the marginal of p_{j*} is a Beta distribution with parameters $(\delta_{j*},\sum_{j=1}^{M}\delta_j)$ [41]. Note that in order to keep the mixture constraint $\sum_{j=1}^{M}p_j+p_{j*}=1$, the previous weights p_j, $j=1,\ldots,M$ have to be rescaled and then all multiplied by $(1-p_{j*})$. The Jacobian corresponding to the birth move is then $(1-p_{j*})^M$. For the opposite move, we choose randomly an existing empty component to delete, then of course the remaining weights have to be rescaled to keep the unit-sum constraint. The acceptance probabilities of birth and death moves: $\min\{1,R\}$ and $\min\{1,R^{-1}\}$, are calculated according to Eq. (31):

$$R=\frac{p(M+1)}{p(M)}\frac{\Gamma(\delta_{j*}+\sum_{j=1}^{M}\delta_j)}{\Gamma(\delta_{j*})\Gamma\left(\sum_{j=1}^{M}\delta_j\right)}p_{j*}^{\delta_{j*}-1}(M+1)$$
$$\times(1-p_{j*})^{N+\sum_{j=1}^{M}\delta_j-M}\frac{b_{M+1}}{a_M(M_0+1)}\frac{1}{p(p_{j*})}(1-p_{j*})^M \tag{45}$$

where M_0 is the number of empty components before the birth.

Table 1
Mean estimated posterior probabilities of the number of components given the data for the two datasets, with percentage of accepted split–combine, and birth–death moves in multi-RJMCMC.

Data	$p(M\|\mathcal{X})$	% Split/combine	% Birth/death
$\mathcal{X}^{2,8}_{800,4}$	$p(1\|\mathcal{X})=0.0000p(2\|\mathcal{X})=0.0271$ $p(3\|\mathcal{X})=0.2298p(4\|\mathcal{X})=0.4622$ $p(5\|\mathcal{X})=0.2782\ \sum_{M\geq 6}^{\sigma}p(M\|\mathcal{X})=0.0027$	5.82 ± 0.71	2.72 ± 0.54
$\mathcal{X}^{3,2}_{1000,5}$	$p(1\|\mathcal{X})=0.0000p(2\|\mathcal{X})=0.0000$ $p(3\|\mathcal{X})=0.0451p(4\|\mathcal{X})=0.2274$ $p(5\|\mathcal{X})=0.5728p(6\|\mathcal{X})=0.1231$ $p(7\|\mathcal{X})=0.0285\ \sum_{M\geq 8}^{\sigma}p(M\|\mathcal{X})=0.0031$	4.13 ± 0.87	3.61 ± 0.32

The second part of our model $(\sum_{k=1}^{K_d}\eta_{kd}p_b(Y_{id}|\psi_{kd}))$ can be reduced to the learning of D finite Beta mixture models and the same Bayesian approach as the above (multidimensional RJMCMC) is applied by considering of course one-dimensional data (one dimensional RJMCMC). The complete algorithm to learn our model can be summarized as follows:

```
1. Input: Dataset $\mathcal{Y}$, $\sigma$, $\mathcal{X}=(\varphi,\varrho,\lambda,\mu,\phi)$, IterMax (Maximum number of iteration for the RJMCMC method)
2. Output: $\vec{\theta}$, $\psi$, $\vec{P}$, $\eta$, $\vec{\epsilon_1}=(\epsilon_{11},\ldots,\epsilon_{D1})$, $\vec{\epsilon_2}=(\epsilon_{12},\ldots,\epsilon_{D2})$, $M$, $\vec{K}=(K_1,\ldots,K_D)$
3. Run the Multidimensional RJMCMC
   (a) Input: Dataset $\mathcal{Y}$, IterMax, $\sigma$, $\mathcal{X}$
   (b) Output: $\vec{\theta}$, $\vec{P}$, $Z$, $M$
   (c) for $t=1$:1: IterMax do {Multidimensional RJMCMC sweep}
4. Run $D$ unidimensional RJMCMC
   (a) Input: Dataset $\mathcal{Y}$, IterMax, $\sigma$, $\mathcal{X}$
   (b) Output: $\psi$, $\eta$, $W$, $\vec{K}$
   (c) for $d=1$:1:$D$ do {Unidimensional RJMCMC}
       (i) Input: Dimension $d$ of $\vec{Y_d}$ in the Dataset, IterMax, $\sigma$, $\mathcal{X}$
       (ii) Output: $\psi_d$, $\vec{\eta_d}$, $\vec{W_d}$, $K_d$
       (iii) for t=1:1: IterMax do {Unidimensional RJMCMC sweep}
5. for $d=1$:1:$D$ Get the estimation of
   $\phi_{id}=\frac{\sum_{j=1}^{M}p_jp_b(Y_{id}|\theta_{jd})}{\sum_{j=1}^{M}p_jp_b(Y_{id}|\theta_{jd})+\sum_{k=1}^{K}\eta_{kd}p_b(Y_{id}|\psi_{kd})}$
6. Calculate $\vec{\epsilon_1}$ using $\vec{\phi}$
7. Get $\vec{\epsilon_2}=1-\vec{\epsilon_1}$
```

4. Experimental results

In this section we report results on synthetic data as well as three different interesting applications namely human action videos categorization, pedestrian detection and face recognition. We investigate the effectiveness of our algorithm by comparing it to other state of the art methods. In these applications our specific choices for the hyperparameters were $\eta_1=\ldots,\eta_M=1$, and $(\varphi,\varrho,\lambda,\mu,\phi,\sigma)=(2,5,0.2,2,1,20)$. We have used 20 parallel chains and considered the criterion of Gelman and Rubin [42] to detect convergence. In general for $M=1,2$, the Gelman–Rubin scale reduction factor came down to 1 within 50 iterations, however, for $M>3$, the scale reduction factor was slightly larger than 1 even after 2000 iterations. A burn-in period of 1000 iterations followed by 10 000 iterations was sufficient for convergence in these applications.

4.1. Synthetic data

We dedicate this section for the analysis of two generated datasets. The goal is to investigate if our algorithm is able to identify relevant features and to select the number of clusters effectively. Note that, for each of the two datasets we run the algorithm 10 times. The first synthetic dataset consists of 800 data points from a mixture of four equiprobable two dimensional Beta distributions. We also add to this data eight irrelevant features generated from Beta distributions yielding a set of 800 10-dimensional vectors. The second data is composed of 1000 data points from a mixture of five equiprobable three dimensional Beta distributions with two irrelevant features generated from mixtures of two equiprobable Beta components. Suppose that D_r and D_{ir} are the numbers of relevant and irrelevant features, respectively, such that $D_r+D_{ir}=D$. We denote a dataset $\mathcal{X}$ with N samples, M clusters, D_r relevant features and D_{ir} irrelevant features by $\mathcal{X}^{D_r,D_ir}_{N,M}$, then the two datasets will be $\mathcal{X}^{2,8}_{800,4}$ and $\mathcal{X}^{3,2}_{1000,5}$, respectively. The percentage of accepted split–combine, and birth–death moves are represented in Table 1. The saliencies of all the features, together with their standard deviations (error bars), are shown in Fig. 2. We can conclude that in both cases the algorithm successfully selects the true number of clusters and correctly assigns the feature saliencies.

