J Intell Inf Syst
DOI 10.1007/s10844-013-0282-3

Fast progressive training of mixture models for model selection

Prem Raj Adhikari · Jaakko Hollmén

Received: 11 January 2013 / Revised: 15 September 2013 / Accepted: 4 October 2013

Abstract Finite mixture models (FMM) are flexible models with varying uses such as density estimation, clustering, classification, modeling heterogeneity, model averaging, and handling missing data. Expectation maximization (EM) algorithm can learn the maximum likelihood estimates for the model parameters. One of the prerequisites for using the EM algorithm is the a priori knowledge of the number of mixture components in the mixture model. However, the number of mixing components is often unknown. Therefore, determining the number of mixture components has been a central problem in mixture modelling. Thus, mixture modelling is often a two-stage process of determining the number of mixture components and then estimating the parameters of the mixture model. This paper proposes a fast training of a series of mixture models using progressive merging of mixture components to facilitate model selection algorithm to make appropriate choice of the model. The paper also proposes a data driven, fast approximation of the Kullback–Leibler (KL) divergence as a criterion to measure the similarity of the mixture components. We use the proposed methodology in mixture modelling of a synthetic dataset, a publicly available zoo dataset, and two chromosomal aberration datasets showing that model selection is efficient and effective.

Keywords Model selection · Mixture models · KL divergence · Training · 0–1 data

P. R. Adhikari (✉) · J. Hollmén
Helsinki Institute for Information Technology (HIIT),
Department of Information and Computer Science (ICS),
Aalto University School of Science,
PO Box 15400, 00076 Aalto, Espoo, Finland
e-mail: prem.adhikari@aalto.fi

J. Hollmén
e-mail: jaakko.hollmen@aalto.fi

Published online: 01 December 2013

1 Introduction

Finite mixture models are flexible probabilistic models suitable for modelling complex data distributions. They have varying uses such as density estimation, clustering, classification, model averaging, and handling missing data (McLachlan and Peel 2000; Everitt and Hand 1981). EM algorithm provides a conceptual framework to estimate the maximum likelihood parameters from incomplete data (Dempster et al. 1977). Formulation of the EM algorithm provided the necessary impetus to the growing use of mixture models. Estimation of the parameters of mixture models involves numerous challenges. One of those challenges is the requirement of a priori knowledge of the number of components in the mixture model. Model selection in mixture model is the method of selecting the appropriate number of components in a mixture model (Tikka et al. 2007). It is one of the central problems in the finite mixture modelling.

A large number of mixture components fit the data better producing a high likelihood values for the training set. However, it also increases the model complexity, and results in an over-fitted model. The over-fitted model often generalizes poorly on future data. Conversely, smaller number of mixture components may result in an under-fitted model, which provides poor accuracy in modeling. Often some validation methods optimize this trade-off between the model accuracy and the generalization ability. However, mixture models are unsupervised models making it difficult to determine the error measure which is most widely used during validation.

Data likelihood is widely used to compare and determine the generative performance of mixture models in an unsupervised setting using cross-validation (Smyth 2000). Authors have proposed different deterministic, and stochastic and re-sampling methods to estimate the number of components in a mixture model (Figueiredo and Jain 2002). Deterministic methods consist of methods such as Akaike Information Criterion (AIC), Bayesian Information Criterion (BIC), Minimum Description Length (MDL). Stochastic and re-sampling methods consist of methods such as Markov Chain Monte Carlo (MCMC). We have used cross-validated likelihood to choose the number of components in a mixture model to model the copy number aberration patterns in cancer patients in our previous work in Tikka et al. (2007), Hollmén and Tikka (2007) and Adhikari and Hollmén (2010a, b). Similarly, in Ueda et al. (2000), the authors used splitting and merging of mixture components to ameliorate the problem of local optima in the EM algorithm. Furthermore, Zhang et al. (2003) presented another split and merge algorithm which uses different split and merge criterion such as Singular Value Decomposition (SVD) and Cholesky decomposition to split and merge components. In Ueda et al. (2000) and Zhang et al. (2003), the authors used a fixed number of components and repeated splitting and merging on the same number of mixture components to search for the global optimum.

Authors use split and merge strategy in conjunction with a validation method to select number of components in a mixture model. For example, in Li and Li (2009a), the authors used MDL criterion with split and merge algorithm to select the number of components in mixture model. Similarly, Zhang et al. (2004) proposed a competitive EM algorithm that automatically learns the number of mixture components and is insensitive to the initial configuration of number of mixture components and model parameters. Similarly, Blekas and Lagaris (2007) proposed an optimization strategy to determine the optimal number of mixing components using repeated split and merge operations. Model selection based on AIC (Akaike 1974) also uses the KL divergence as a criterion to select the number of components (Windham and Cutler 1992). In Windham and Cutler (1992), authors used the KL divergence between two prospective models penalizing the model with the higher number of mixture components. Some other distance measures proposed on measuring the dissimilarity between two Hidden Markov Models (HMMs) such as the one used in Juang and Rabiner (1985) are also based on the symmetric version of the KL divergence.

In all of these and the family of related methods, usually considered merge criteria is the similarity between the posterior probabilities of the component distributions. Ueda et al. (2000) used the normalized Euclidean distance between the two component distributions as a merge criterion. However, since the component distributions are probability distributions, using a geometric distance measure such as the Euclidean distance is unsuitable. Similarly, Zhang et al. (2004), Blekas and Lagaris (2007) and Li and Li (2009b) use the local KL divergence to measure the distance between the local data density and the model density of the component. However, the two component distributions are probability distributions and the local KL divergence provides unsatisfactory measure of the difference between them. The use of the full KL divergence is also restricted by its computationally expensive calculation.

Generally, the KL divergence between the two densities does not have a closed-form solution. Hence, authors have proposed different approximations of the KL divergence in the literature (Goldberger et al. 2003; Hershey and Olsen 2007). Similar to our method, approximation in Goldberger et al. (2003) and Hershey and Olsen (2007) are also based on the data re-sampling approach. However, they base their re-sampling approach on the MCMC while our assumption is that the samples in the data are the true samples of the distribution. Furthermore, most of the methods consider the Gaussian mixture model, and have adapted the algorithm to suit this particular choice of distribution. In our experiments, we use the finite mixture models of the multivariate Bernoulli distributions to model the chromosomal aberration patterns in cancer.