4.2. Visual recognition of human actions

The proliferation of digital equipments has increased the number of video collections more and more. Thus, a crucial problem is the categorization and analysis of this content which has been the topic of extensive research in the past. In particular, automatically recognizing human actions in videos has received a lot of attention because of

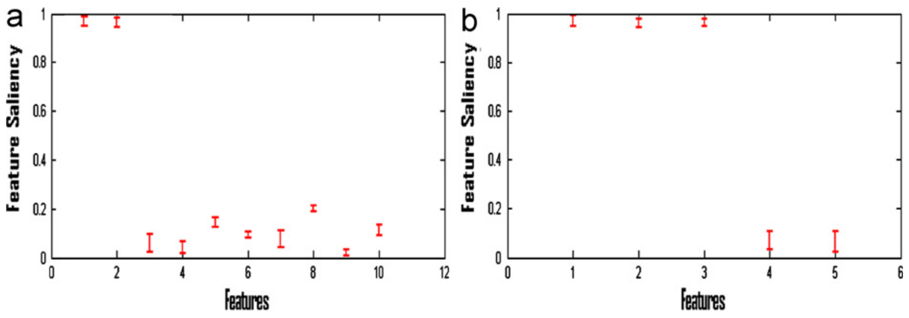

Fig. 2. Feature saliencies for (a) $\mathcal{X}^{2,8}_{800,4}$and (b) $\mathcal{X}^{3,2}_{1000,5}$.

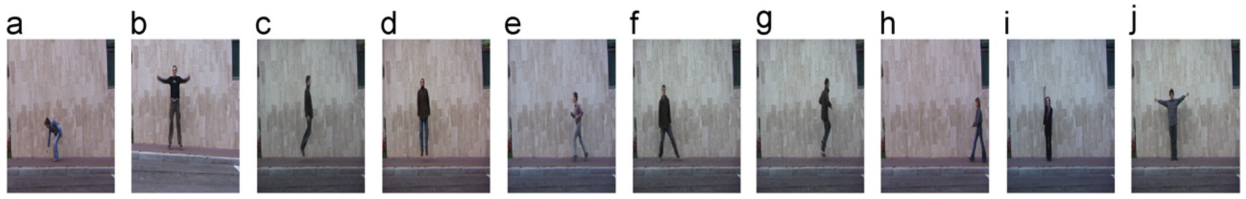

Fig. 3. Sample frames from Weizmann dataset. (a) Bend, (b) jack, (c) jump, (d) pjump, (e) run, (f) side, (g) skip, (h) walk, (i) wave1, and (j) wave2.

its potential number of applications such as in surveillance and man–machine interaction [43,44].[1] The problem is, however, challenging for computers due to cluttered background, occlusion, geometric and photometric variances of objects, and camera motion [46–49]. Recently, different sophisticated approaches have been introduced in order to obtain the best recognition results. In [50], for instance, the problem of human action recognition was tackled by using four kinds of slow feature learning strategies to extract slow feature functions from a large amount of training cuboids obtained by random sampling in motion boundaries. Then, the squared first order temporal derivatives are accumulated over all transformed cuboids in one feature vector to represent action sequences. This feature vector encodes the statistical distribution of slow features in an action sequence. Finally, a linear support vector machine (SVM) is trained to classify actions represented by these feature vectors.

The goal of this section is to investigate the problem of human action categorization in video sequences using our model. The problem of human actions categorization in a given video can be viewed as the process that given an unknown input video, the system reveals the classes of the actions present in this video. This process is generally based on three main tasks: features extraction, video representation, and action recognition. The features extraction step is mainly used in order to decrease the dimensionality of videos and this can be done by representing each video as a feature vector. Recently several approaches have advocated the use of local spatial–temporal features obtained from local video patches [51–56,44]. Thus, we follow these approaches based on the so-called bag of keypoints technique which corresponds actually to a histogram of the number of occurrences of particular visual patterns in a given image or video. Despite its simplicity this technique has different advantages such as invariance to affine transformations, occlusion, lighting and intra-class variations which allow efficient recognition [56].

In this application, we use the separable linear filter method, proposed in [57], for the spatial–temporal interest points detection since it has been shown to generally produce a high number of detections as argued in [56]. Thus, given a set of training videos we learn the vocabulary of spatial–temporal words. This vocabulary (or codebook) is constructed by applying the k-means algorithm with Euclidean distance as the clustering metric. The center of each resulting cluster is used to represent a spatial–temporal word (or codeword). Thus, each detected interest point can be assigned a unique cluster membership, allowing each video to be represented as a collection of spatial–temporal words from the codebook. Then, normalizing each resulting histogram allows to represent each video as a vector of proportions. Our developed model can be used then to cluster the set of action categories using the well-known Bayes decision rule.

We have tested our algorithm on three datasets: Weizmann human action dataset [53], KTH human motion dataset [58] and UCF sports dataset [59]. The Weizmann human action dataset contains 90 low-resolution video sequences showing nine different people, each performing 10 different actions: run, walk, jumping jack (jack), jump forward (jump), jump in place (pjump), gallop sideways (side), wave two hands (wave2), wave one hand (wave1), skip, and bend. Note that in this dataset the videos are taken with static camera and static background. Sample frames from each of the 10 actions are shown in Fig. 3. For learning the codebook we use all feature descriptors obtained from all the training video sequences [56]. We have tested seven dictionary sizes

[1] The problem is different from recognizing and discovering classes of actions in still images which has received a lot of attention, too [45].

starting with 500 words and varying it by 50 each time. Adopting a leave-one-out scheme to test the efficiency of our approach in recognition, each time we learn a model from the videos of eight subjects, and test those of the remaining subject. Note that, the result is reported as the average of nine runs.

The KTH human motion dataset consists of six types of human actions (walking, jogging, running, boxing, hand waving and hand clapping) performed several times by 25 subjects in four different scenarios: outdoors, outdoors with scale variation, outdoors with different clothes and indoors. Representative frames of this dataset are shown in Fig. 4. In order to build the codebook, we use only two videos of each action from three subjects and keep these sequences out of the training and testing sets, to avoid contamination in the data. As for the dictionary size, we learn dictionaries having seven different sizes starting by 900 visual words with 50 visual words increase used. We adopt the leave-one-out testing paradigm, also, and for each run of the learning algorithm we learn a model from the videos of 24 subjects (except those videos used to build the codebook) and perform our testing on the videos of the remaining subject.

The UCF sports dataset is a set of 150 broadcast sports video sequences and contains 10 different actions shown in Fig. 5. It is a challenging dataset with a wide range of scenes and viewpoints variability. We use all the training video sequences to build our codebook. We learn

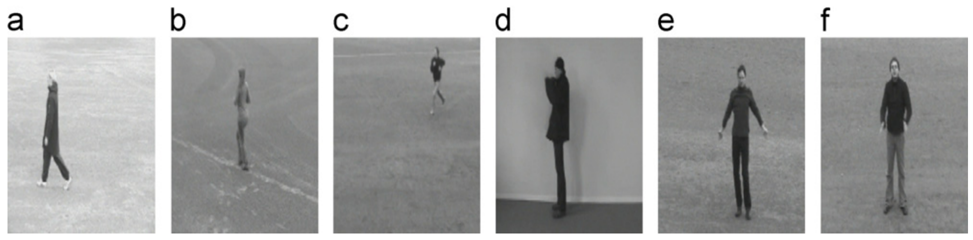

Fig. 4. Sample frames from KTH dataset. (a) Walking, (b) jogging, (c) running, (boxing, (e) hand waving, and (f) hand clapping.

Fig. 5. Sample frames from UCF sports dataset. (a) Dive, (b) golf, (c) kick, (d) w. lift, (e) ride, (f) run, (g) SK-board, (h) swing1, (i) swing2, and (j) walk.

Table 2
Confusion matrix for Weizmann dataset when using 700 codewords.

	Bend	Jack	Jump	Pjump	Run	Side	Skip	Walk	Wave1	Wave2
Bend	1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Jack	0.00	1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
Jump	0.00	0.00	0.82	0.00	0.10	0.00	0.08	0.00	0.00	0.00
Pjump	0.00	0.00	0.00	1.00	0.00	0.00	0.00	0.00	0.00	0.00
Run	0.00	0.00	0.00	0.00	0.84	0.00	0.00	0.16	0.00	0.00
Side	0.00	0.00	0.00	0.00	0.00	1.00	0.00	0.00	0.00	0.00
Skip	0.00	0.00	0.27	0.00	0.31	0.00	0.42	0.00	0.00	0.00
Walk	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.00	0.00	0.00
Wave1	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.91	0.09
Wave2	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.00

Table 3
Confusion matrix for KTH human motion dataset when using 1050 codewords.

	Walking	Running	Jogging	Hand waving	Hand clapping	Boxing
Walking	0.87	0.00	0.13	0.00	0.00	0.00
Running	0.02	0.92	0.06	0.00	0.00	0.00
Jogging	0.03	0.24	0.73	0.00	0.00	0.00
Hand waving	0.00	0.00	0.00	0.97	0.00	0.03
Hand clapping	0.00	0.00	0.00	0.03	0.90	0.07
Boxing	0.00	0.00	0.00	0.00	0.00	1.00

Table 4
Confusion matrix for UCF sports dataset when using 1200 codewords.

	Dive	Golf	Kick	W. Lift	Ride	Run	SK-board	Swing1	Swing2	Walk
Dive	0.91	0.00	0.00	0.00	0.00	0.07	0.00	0.00	0.00	0.02
Golf	0.00	0.97	0.00	0.00	0.00	0.02	0.00	0.00	0.00	0.01
Kick	0.00	0.00	0.72	0.00	0.07	0.13	0.00	0.00	0.00	0.08
W. Lift	0.02	0.00	0.00	0.98	0.00	0.00	0.00	0.00	0.00	0.00
Ride	0.00	0.00	0.06	0.00	0.93	0.00	0.00	0.00	0.00	0.01
Run	0.00	0.05	0.23	0.00	0.00	0.61	0.04	0.00	0.00	0.07
SK-board	0.00	0.00	0.11	0.00	0.00	0.00	0.89	0.00	0.00	0.00
Swing1	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.00	0.00	0.00
Swing2	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.00	0.00
Walk	0.00	0.21	0.03	0.00	0.01	0.03	0.09	0.00	0.00	0.63

Table 5
Recognition accuracy (%) for the Weizmann, KTH and UFC sports datasets when adopting dictionary sizes of 700, 1050 and 1200, respectively.

	GD+FS (%)	LDA (%)	pLSA (%)
Weizmann	89.90	80.13	80.11
KTH human motion	89.83	77.91	82.62
UFC sports dataset	86.4	78.25	79.56

dictionaries with different seven sizes starting from an initial size of 1100 increased by 50 each time. In order to test the efficiency of our approach for the recognition task, we adopt a five-fold cross-validation setup.

Tables 2–4 represent the confusion matrices of Weizmann, KTH human motion and UCF sports datasets, respectively. From the three confusion matrices we can notice that our model provides a good accuracy but has a problem in recognizing similar action classes such as those involving hand motions or leg motions. To evaluate our method we have compared it with two state of the arts methods adopted in [56]: the Latent Dirichlet Allocation (LDA) model [60] and the simpler probabilistic Latent Semantic Analysis (pLSA) [61]. From Table 5 we can notice that our approach has highest recognition rates compared to the two others. Note that these accuracies are quite high since the adopted approach does not require any tracking or background subtraction (Fig. 6).

4.3. Pedestrian detection

Pedestrian detection is an essential and fundamental task in several applications including robotics, surveillance, and automotive safety. In recent years, there has been a surge of interest in pedestrian detection [62]. However, pedestrian detection still remains an active research area in computer vision. The pedestrian detection task is challenging from a computer vision perspective due to the great variety of human appearances (very high intraclass variability), background structure, partial occlusions, and lighting conditions. Previous approaches to pedestrian detection have used either holistic or parts-based techniques. In the holistic approach a classifier uses global models such as full-body appearance [63] or silhouettes [64,65] to determine if a Region of interest (ROI) contains a full pedestrian. Parts-based approaches consist of two step: first use different image features classifiers to search for predefined parts (e.g. head, legs and arms) inside of a ROI; then, use the output of such classifiers as input of a full pedestrian classifier (i.e. local feature or part detectors). Here, we are interested in local approaches. In general, the first task when considering local approaches is to apply an interest point detector to extract keypoints. For feature description, we adopt the bag of visual words model. Our method for pedestrian detection can be divided then into four steps: (1) Use the Harris–Laplace [66] interest point detector to extract keypoints. (2) Use codebook computation (bag of visual words model) to represent each image by a frequency histogram over the visual words. (3) Apply our algorithm to model each of the two classes (Person and non-person) in the training set by a generalized Dirichlet mixture. (4) In order to perform the assignments of the images in the test set to one of the two classes we use the Bayes rule.