Authors have proposed different data driven approximations of the KL divergence. Lee and Park (2006) proposed two estimators of the KL divergence by local likelihood. Leonenko et al. (2008) proposed a technique to estimate the KL divergence using the Monte-Carlo estimator. Similarly, Wang et al. (2005) proposed a data–driven universal estimator of the divergence but only for continuous functions. Additionally, Perez-Cruz (2008) also proposed method for estimating the KL divergence between two continuous densities without the need for estimating the densities. These above methods are suitable for continuous densities. Authors have also proposed methods for discrete data. For instance, Cai et al. (2006) proposed two algorithms borrowed from data compression techniques to estimate the divergence from the realizations of two unknown finite–alphabet sources. Furthermore, our proposed approximation satisfies all three properties of the KL divergence but it is unusable as a metric because similar to the full KL divergence it dissatisfies triangle inequality.

This paper is the extension of our earlier paper (Adhikari and Hollmén 2012). In this paper, we propose an approximation of the exact KL divergence to determine the similarity between the mixture components with an aim to merge the most similar components. We do not propose a model selection criterion and our series of models works in conjunction with any model selection criterion such as AIC, BIC, and MDL. We repeatedly merge mixture components and retrain the mixture models to generate a series of mixture models. For example, using cross-validation on the series of models, we can select the model with optimal number of components expected to produce the best generalization performance. We also propose a fast data driven approximation of the KL divergence and use it to select two candidate components to merge in a mixture model. We perform the experiments on different 0–1 datasets showing that the results of our methods are plausible.

The organization of current paper is as follows. Section 2 briefly reviews the mixture models of the multivariate Bernoulli distributions and the EM algorithm. Section 3 presents the KL divergence and its derivation for the finite mixture model of multivariate Bernoulli distributions to compare two mixture components in a mixture model. Section 4 discusses the experiments performed on the one synthetic dataset, one publicly available zoo dataset, and two real–world datasets describing chromosomal aberrations patterns in cancer and analyzes the obtained results. Section 5 draws the conclusions from the experimental results.

2 Mixture models and EM algorithm

Finite mixture models represent a statistical distribution using a mixture (or weighted sum) of simple distributions such as Gaussian, Poisson, and Bernoulli. We achieve this by decomposing the probability density function into a set of component density functions (McLachlan and Peel 2000; Everitt and Hand 1981). $\Theta = \{J, \{\pi_j, \theta_j\}_{j=1}^J\}$ parameterizes a finite mixture of multivariate Bernoulli distributions having J components for a data set $\boldsymbol{X} = \{\mathbf{x}_1, \ldots, \mathbf{x}_N\}$ consisting of data vectors of dimensionality d. Mathematically, we define mixture models as:

$$p(\mathbf{x} \mid \Theta) = \sum_{j=1}^{\mathrm{J}} \pi_j \prod_{i=1}^{\mathrm{d}} \theta_{ji}^{x_i} (1 - \theta_{ji})^{1-x_i}. \tag{1}$$

Here π_j are the mixture proportions satisfying the properties such as convex combination: $\pi_j \geq 0$ and $\sum_{j=1}^{\mathrm{J}} \pi_j = 1 \ \forall \ j = 1, \ldots, J$. θ_{ji} defines the probability that random variable of the jth component in the ith dimension will take the value 1. Similarly, Θ_j denotes the vector of random variables of the component j. Therefore, $\Theta_j = \theta_{j,1}, \theta_{j,2}, \theta_{j,3}, \ldots, \theta_{j,d}$ where d denotes the dimensionality of the data. Θ denotes all the parameters of the data including mixture components. Therefore, $\Theta = \{J, \{\pi_j, \theta_j\}_{j=1}^J\}$. x_i denotes the data point such that $x_i \in \{0, 1\}$. Learning the parameters of a mixture model of Bernoulli distributions means learning the parameters Θ which includes the number of components J from the given data $\boldsymbol{X}$ of dimensionality d. We can formulate the learning in log-likelihood terms as:

$$\mathcal{L}(\Theta) = \sum_{n=1}^{\mathrm{N}} \log \ P(x_n \mid \Theta) = \sum_{n=1}^{\mathrm{N}} \log \left[\sum_{j=1}^{\mathrm{J}} \pi_j \prod_{i=1}^{\mathrm{d}} \theta_{ji}^{x_{ni}} (1 - \theta_{ji})^{1-x_{ni}} \right]. \tag{2}$$

Component-wise differentiation of (2) with respect to results θ_j and π_j in:

$$\frac{\delta\mathcal{L}}{\delta\pi_j} = \frac{1}{\pi_j}\sum_{n=1}^{\mathrm{N}} P(j|x_n;\pi,\Theta) - N \quad j = 1,\ldots,J \tag{3}$$

And also

$$\frac{\delta\mathcal{L}}{\delta\theta_{ji}} = \frac{1}{\theta_{ji}(1-\theta_{ji})}\sum_{n=1}^{\mathrm{N}} P(j|x_n;\pi,\Theta)(x_{ni}-\theta_{ji})$$
$$\text{where} \quad j = 1,\ldots,J \text{ and } i = 1,\ldots,d \tag{4}$$

The term -N in equation satisfies the constraint $\sum_{j=1}^{J}\pi_j$ introduced in loglikelihood via Lagrange multiplier. Now, from Bayes' theorem, we can calculate the posterior probability as:

$$P(j \mid \boldsymbol{X};\Theta) = \frac{\pi_j\prod_{i=1}^{d}\theta_{ji}^{x_{ni}}(1-\theta_{ji})^{1-x_{ni}}}{\sum_{j'=1}^{J}\prod_{i=1}^{d}\theta_{j'i}^{x_{ni}}(1-\theta_{j'i})^{1-x_{ni}}}. \tag{5}$$

Now, the EM algorithm (Dempster et al. 1977; Wolfe 1970) is two stage iterative algorithm defined by:

E-step: E-step computes the posterior probability using (5) for the most recent values of parameters π^τ, Θ^τ at iteration τ, i.e., calculates $P(j \mid x_n;\pi^\tau,\Theta^\tau)$