We tested our approach on the Daimler Pedestrian benchmark [67] that consists of 49 000 images of size 18×36 each, where 24 000 contain pedestrians and 25 000 with no pedestrians. Images were recorded at various times and locations with no particular constraints on pedestrian pose or clothing. (See Fig. 7 for samples of pedestrian and non-pedestrian images in the dataset). We split the database into five fully disjoint sets, three for training and two for testing [67], where each set has 4800 pedestrian images and 5000 non-pedestrian images. This method will allow for a variation of training and test sets during the experiments. For the dictionary size, we have applied our algorithm seven times, where each time we have increased the dictionary by 50 words. We have started with 900 word and ended it with 1200 words. Fig. 8 represents the average accuracy of our approach when changing the dictionary size.

We have compared our method with two state of the art methods. The first method is based on Haar wavelets and edge orientation histograms as features and AdaBoost for learning [68], while the second uses histograms of oriented gradients (HOG) features and support vector machine (SVM) learning [69]. In order to have a quantitative evaluation of the performance, we have used two well-known metrics, precision and recall, to quantify how

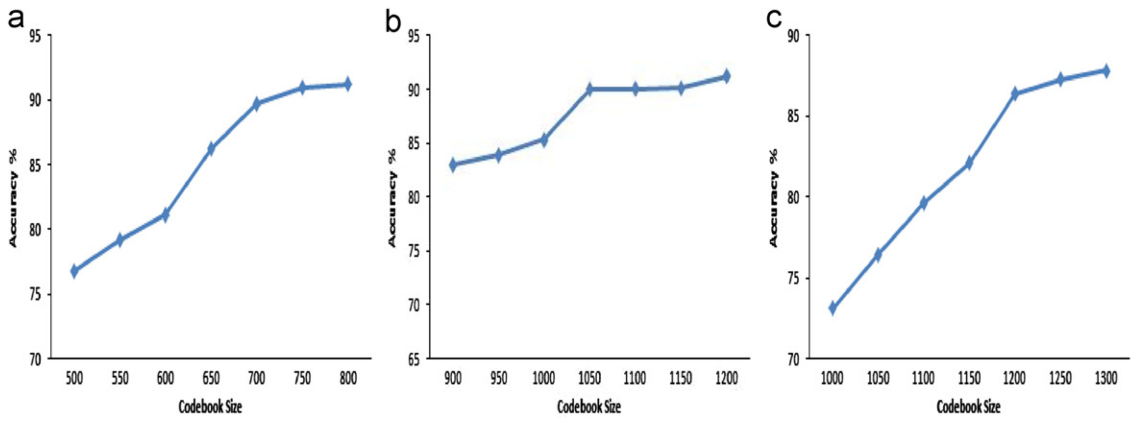

Fig. 6. Classification accuracy vs. codebook size when using our approach for (a) Weizmann, (b) KTH, and (c) UFC sports datasets.

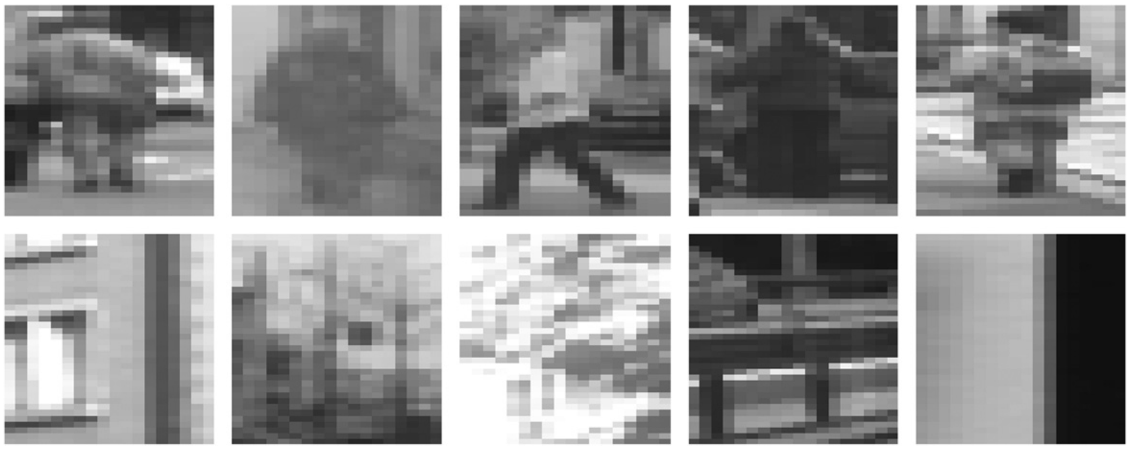

Fig. 7. Samples of pedestrian and non-pedestrian images in the Daimler pedestrian. First row: pedestrian, second row: non-pedestrian.

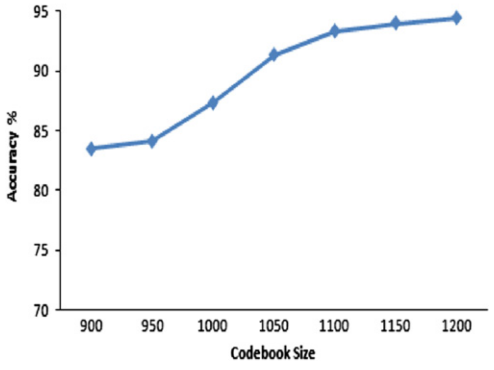

Fig. 8. Classification accuracy vs. codebook size when using our approach on the Daimler pedestrian dataset.

Table 6
Precision and recall (%) for pedestrian detection.

	GD+FS (%)	Haar+AdaBoost (%)	HOG+SVM
Precision	98.89	97.77	96.12
Recall	93.31	92.25	92.22

well each algorithm works in classifying the data. Precision (Eq. (46)) represents the percentage of number of pedestrian images correctly identified by the algorithm to the total number classified as pedestrian images. Recall (Eq. (47)) is the percentage of number of pedestrian images correctly identified by the algorithm to the total number of pedestrian images in the dataset:

$$Precision = \frac{TP}{TP+FP} \tag{46}$$

$$Recall = \frac{TP}{TP+FN} \tag{47}$$

where TP is the number of pedestrian images in the dataset that were correctly classified by the algorithm, FP is the number of non-pedestrian images that were classified as pedestrian with the algorithm, and FN is the number of pedestrian images in the dataset that were identified as non-pedestrian with the algorithm. Table 6 represents the average recall and precision for our method (GD+FS) when using 1100

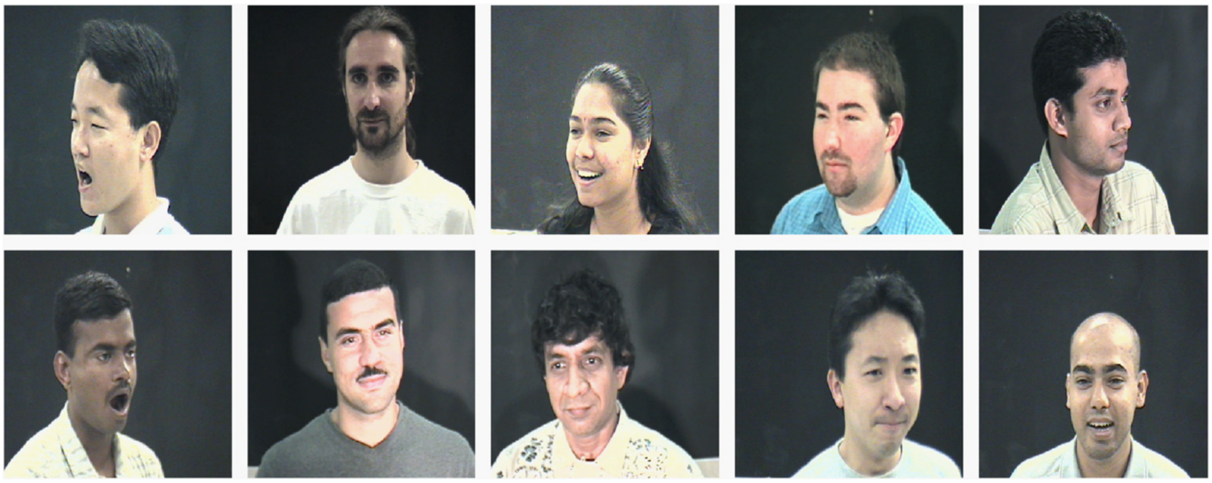

Fig. 9. Samples with different expressions, illuminations, and poses from the IRIS database.

dictionary size, the Haar wavelets and edge orientation histograms with AdaBoost (Haar+AdaBoost), and HOG with SVM (HOG+SVM). From this table we can deduce that our model is capable of detecting pedestrians efficiently.

4.4. Face recognition

In this section we consider the problem of recognizing faces in still images which has a wide range of potential applications related to security and safety [11]. Face recognition is the process that given an unknown input face, the system should reveal its identity by comparing it with a database of known persons. This process can be phrased as a learning problem and is based generally on three main tasks: face detection, feature extraction, and face identification. Face detection is used to recognize face-like objects in the given image. In order to decrease the dimensionality of face images, they can be represented in terms of low-level feature vectors in lower dimensional feature space (i.e. face signature) for recognition. Face identification is concerned with identifying the input person face by searching a database of known individuals. Despite the variety of approaches and tools studied, face recognition is still a challenging problem caused by the variations in appearance that a given face may have due to noise, illumination conditions, pose variation, and image resolution [11].

The goal here is to apply our model to recognize faces. For feature extraction step we have employed both the edge orientation histograms which provide spatial information [70] and the co-occurrence matrices which capture the local spatial relationships between gray levels [71]. Edge orientation histograms are used in order to describe the shape information contained in the images on the basis of its significant edges. We started first by applying Canny edge operator [72] to extract the edge information contained in the image, then a histogram of edge directions is used to represent the shape attribute. The main disadvantages of the edge directions histogram are that it is not scale or rotation invariant. In order to solve the problem of scale invariant, we normalized the edge histograms with respect to the number of edge points in the image. Note that, the corresponding edge directions are quantized into 72 bins of five each. Texture is another essential feature widely used in object recognition. In order to model the texture features we have computed a set of features derived from the co-occurrence matrices. It is well-known that to obtain good results, several co-occurrence matrices should be computed, each one considering a given neighborhood and direction. In our experiments, we have considered the following four neighborhoods: $(1;0)$, $(1,\pi/4)$, $(1,\pi/2)$, and $(1,3\pi/4)$, respectively [73]. For each of these neighborhoods, we calculated the corresponding co-occurrence, then derived from it the following features: mean, variance, energy, correlation, entropy, contrast, homogeneity, and cluster prominence [73]. Using the co-occurrence matrices and the histogram of edge directions each image was represented by a 110-dimensional vector. Our Bayesian approach was then used to model each class in the training set by a finite generalized Dirichlet mixture. In order to perform the assignments of the images in the test set to the different classes, we have used the following rule: $X \mapsto \text{argmax}_k \, p(\vec{X}|\Theta_k)$, where $\vec{X}$ is a 110-dimensional vector of features representing an input test image to be assigned to a class and $p(\vec{X}|\Theta_k)$ is a mixture of distributions representing class k, $k=1,\ldots,K$.

In our experiments, we performed face recognition using images from two widely used datasets: the Iris Visible Face database[2] and the AT&T face database [74]. The Iris Visible Face database contains color images of size 320×240 each and represent different persons under different expressions, poses, and illuminations. Fig. 9 shows images of different subjects in the Iris database. We have used 945 images of 15 persons not wearing

[2] OTCBVS Benchmark dataset Collection (http://www.cse.ohio-state.edu/otcbvs-bench/).

glasses. In order to evaluate our method with different set sizes we used it twice. We used for each subject 18 images as training and 45 as testing (270 and 675 for training and testing, respectively, and vice versa). On the other hand, the AT&T face database is made up of 10 different images of 40 distinct subjects. Images were taken at different orientations, illumination, and facial expressions for each subject. All the images were taken against a dark homogeneous background with the subjects in an upright, frontal position. Sample images from this dataset are shown in Fig. 10. In order to evaluate the performance with different training set sizes, we carried out three experiments. First, the dataset was subdivided into a training (tr) and testing (ts) set of 200 images (five images per subject) each. The second experiment was run using training set of size 20% and test set of 80%, while the last using 80% training and 20% testing. Note that, in all the experiments on both datasets 10 independent trials were performed with randomly chosen training and test sets. Our approach was compared to the well established algorithms of Eigenfaces [75] and Fisherfaces [76,77]. Both Eigenfaces and Fisherfaces were used with both the K-nearest neighbour approach (KNN) and the nearest mean (NM) approach; and for each of them we have used Euclidean, City-Block, and Cosine distances. The average accuracies of the three approaches are presented in Tables 7–9 for the IRIS dataset, and in Tables 10–12 for the AT&T dataset. From these tables we can see clearly that our mixture model provides good results in both datasets.