M-step: M-step recomputes the values of parameters $\pi^{\tau+1}$, $\Theta^{\tau+1}$ for the next iteration.

$$\pi_j^{(\tau+1)} = \frac{1}{N}\sum_{n=1}^{N} P\left(j \mid x_n;\pi^{(\tau)},\Theta^{(\tau)}\right)$$
$$\Theta_j^{(\tau+1)} = \frac{1}{N\pi_j^{(\tau+1)}}\sum_{n=1}^{N} P\left(j \mid x_n;\pi^{(\tau)},\Theta^{(\tau)}\right)x_n \tag{6}$$

Iterations between the E and the M steps produce a succession of monotonically increasing sequence of log-likelihood values for the parameters $\tau = 0, 1, 2, 3, \ldots$ regardless of the starting point $\{\pi^{(0)},\Theta^{(0)}\}$ (Mclachlan and Krishnan 1996). The EM algorithm is sensitive to initializations but is deterministic for a given initialization and a given dataset.

3 Kullback–Leiber divergence

The Kullback–Leibler divergence is the non-symmetric difference between two probability distributions (Kullback and Leibler 1951; Cover and Thomas 1991). Mathematically, the KL divergence between two discrete probability distributions P, and Q on a finite set $\boldsymbol{X}$ is:

$$\mathcal{D}_{KL}(P \,||\, Q) = \sum_{x\in\boldsymbol{X}} P(x)\log\frac{P(x)}{Q(x)} \tag{7}$$

KL divergence is unsymmetric because KL divergence from P to Q is different from the KL divergence from Q to P. Nevertheless, we can symmetrize the KL divergence by summing the KL divergence from P to Q and from Q to P (Juang and Rabiner 1985). Symmetrized KL divergence satisfies the properties of distance metric such as positivity, self–similarity, and self-identification. However, KL divergence does not satisfy triangle inequality. Both the KL divergence and the EM algorithm work on the same quantity of likelihood. However, we are using the KL divergence to compare the two component distributions and not two prospective mixture models. We write the symmetrized KL divergence and reorder the terms to separate the difference of probabilities and the log likelihood ratio of the models as:

$$\begin{aligned}\mathcal{D}_{KL} &= \mathcal{D}_{KL}(P\,||\,Q) + \mathcal{D}_{KL}(Q\,||\,P)\\ &= \sum_{x\in\boldsymbol{X}} P(x)\log\frac{P(x)}{Q(x)} + \sum_{x\in\boldsymbol{X}} Q(x)\log\frac{Q(x)}{P(x)}\\ &= \sum_{x\in\boldsymbol{X}}\left\{P(x)\log\frac{P(x)}{Q(x)} + Q(x)\log\frac{Q(x)}{P(x)}\right\}\\ &= \sum_{x\in\boldsymbol{X}}\left\{(P(x)-Q(x))\log\frac{P(x)}{Q(x)}\right\}. \qquad (8)\end{aligned}$$

3.1 KL divergence between components of FMM of multivariate Bernoulli distributions

Let us denote the two component distributions, first P and second Q, from a mixture model as θ and β respectively. Also, let θ_k and β_k denote the k^{th} parameters of the component distributions θ and β respectively. Similarly, $\boldsymbol{X}$ is a matrix of random variables (x_{ik}) that denotes the binary state-space for the random variable x of given dimensionality d indexed by k. From Adhikari and Hollmén (2012), we can write the symmetric KL divergence, generalized to an arbitrary dimension of data d, for two component distributions in a mixture model as:

$$\begin{aligned}KL_{\theta\beta} = \sum_{i=1}^{2^d}\Bigg[&\left\{\prod_{k=1}^{d}\left(\theta_k^{x_{ik}}(1-\theta_k)^{(1-x_{ik})}\right) - \prod_{k=1}^{d}\left(\beta_k^{x_{ik}}(1-\beta_k)^{(1-x_{ik})}\right)\right\}\\ &\cdot\ \log\prod_{k=1}^{d}\frac{\theta_k^{x_{ik}}(1-\theta_k)^{(1-x_{ik})}}{\beta_k^{x_{ik}}(1-\beta_k)^{(1-x_{ik})}}\Bigg]. \qquad (9)\end{aligned}$$

We can replace the log and the product in the last term with a summation and log resulting in an equation of the form:

$$\begin{aligned}KL_{\theta\beta} = \sum_{i=1}^{2^d}\Bigg[&\left\{\prod_{k=1}^{d}\left(\theta_k^{x_{ik}}(1-\theta_k)^{(1-x_{ik})}\right) - \prod_{k=1}^{d}\left(\beta_k^{x_{ik}}(1-\beta_k)^{(1-x_{ik})}\right)\right\}\\ &\cdot\sum_{k=1}^{d}\ \log\ \frac{\theta_k^{x_{ik}}(1-\theta_k)^{(1-x_{ik})}}{\beta_k^{x_{ik}}(1-\beta_k)^{(1-x_{ik})}}\Bigg]. \qquad (10)\end{aligned}$$

Here the first summation $\sum_{i=1}^{2^d}$ is a large sum and so the calculation is computationally expensive. The number of comparisons required for a mixture model having J components modelling a data of dimensionality d is $2^d J^2$. Furthermore, we use ten-fold cross-validation in our experiments which further increases the complexity. This computation is feasible when the dimensionality of data is low ($d \ll n$), often less than 10.

We can enumerate all the possible states *present* in the data instead of enumerating all the possible states. The states absent in the data are improbable and the samples present in the dataset better approximate the KL divergence. Furthermore, using only the data samples in the data provides a data driven approach to approximating to KL divergence. Thus, $\sum_{i \in X^*}$ can approximate the summation $\sum_{i=1}^{2^d}$, where $\boldsymbol{X}$ denotes the dataset and $\boldsymbol{X}^* = \{x^* : x^* \in \boldsymbol{X}\}$ is a set of all the unique data samples present in the dataset. Here also i indexes all the unique samples in the dataset. Now, we can approximate (10) as:

$$KL_{\theta\beta} = \sum_{i \in x^*} \left[\left\{ \prod_{k=1}^{d} \left(\theta_k^{x_{ik}^*} (1-\theta_k)^{(1-x_{ik}^*)} \right) - \prod_{k=1}^{d} \left(\beta_k^{x_{ik}^*} (1-\beta_k)^{(1-x_{ik}^*)} \right) \right\} \cdot \sum_{k=1}^{d} \log \frac{\theta_k^{x_{ik}^*} (1-\theta_k)^{(1-x_{ik}^*)}}{\beta_k^{x_{ik}^*} (1-\beta_k)^{(1-x_{ik}^*)}} \right]. \tag{11}$$

In the Fig. 2, we empirically verify that there is no considerable loss of information when approximating KL divergence using the unique samples in the data. We sacrifice the accuracy of the KL divergence computation for gain in computational speed. We tabulate the gain in the computational efficiency in Table 1.