5. Conclusion

Four of the critical issues that arise when clustering and modeling objects using finite mixture models are (1) determination of what features best discriminate among the different clusters, (2) choice of the probability density functions, (3) estimation of the mixture parameters and (4) automatic determination of the number of mixture

Table 9
Our algorithm average accuracies (%) for the IRIS dataset.

	270tr+675ts	675tr+270ts
GD+FS (%)	89.63	96.30

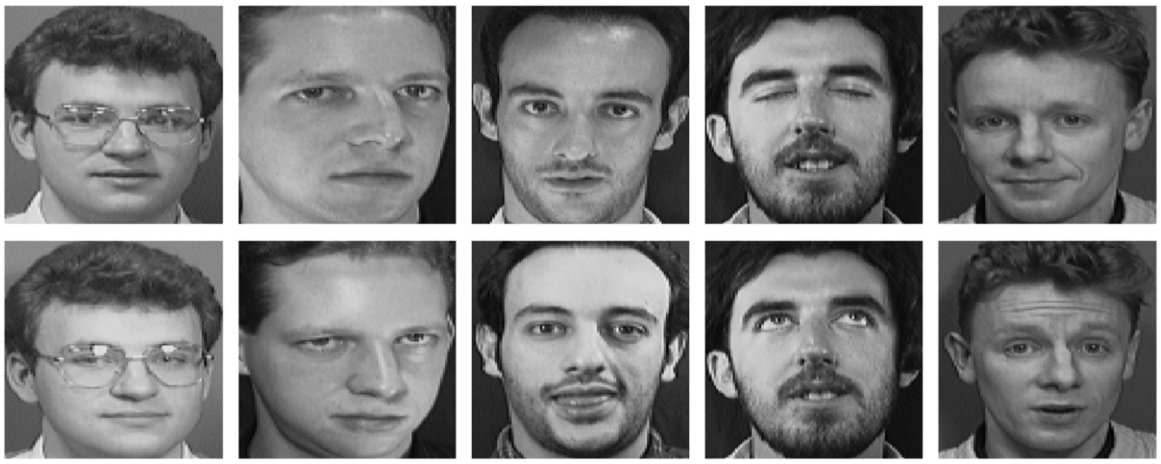
Fig. 10. Samples with different expressions, illuminations, and poses from the AT&T database.

Table 7
Eigenfaces average accuracies (%) for the IRIS dataset.

	KNN+Euclidean (%)	KNN+CityBlock (%)	KNN+Cosine (%)	NM+Euclidean (%)	NM+CityBlock (%)	NM+Cosine (%)
270tr+675ts	73.38	80.62	84.30	75.50	80.40	83.92
675tr+270ts	84.18	85.30	93.85	87.04	86.76	92.70

Table 8
Fisherfaces average accuracies (%) for the IRIS dataset.

	KNN+Euclidean (%)	KNN+CityBlock (%)	KNN+Cosine (%)	NM+Euclidean (%)	NM+CityBlock (%)	NM+Cosine (%)
270tr+675ts	72.41	82.73	71.91	75.37	81.01	72.10
675tr+270ts	81.90	90.60	78.80	78.20	89.70	77.33

Table 10
Eigenfaces average accuracies (%) for the AT&T dataset.

	KNN+Euclidean (%)	KNN+CityBlock (%)	KNN+Cosine (%)	NM+Euclidean (%)	NM+CityBlock (%)	NM+Cosine (%)
20%tr+80%ts	76.00	80.10	76.10	71.60	79.20	70.00
50%tr+50%ts	89.30	92.90	89.00	74.40	87.10	73.70
80%tr+20%ts	96.00	97.20	95.50	78.60	91.30	76.50

Table 11
Fisherfaces average accuracies (%) for the AT&T dataset.

	KNN+Euclidean (%)	KNN+CityBlock (%)	KNN+Cosine (%)	NM+Euclidean (%)	NM+CityBlock (%)	NM+Cosine (%)
20%tr+80%ts	76.80	74.70	84.60	79.00	77.40	85.00
50%tr+50%ts	91.30	90.80	93.80	91.40	91.10	93.70
80%tr+20%ts	95.20	94.10	96.00	95.60	94.60	96.20

Table 12
Our algorithm average accuracies (%) for the AT&T dataset.

	20%tr+80%ts	50%tr+50%ts	80%tr+20%ts
GD+FS (%)	90.60	97.10	99.38

components. Toward objects clustering goal, we have proposed in this paper a unified statistical Bayesian framework based on finite generalized Dirichlet mixture models that tackles all these problems simultaneously. In fact, the fully Bayesian learning approach adopted in this paper leads to a coherent treatment of the problem of unsupervised feature selection via the adoption of a set of hierarchical suitable priors for the parameters and RJMCMC sampling. The merits of the proposed work are shown through complicated computer vision examples and applications, involving high-dimensional data and large number of classes, namely human actions categorization, pedestrian detection and face recognition. Future works could be devoted to the development of a variational approach to learn the proposed model and its application to other image and signal processing applications.

Acknowledgments

The completion of this research was made possible thanks to the Natural Sciences and Engineering Research Council of Canada (NSERC).

References

[1] D.G. Stork, Toward a computational theory of data acquisition and truthing, in: D.P. Helmbold, B. Williamson (Eds.), COLT/EuroCOLT, Lecture Notes in Computer Science, vol. 2111, Springer, 2001, pp. 194–207.

[2] N. Guan, D. Tao, Z. Luo, Bo. Yuan, NeNMF: an optimal gradient method for nonnegative matrix factorization, IEEE Transactions on Signal Processing 60 (6) (2012) 2882–2898.

[3] G.J. McLachlan, D. Peel, Finite Mixture Models, Wiley, New York, 2000.

[4] A.P. Benavent, F.E. Ruiz, J.M. Sáez, Learning Gaussian mixture models with entropy-based criteria, IEEE Transactions on Neural Networks 20 (11) (2009) 1756–1771.

[5] N. Bouguila, D. Ziou, High-dimensional unsupervised selection and estimation of a finite generalized Dirichlet mixture model based on minimum message length, IEEE Transactions on Pattern Analysis and Machine Intelligence 29 (10) (2007) 1716–1731.

[6] J. Mao, A.K. Jain, Artificial neural network for feature extraction and multivariate data projection, IEEE Transactions on Neural Networks 6 (2) (1995) 296–317.

[7] J.S. Beis, D.G. Lowe, Shape indexing using approximate nearest-neighbor search in high-dimensional spaces, in: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR), 1997, pp. 1000–1006.