Again coming back to (8), we can define the probability distributions $P(x)$ and $Q(x)$ for the component distributions from a multivariate Bernoulli distribution in the following intervals:

$$P(x) \in [0, 1] \text{ and } Q(x) \in [0, 1] \Rightarrow \frac{P(x)}{Q(x)} \in [0, \infty] \text{ and } \log \frac{P(x)}{Q(x)} \in [-\infty, \infty] \tag{12}$$

Equation (12) shows that $\log \frac{P(x)}{Q(x)}$ in (8) has infinite range with the possibility of taking any values between $+\infty$ and $-\infty$. However, the probability terms $P(x)$ and $Q(x)$ are generally small because they are the product of numerous probability terms. For example, in chromosomal aberration dataset, it is always product of more than 8 (smallest dimensionality of data in chromosome 21) probability terms. Furthermore, we have a small background probability, ϵ>0, in our model such that both $P(x)$ and $Q(x)$ are never zeros.

The use of ϵ is also compensates for the left out state-space from possible 2^d samples. There will be no problem of normalization due to the addition of ϵ because we

Table 1 Difference in time requirement for the calculation of full KL divergence and our approximation

Chromosome	Time in sec. for KL	
(dimension)	Full	Approx
21 in Data 1 (8)	0.0992	0.0156
20 in Data 1 (10)	0.4863	0.0567
21 in Data 2 (14)	10.3447	0.0295
20 in Data 2 (20)	900.7118	0.0965

add ϵ to individual probabilities which is not greater than 1. Additionally, (8) describes the symmetric KL divergence and the choice of $P(x)$ and $Q(x)$ is arbitrary. Hence, multiplying (8) by log ratio only weighs the information in the difference measure of previous terms.

Figure 1 shows the scatter-plot of the minimum KL divergence obtained by our approximation against the full and accurate KL divergence between two random components. We experiment with one hundred, ten dimensional random models parameterized by six component distributions. The random models are mixture models initialized at random. However, the two components selected, based on the minimum KL divergence can mismatch between full and accurate KL divergence and our approximation. We can then merge two mistakenly selected components. Nevertheless, we compensate for such mismatches by retraining the mixture models after merging the mixture components.

It is important to note that we are primarily interested in determining the two closest component distributions in a mixture model. The exact minimum values between two component distributions in a mixture model or the true KL distances is of secondary concern. The chain of models we train and not the accuracy of approximation of the KL divergence determines the quality of our approximation. Figure 1 shows that our approximation of the KL divergence is less extreme but the minimum values are highly correlated with the full and accurate symmetrized KL divergence. Figure 1 also shows that the approximated KL divergence varies from 0.3 to 1 whereas the exact and accurate KL divergence varies from 4 to 16. This difference coupled with our good accuracy in selection of two components with the minimum KL divergence shows that our approximation ignores the terms that amplifies the difference but keeps the terms that contribute to the difference.

We used two different assumptions to approximate the full KL divergence: firstly, using only the samples present in the data and secondly, dropping the log term. We calculate the accuracy of our approximation compared to that of the full KL divergence to empirically verify that our approximation is suitable estimate of the full KL divergence. We need a mixture model and a dataset to run our algorithm.

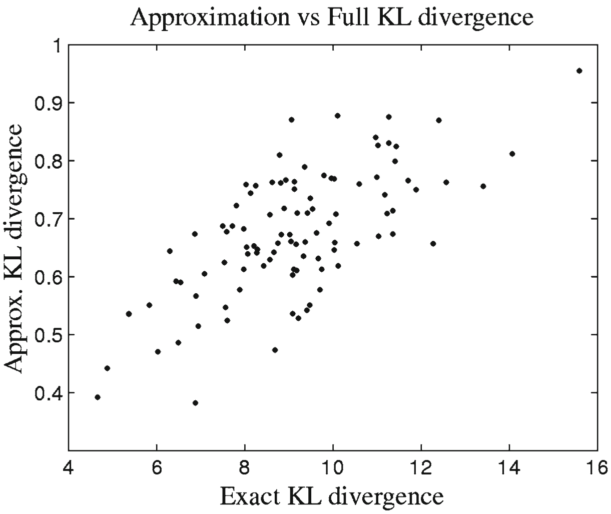

Fig. 1 Scatter-plot of minimum KL divergence values using our approximation of the KL divergence dropping the log term and also using the unique samples of data instead of the full binary state-space of the random variable x against the full and accurate KL divergence

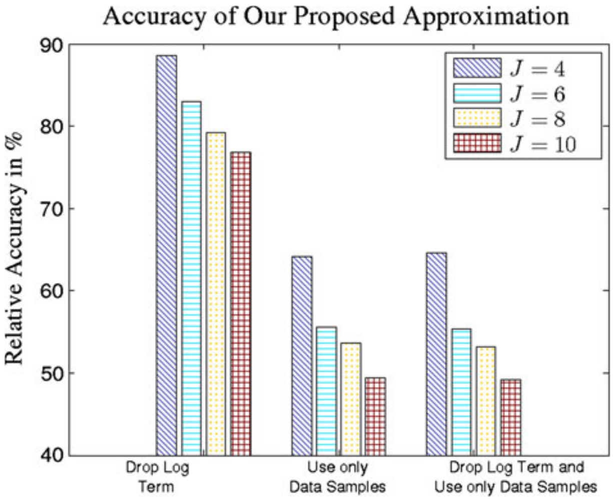

Fig. 2 The relative accuracy in the calculation of minimum KL divergence values using our approximation of the KL divergence dropping the log term and also using the unique samples of data instead of the full binary state-space of the random variable *x*

Thus, we first initialize four different mixture models with four, six, eight and ten components respectively. Secondly, we generate 5000 data points each from each mixture model. Thirdly, we use our algorithm to approximate two components in the mixture model.

Accuracy of calculation is two-fold because we approximate two components. In the experiments, either one of the two components matches or both the components match. Hence, we report both the accuracies showing that our approximation is similar to the full symmetric KL divergence. Figure 2 shows that the accuracy of matching of both the components is more than 50 % when the number of components is 10. Results of random matching is the combination of 10 components taken two at a time which is $1/\frac{n!}{k!(n-k)!} = 1/\frac{10!}{2!(10-2)!} \approx 2$ %. As the number of components gets higher, as expected, the accuracy gets lower.

Furthermore, Fig. 2 shows that most of the inaccuracy in the KL divergence approximation originates from using the data samples instead of the entire state-space of the random variable. However, using only the data–samples also contributes considerably to the speeding up of the algorithm. Finally, we can approximate the KL divergence between two component distributions omitting the log term and using only the samples present in the data as:

$$KL_{\theta\beta} = \sum_{i \in x^*} \left\{ \prod_{k=1}^{d} \left(\theta_k^{X_{ik}^*} (1-\theta_k)^{(1-x_{ik}^*)} \right) - \prod_{k=1}^{d} \left(\beta_k^{x_{ik}^*} (1-\beta_k)^{(1-x_{ik})} \right) \right\}. \qquad (13)$$

3.2 Proposed fast progressive training of mixture models

Mixture models are widely used for clustering and our proposed algorithm in probabilistic modelling domain is similar to the hierarchical agglomerative clustering (Beeferman and Berger 2000). We merge the mixture components in the same manner as in the hierarchical agglomerative clustering. Additionally, we train all the parameters of the model subsequent to the merge operations. This is where our approach differs from the hierarchical agglomerative clustering. Furthermore, the hierarchical agglomerative clustering starts with the number of clusters equal to the data points. In contrast, our algorithm starts with smaller values for the number of components such as 20 in our experiments, because the complexity of the mixture models is generally high.

Algorithm 1 Fast Progressive Training of Finite Mixture Models

```
Input: Dataset $\boldsymbol{X}$, and Maximum No. of Components $J$
Output: A series of Mixture models $\{\boldsymbol{\Theta_j}\}_{j=1}^{J}$ with different number of component distributions $j = 1, 2, \dots J$
 1: $\boldsymbol{\Theta}_J$ ← Best of 100 mixture models trained on data $\boldsymbol{X}$ having $J$ components based on likelihood on $\boldsymbol{X}$
 2: for $j$ in $J$ to 1 do
 3:     if $j! = J$ then
 4:         $\boldsymbol{\Theta}_j$ ← A trained mixture model on $\boldsymbol{X}$ using $\bar{\boldsymbol{\Theta}}_j$ as initialization
 5:     end if
 6:     if $j! = 1$ then
 7:         $(k^*, l^*) \leftarrow \underset{k,l}{\operatorname{argmin}} \ \boldsymbol{\mathcal{D}_{KL}}(p(x;\Theta_k)); p(x;\Theta_l))$
            where $k, l \in (1 \dots J); k \neq l$
 8:         $\bar{\boldsymbol{\Theta}}_{j-1}$ ← Mixture model where components $\pi_{k^*}, \pi_{l^*}$ in $\boldsymbol{\Theta}_j$ are merged
 9:     end if
10: end for
11: return Series of mixture models $\{\boldsymbol{\Theta_j}\}_{j=1}^{J}$
```

Algorithm 1 shows the algorithmic flow of our proposed algorithm. In Algorithm 1, $\bar{\Theta}_j$ and Θ_j denote initialized and trained model having j components respectively. The algorithm consists of three main operations. Firstly, we calculate the KL divergence between different components in a mixture model to determine the mixture components having the minimum KL divergence (Step 7 in Algorithm 1). Secondly, we merge the mixture components with the minimum KL divergence (Step 8 in Algorithm 1). Finally, we retrain the mixture models (Step 4 in Algorithm 1). We can use the algorithm in conjunction with any model validation criterion such as cross–validation, MDL, AIC, and BIC. For example, in Adhikari and Hollmén (2012), we have listed an algorithm that uses cross–validation in conjunction with our strategy to select optimal number of mixture components in a mixture model.

3.3 Merging of mixture components

We select two components that have the minimum symmetric KL divergence in a mixture model (Ueda et al. 2000). We merge the selected components and their parameter values as in (14) and (15), respectively. Here, $\pi_{\text{klmin},1}$ and $\pi_{\text{klmin},2}$ are the two candidate mixing coefficients of the component distributions with the minimum KL divergence selected to merge. Similarly, π_{merged} is the mixing coefficient of the component distributions obtained after merging the two component distributions and $\pi_{\text{klmin},1}$ and $\pi_{\text{klmin},2}$.

$$\pi_{\text{merged}} = \pi_{\text{klmin},1} + \pi_{\text{klmin},2} \tag{14}$$

$$\Theta_{\text{merged}} = \frac{\pi_{\text{klmin},1} \times \Theta_{\text{klmin},1} + \pi_{\text{klmin},2} \times \Theta_{\text{klmin},2}}{\pi_{\text{klmin},1} + \pi_{\text{klmin},2}} \tag{15}$$

Similarly, we can merge the parameters according to the weight of the component distributions as in (15). Here, Θ_{merged} are the parameter vectors of the component π_{merged}, obtained by merging the two components in (14). Similarly, $\Theta_{\mathrm{klmin},1}$ and $\Theta_{\mathrm{klmin},2}$ are the parameter vectors of the two components $\pi_{\mathrm{klmin},1}$ and $\pi_{\mathrm{klmin},2}$ having the minimum KL divergence.

4 Experiments with fast progressive training of mixture models

We perform experiments with our proposed algorithm on one artificial dataset, one publicly available dataset, and two different real world chromosomal aberrations dataset. We use BernoulliMix, an open-source program package for finite mixture modelling of Multivariate Bernoulli distributions, to learn the parameters of mixture model using the EM algorithm.