[8] B. Schölkopf, J.C. Platt, J. Shawe-Taylor, A.J. Smola, R.C. Williamson, Estimating the support of a high-dimensional distribution, Neural Computation 13 (7) (2001) 1443–1471.

[9] B. Heisele, T. Serre, S. Mukherjee, T. Poggio, Feature reduction and hierarchy of classifiers for fast object detection in video images, in: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR), 2001, pp. 18–24.

[10] P. Viola, M.J. Jones, Robust real-time face detection, International Journal of Computer Vision 57 (2) (2004) 137–154.

[11] S.G. Kong, J. Heo, B.R. Abidi, J. Paik, M. Abidi, Recent advances in visual and infrared face recognition—a review, Computer Vision and Image Understanding 97 (2005) 103–135.

[12] N. Vasconcelos and M. Vasconcelos, Scalable Discriminant Feature Selection for Image Retrieval and Recognition, in: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR), 2004, pp. 770–775.

[13] S. Boutemedjet, N. Bouguila, D. Ziou, A hybrid feature extraction selection approach for high-dimensional non-Gaussian data clustering, IEEE Transactions on Pattern Analysis and Machine Intelligence 31 (9) (2009) 1429–1443.

[14] M.H.C. Law, M.A.T. Figueiredo, A.K. Jain, Simultaneous feature selection and clustering using mixture models, IEEE Transactions on Pattern Analysis and Machine Intelligence 26 (9) (2004) 1154–1166.

[15] J.S. Liu, J.L. Zhang, M.J. Palumbo, C.E. Lawrence, Bayesian clustering with variable and transformation selections (with discussion), in: J.M. Bernardo, et al. (Ed.), Bayesian Statistics, 2003, pp. 249–275.

[16] W. Pan, X. Shen, Penealized model-based clustering with application to variable selection, Journal of Machine Learning Research 8 (2007) 1145–1164.

[17] G. Verbeke, E. Lesaffre, A linear mixed-effects model with heterogeneity in the random-effects population, Journal of the American Statistical Association 91 (433) (1996) 217–221.

[18] C. Fredembach, M. Schroder, S. Susstrunk, Eigenregions for image classification, IEEE Transactions on Pattern Analysis and Machine Intelligence 26 (12) (2004) 1645–1649.

[19] N. Bouguila, D. Ziou, A hybrid SEM algorithm for high-dimensional unsupervised learning using a finite generalized Dirichlet mixture, IEEE Transactions on Image Processing 15 (9) (2006) 2657–2668.

[20] C.S. Wallace, Statistical and Inductive Inference by Minimum Message Length, Springer-Verlag, 2005.