We calculate the KL divergence between all the pairs of component distributions in the mixture model and select the one with the minimum KL divergence. We merge the selected components and their parameters as in (14) and (15). We obtain model with $(j-1)$ components by merging the two components in a mixture model having (j) components. This model with $(j-1)$ components will be an initialization model to train a final mixture model having $(j-1)$ components. This merging and retraining starts with 20 components and ends when the number of components becomes 1.

Figure 3 shows two adjoining mixture models that summarize the aberration patterns in cancer patients. Seven rows in the left denote seven components in the mixture model whereas 6 different rows in the right indicate 6 different components in the mixture model. The detached blocks in the left of each of the mixture models spanning one block each in each row visualizes the parameters of the component distributions while the adjoining blocks spanning eight blocks visualizes the parameters of that component distribution. Darker color denotes the higher values of the parameters and lighter color denotes lower values of the parameters. Adjacent to it on the right after the block arrow shows a mixture model having six components obtained after merging components 6 and 7 in the mixture model shown on the left panel. We train the mixture model in right panel to convergence. The correspondence between the components in the mixture models in the left and the right panels can be easily established. Components 1 to 5 in left panel panel corresponds to the components 1 to 5 in right panel. Combination of the components

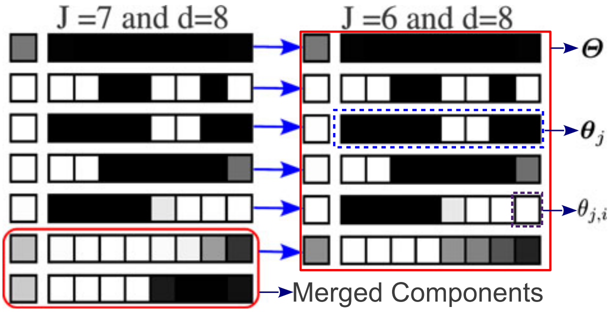

Fig. 3 Merging of components in the mixture model. Components 6 and 7 of the mixture model in the *left panel* of the figure have the minimum KL divergence of all the components. Hence, these two components are merged to form the single component in the mixture model in the *right panel* resulting in a mixture model having six components

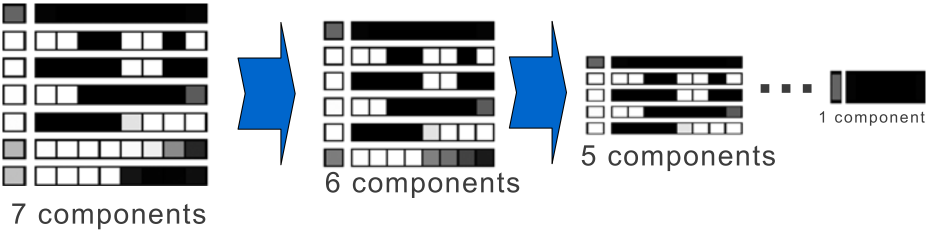

Fig. 4 Small snapshot of proposed algorithm where two components in a mixture models with 7 components are merged to generate a mixture model with 6 components and then another mixture model with five components progressively until we have a mixture model with a single component

6 and 7 in the model on the left panel results in component 6 in the model on the right of the left panel.

Our strategy for fast progressive training of mixture models is a search-based procedure that proceeds by going from complex to simple models, and is thus similar to the backward subset selection algorithm in feature selection literature (Kittler 1986). We have used a similar strategy in our sequential input selection algorithm SISAL in time–series prediction setting (Tikka and Hollmén 2008). However, here we select the component distributions unlike the data features (Tikka and Hollmén 2008).

We initially train the mixture model with a high number of mixture components, select two component distributions that are closest to each other, and merge them. This is progressively repeated until the number of components becomes 1. Furthermore, we restrict the maximum number of components to 20 because the highest dimensionality of data is 63 and mixture models with more than 20 components overfits the data. A mixture model with 20 components for data of dimensionality 63 consists of 1280 ($20 \times 63 + 20$) parameters. This big number of parameters is difficult to optimize with small size of data samples. Initially, we train 100 different models with 20 components via the EM algorithm using BernoulliMix program package. We select the best performing model of the 100 models for merging based on the likelihood in the data to minimize the problem of local optima of the EM algorithm.

Figure 4 shows the working of the algorithm. We merge the two components of a mixture model having 7 components to generate a mixture model with 6 components. We again merge the two components in a mixture model having 6 components to obtain a mixture model with 5 components. We repeat the procedure of merging of mixture components until we have a mixture model with a single mixture components. This progressive training results in a series of models of different complexities. We can use model validation techniques such as cross-validation with this strategy to select the optimal number of mixture components.

4.1 DNA copy number aberration dataset

We use two DNA copy number aberration datasets in the experiments. In the first data set, there are 393 different parts in the genome (data dimension, $d = 393$) (Hollmén and Tikka 2007; Myllykangas et al. 2008). Genome in the second data has 862 different parts (data dimension, $d = 862$) (Baudis 2007). We then transform the two available datasets to 0–1 matrix where rows denote the cancer patients, and columns denote the chromosomal bands. $x_i = 1$ indicates that the chromosome band indexed by i for the cancer patient is aberrated whereas $x_i = 0$ indicates that the chromosome band is unaberrated.

Both the datasets used in the experiments have a limited number of samples; the first data consists of 4590 samples and the second data contains approximately 4000 samples. Thus, we perform chromosomewise mixture modelling because of the scarcity of data samples to constrain the complexity of mixture models. Mixture models trained with a small number of samples are susceptible to over–fitting and under-fitting. Unlike the mixture models of Bernoulli distributions, in Gaussian mixture model, higher number of mixture components sometimes helps in modelling the tails of the distributions.

When we divide the genome into chromosomes for chromosomewise analysis, each chromosome will have different number of chromosome bands, and different dimensionality. The reduced dimensionality will be considerably smaller than the original dimensionality of data. This decrease in dimensionality ameliorates the problem of curse of dimensionality (Donoho 2000). Chromosomewise mixture modelling will be computationally easier as the largest dimensionality is 63 (Chromosome 1) compared to the dimensionality of 862 for whole genome. Similarly, the smallest dimension is 8 (Chromosome 21) when we divide the genome with dimensionality 393 into different chromosomes. Furthermore, studying each chromosome separately can provide new insights into data; for example, chromosome specific patterns (Adhikari and Hollmén 2010a; Myllykangas et al. 2008) (Fig. 5).