[21] G.J. McLachlan, T. Krishnan, The EM Algorithm and Extensions, Wiley-Interscience, New York, 1997.
[22] C.P. Robert, G. Casella, Monte Carlo Statistical Methods, Springer-Verlag, 1999.
[23] D.A. Mcallester, PAC-Bayesian stochastic model selection, Machine Learning 51 (2003) 5–21.
[24] A. Srivastava, E. Klassen, Monte Carlo extrinsic estimators of manifold-valued parameters, IEEE Transactions on Signal Processing 50 (2) (2002) 299–308.
[25] T. Elguebaly, N. Bouguila, Bayesian learning of finite generalized Gaussian mixture models on images, Signal Processing 91 (4) (2011) 801–820.
[26] S. Richardson, P.J. Green, On Bayesian analysis of mixtures with an unknown number of components (with discussion), Journal of the Royal Statistical Society: Series B 59 (4) (1997) 731–792.
[27] J.S. Liu, C. Sabatti, Simulated sintering: Markov chain Monte Carlo with spaces of varying dimensions, in: A.P.D.J.M. Bernardo, J.O. Berger, A.F.M. Smith (Eds.), Bayesian Statistics, 1999, pp. 389–413.
[28] M.E.A. Hodgson, A Bayesian restoration of an ion channel signal, Journal of the Royal Statistical Society. Series B (Methodological) 61 (1) (1999) 95–114.
[29] S.P. Brooks, P. Giudici, G.O. Roberts, Efficient construction of reversible jump Markov chain Monte Carlo proposal distributions (with discussion), Journal of the Royal Statistical Society: Series B 65 (1) (2003) 3–55.
[30] Z. Tu, X. Chen, A.L. Yuille, S.-C. Zhu, Image parsing: unifying segmentation, detection, and recognition, International Journal of Computer Vision 63 (2) (2005) 113–140.
[31] N. Bouguila, T. Elguebaly, A fully Bayesian model based on reversible jump MCMC and finite Beta mixtures for clustering, Expert Systems with Applications 39 (5) (2012) 5946–5959.
[32] N. Bouguila, D. Ziou, E. Monga, Practical Bayesian estimation of a finite beta mixture through Gibbs sampling and its applications, Statistics and Computing 16 (2) (2006) 215–225.
[33] D.J.C. Mackay, L. Peto, A hierarchical Dirichlet language model, Natural Language Engineering 1 (3) (1994) 1–19.
[34] J. Shawe-Taylor, R.C. Williamson, A PAC analysis of a Bayesian estimator, in: Proceedings of the Tenth Annual Conference on Computational Learning Theory (COLT), 1997, pp. 2–9.
[35] M. Kearns, Y. Mansour, A.Y. Ng, D. Ron, An experimental and theoretical comparison of model selection methods, in: Proceedings of the Eighth Annual Conference on Computational Learning Theory (COLT), 1995, pp. 21–30.
[36] R.E. Kass, A.E. Raftery, Bayes factors, Journal of the American Statistical Association 90 (1995) 773–795.
[37] B.P. Carlin, T.A. Louis, Bayes and Empirical Bayes Methods for Data Analysis, 2nd edition, Chapman & Hall/CRC, 2000.
[38] A. Gelman, G.O. Roberts, W.R. Gilks, Efficient Metropolis jumping rules, in: J.M. Bernardo, et al. (Ed.), Bayesian Statistics, 1996, pp. 599–607.
[39] Z. Zhang, K.L. Chan, Y. Wu, C. Chen, Learning a multivariate Gaussian mixture model with the reversible jump MCMC algorithm, Statistics and Computing 14 (4) (2004) 343–355.
[40] W.R. Gilks, P. Wild, Algorithm AS 287: adaptive rejection sampling from log-concave density functions, Applied Statistics 42 (4) (1993) 701–709.
[41] N.L. Johnson, S. Kotz, N. Balakrishman, Continuous Univariate Distributions, vol. 2, John Wiley and Sons, New York, 1995.
[42] A. Gelman, D.B. Rubin, Inference from iterative simulation using multiple sequences (with discussion), Statistical Science 7 (4) (1992) 457–472.
[43] D. Ramanan, D.A. Forsyth, Automatic annotation of everyday movements, in: Advances in Neural Information Processing Systems (NIPS), 2003.
[44] A. Oikonomopoulos, I. Patras, M. Pantic, Spatiotemporal salient points for visual recognition of human actions, IEEE Transactions on Systems, Man, and Cybernetics—Part B: Cybernetics 36 (3) (2006) 710–719.
[45] Y. Wang, H. Jiang, M.S. Drew, Z.-N. Li, G. Mori, Unsupervised discovery of action classes, in: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR), 2006, pp. 1654–1661.
[46] Y. Song, L. Goncalves, P. Perona, Unsupervised learning of human motion, IEEE Transactions on Pattern Analysis and Machine Intelligence 25 (7) (2003) 814–827.
[47] A.A. Effros, A.C. Berg, G. Mori, J. Malik, Recognizing action at a distance, in: Proceedings of the IEEE International Conference on Computer Vision (ICCV), 2003, pp. II.726–II.733.
[48] A. Yilmaz, M. Shah, Recognizing human actions in videos acquired by uncalibrated moving cameras, in: Proceedings of the IEEE International Conference on Computer Vision (ICCV), 2005, pp. 150–157.
[49] C. Fanti, L. Zelnik-Manor, P. Perona, Hybrid models for human motion recognition, in: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR), 2005, pp. 1166–1173.
[50] Z. Zhang, D. Tao, Slow Feature Analysis for Human Action Recognition, IEEE Transactions on Pattern Analysis and Machine Intelligence 34 (3) (2012) 436–450.
[51] F. Schaffalitzky, A. Zisserman, Automated scene matching in movies, in: Proceedings of the International Conference on Image and Video Retrieval (CIVR), 2002, pp. 186–197.
[52] F. Souvannavong, B. Mérialdo, B. Huet, Improved video content indexing by multiple latent semantic analysis, in: Proceedings of the Third International Conference on Image and Video Retrieval (CIVR), 2004, pp. 483–490.
[53] M. Blank, L. Gorelick, E. Shechtman, M. Irani, R. Basri, Actions as space-time shapes, in: Proceedings of the IEEE International Conference on Computer Vision (ICCV), 2005, pp. 1395–1402.
[54] I. Laptev, On space-time interest points, International Journal of Computer Vision 64 (2–3) (2005) 107–123.
[55] Y. Ke, R. Sukthankar, M. Hebert, Efficient visual event detection using volumetric features, in: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR), 2005, pp. 166–173.
[56] J.C. Niebles, H.C. Wang, L. Fei-Fei, Unsupervised learning of human action categories using spatial–temporal words, International Journal of Computer Vision 79 (2008) 299–318.
[57] P. Dollár, V. Rabaud, G. Cottrell, S. Belongie, Behavior recognition via sparse spatio-temporal features, in: Proceedings of the 2nd Joint IEEE International Workshop on Visual Surveillance and Performance Evaluation of Tracking and Surveillance, 2005, pp. 65–72.
[58] C. Schuldt, I. Laptev, B. Caputo, Recognizing human actions: a local svm approach, in: Proceedings of the International Conference on Pattern Recognition (ICPR), 2004, pp. 32–36.
[59] M.D. Rodriguez, J. Ahmed, M. Shah, Action MACH: a spatio-temporal maximum average correlation height filter for action recognition, in: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR), 2008, pp. 1–6.
[60] D.M. Blei, A.Y. Ng, M.I. Jordan, Latent Dirichlet allocation, Journal of Machine Learning Research 3 (2003) 993–1022.
[61] T. Hofmann, Probabilistic latent semantic indexing, in: Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, 1999, pp. 50–57.
[62] O. Tuzel, F.M. Porikli, P. Meer, Pedestrian detection via classification on Riemannian manifolds, IEEE Transactions on Pattern Analysis and Machine Intelligence 30 (10) (2008) 1713–1727.
[63] C. Papageorgiou, T. Poggio, A trainable system for object detection, International Journal of Computer Vision 38 (1) (2000) 15–33.
[64] D. Gavrila, Pedestrian detection from a moving vehicle, in: Proceedings of the European Conference on Computer Vision (ECCV)-Part II, 2000, pp. 37–49.
[65] A. Agarwal, B. Triggs, Learning to track 3D human motion from silhouettes, in: Proceedings of the International Conference on Machine Learning (ICML), 2004, pp. 9–16.
[66] K. Mikolajczyk, C. Schmid, A performance evaluation of local descriptors, IEEE Transactions on Pattern Analysis and Machine Intelligence 27 (10) (2005) 1615–1630.
[67] S. Munder, D.M. Gavrila, An experimental study on pedestrian classification, IEEE Transactions on Pattern Analysis and Machine Intelligence 28 (11) (2006) 1863–1868.
[68] D. Geronimo, A. Lopez, D. Ponsa, A.D. Sappa, Haar wavelets and edge orientation histograms for on-board pedestrian detection, in: Proceedings of the 3rd Iberian Conference on Pattern Recognition and Image Analysis (IbPRIA), 2007, pp. 418–425.
[69] N. Dalal, B. Triggs, Histograms of oriented gradients for human detection, in: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR), 2005, pp. 886–893.
[70] A.K. Jain, A. Vailaya, Image retrieval using color and shape, Pattern Recognition 29 (8) (1996) 1233–1244.
[71] K. Shanmugam, R.M. Haralick, I. Dinstein, Texture features for image classification, IEEE Transactions on Systems, Man, and Cybernetics (1973) 610–621.

[72] J. Canny, A computational approach to edge detection, IEEE Transactions on Pattern Analysis and Machine Intelligence 8 (1986) 679–698.
[73] M. Unser, Filtering for texture classification: a comparative study, IEEE Transactions on Pattern Analysis and Machine Intelligence 8 (1) (1986) 118–125.
[74] F. Samaria, A. Harter, Parameterisation of a stochastic model for human face identification, in: Proceedings of the 2nd IEEE Workshop on Applications of Computer Vision, 1994, pp. 138–142.
[75] M. Trunk, A. Pentlan, Face recognition using eigenfaces, in: Proceedings of the IEEE International Conference on Computer Vision and Pattern Recognition, 1991, pp. 586–591.
[76] P.N. Belhumeur, J.P. Hespanha, D.J. Kriegman, Eigenfaces vs. fisherfaces: recognition using class specific linear projection, IEEE Transactions on Pattern Analysis And Machine Intelligence 19 (7) (1996) 711–720.
[77] D.L. Swet, J. Weng, Using discriminant eigenfeatures for image retrieval, IEEE Transactions on Pattern Analysis And Machine Intelligence 18 (8) (1996) 831–836.