We use a backward search based strategy to search for the optimal number of mixture components. We need to select the number of components based on their generalization performance on the future unseen data. Thus, we use the ten-fold cross-validation to train the model of each complexity where the number of components varies from 1 to 20. For each complexity, we obtain the initialization model by merging the two components of mixture model having one more mixture component.

Figure 6 shows that the likelihood smoothly decreases when the number of components decreases, i.e., increases with an increasing number of mixture components. It also shows that the increase in the likelihood with the increasing number of components is initially steep and then flattens out after a certain number

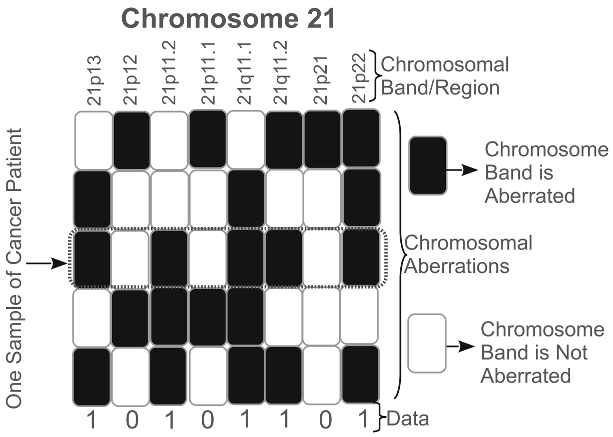

Fig. 5 Visualization of the dataset where rows are the cancer patients and the spatial co-ordinates on the X-axis are the chromosomal band. *Darker color* defines that the chromosomal band is aberrated and *lighter color* determines that chromosomal band is not aberrated. Figure shows only 5 samples in chromosome 21 for clarity

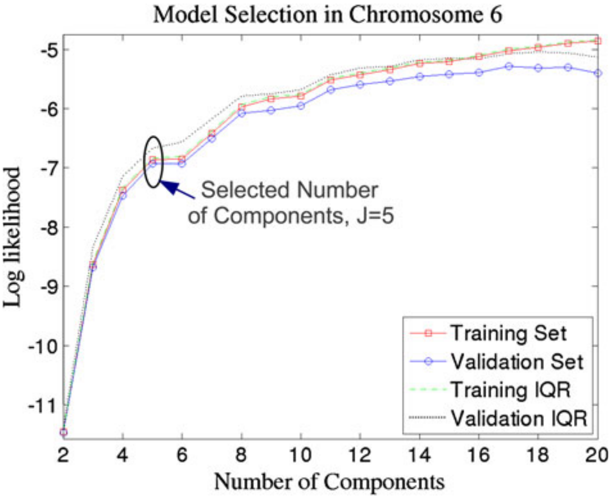

Fig. 6 An example of ten-fold cross-validation for model selection in Chromosome 6 in chromosomal aberrations data with dimensionality 393. The figure depicts averaged log-likelihood for training and validation sets. The interquartile range (IQR) for 10 different training and validation runs in ten-fold cross-validation setting have also been plotted. Here, number of components (J) selected is 5

of components. The figure also shows that the increase in the likelihood after the number of components is six is negligible considering the increase in the complexity of the model. Thus, we select 6 as the final number of components.

We also studied the changes in the likelihood values after merging the components and also after retraining the mixture model with the merged components. The results reported in the Fig. 7 shows that the log-likelihood decreases after merging of mixture components and increases after retraining it. The increase in likelihood obtained by retraining is unable to exactly compensate original value of likelihood of the mixture model having higher number of mixture components. However, figure shows that we achieve considerable improvement in log-likelihood values after retraining the mixture model obtained by merging of the mixture components. The improvement is greater in components 3 to 10 which are more likely to be the number

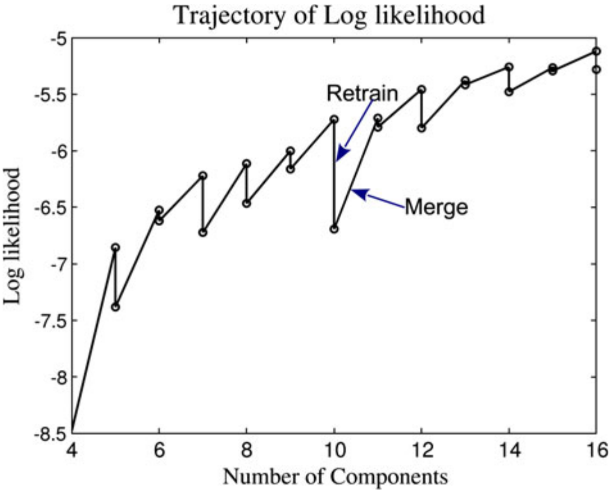

Fig. 7 An example of the trajectory of the log-likelihood in Chromosome 6 in chromosomal aberrations dataset with dimensionality 393. The figure shows log-likelihood on the whole data by the merged mixture model obtained by merging of the mixture components and the trained mixture model which is initialized using the merged mixture model. The model selection for the same data is shown in Fig. 6

of components selected for the data. Improvement decreases after the number of mixture components are greater than 10 showing that the models having more than 10 components may over-fit the data. Similarly, the improvement is unnoticeable when the number of components is less than 3 because the model may have underfit. This repetitive training is one of the advantages of our proposed method over hierarchical agglomerative clustering.

4.2 Artificial dataset

Mixture models are generative models thus offering the facility to generate data samples from the model. This makes it easier to generate datasets with known number of components. In the experiments, we train the mixture model with six components and generate 3000 data points from the mixture model because our dataset of interest DNA Copy Number Aberration dataset contains similar number of samples. We also apply our algorithm of fast progressive training of mixture models on artificial data to determine if it correctly identifies six components that truly generated the data.

Left panel of Fig. 8 shows that our fast progressive training approach correctly identifies the six components present in the artificial data. Unlike the experiments with real–world datasets in Section 4.1 where we had no prior information about the number of components generating the data thus model selection is complicated for Chromosomal Aberration Datasets in Section 4.1. Nevertheless, we also experimented with adding noise (5 % and 10 %) to the artificial dataset. We flip the individual data elements from 1 to 0, and 0 to 1 of 5 % or 10 % of data elements to add 5 % or 10 % noise. The effect of noise is inconsiderable the algorithm from determining the true number of data generating components.

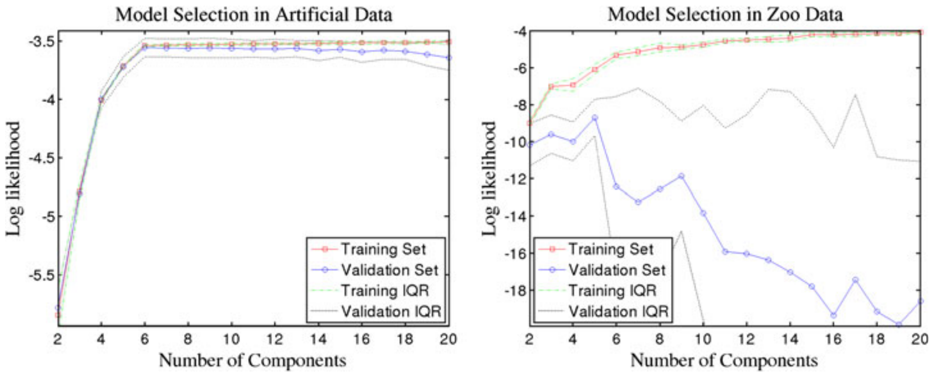

Fig. 8 *Left panel* shows ten-fold cross-validation for model selection in an artificial dataset. The figure depicts averaged log-likelihood for training and validation sets. The interquartile range (IQR) for 10 different training and validation runs in ten-fold cross-validation setting have also been plotted. Here, 6 components (J) generated the data and cross-validation results shows that we would have picked 6 components. Similarly, *right panel* shows ten-fold cross-validation for model selection in publicly available zoo dataset. The figure depicts averaged log-likelihood for training and validation sets. The interquartile range (IQR) for 10 different training and validation runs in ten-fold cross-validation setting have also been plotted. Here, the validation likelihood shows peaks at 3, 5 and 8 components. With background information on dataset, we would select 8 components

4.3 Publicly available zoo dataset

We also experimented our algorithm on a publicly available zoo dataset from UCI Machine Learning Repository (Bache and Lichman 2013). The dataset consists of 18 features. We ignore the first attribute out of the 18 features which is the unique animal name. Similarly, the last attribute is a categorical attribute that determines the cluster indices. The remaining 15 attributes of the 16 are binary attributes. The remaining one final attribute is a numerical attribute describing number of legs of the animal. We performed couple of experiments. Firstly, removing the numerical attribute completely. Secondly, changing the numerical attribute to six binary attribute as in Li (2005). The six attribute corresponds to the presence of 0, 2, 4, 5, 6, and 8 legs.

As in Li (2005), our algorithm identifies 5 clusters present in the zoo dataset. However, there are 7 clusters in the dataset. Right panel of the Fig. 8 shows high variation in likelihood as shown by the IQR curve when the number of components is 7. So, if we have a prior knowledge about the number of clusters, we can essentially select seven components because in different runs of the experiments maximum validation likelihood is when the number of components is 7. Furthermore, we perform experiments in five-fold cross-validation setting. The results in both the experiments were similar but sometimes inconsistent. One of the reasons for variation in results across different runs of cross–validation setting is because the number of samples 100 is considerably less to train the complexity of mixture models.

4.4 Improvement on previous model selection methods

We estimated the time required to compute our approximation of the KL divergence and also that of the full KL divergence to show the performance improvement with regards to the approximation of the KL divergence. The results reported in Table 1 shows that our approximation is considerably faster than the full KL divergence. Furthermore, it was computationally infeasible to calculate full KL divergence in the data of dimensionality greater than 20.

One of the major benefits of merging the mixture components is the faster convergence of the EM algorithm. The Fig. 3 shows that initialization model obtained by merging two components will be almost similar to the final model trained with six components. Therefore, number of iterations of the EM algorithm required to reach the convergence is considerably less. For example, in chromosome 21 in Dataset 1, on an average of 100 runs, a random model requires 47 iterations in the EM algorithm to converge while it requires only 1 iteration to converge from the initialized model. Similarly, on an average of 100 runs, it takes approximately 2.16 and 28.32 seconds to train a mixture model with the merged initialization and a random initialization respectively.

We used random initialization and repeated the experiments 50 times to ameliorate the local optima problem of the EM algorithm in our previous works in Tikka et al. (2007), Hollmén and Tikka (2007), Adhikari and Hollmén (2010a, b). Here we avoid 50 repeats because the EM algorithm is deterministic for the same data with the same initialization. For previous methods, in a ten-fold cross-validation setting with 20 components, we need to train $20 \times 10 \times 50 = 10000$ models whereas in this situation, we train only $20 \times 10 = 200$ different models. In other words, the current proposed method makes one pass through the ten-fold cross validation setting whereas previous methods make 50 different passes through the ten-fold cross validation setting.

The advantage of our algorithm is production of similar models having different number of components by merging similar components. So, our algorithm compares similar models but with different number of components which makes the results more reliable as it helps us to find the accurate number of components. This avoids the situation where by random chance a model for accurate number of components gets stuck in local optima and with inaccurate number of components reaches the global optima. In that situation, we select inappropriate number of components. Although, our algorithm can get stuck in local optima, all models across all components gets stuck in similar local optima thus helping us select accurate number of components.

The additional overhead in the proposed method is the calculation of the KL divergence but as shown in Table 1, the approximations takes less than one-tenth of a second. Furthermore, yet another disadvantage of the merged initialization is that the EM algorithm can get stuck in a local optimum. However, we try to alleviate the problem by initially selecting the best of 100 models.

5 Summary and conclusions

In this paper, we proposed fast progressive training of mixture models to help model selection algorithms determine the number of mixture components by merging the mixture components. In the context of selecting which component distributions to merge, we proposed a fast, data driven approximation of the symmetrized KL divergence to calculate the similarity between two mixture components. Initially, we begin by selecting high number of mixture components and then progressively merge the similar components until the number of components is 1. We can use any model validation technique such as cross-validation, AIC, BIC, and MDL as a criterion for model selection in conjunction with our strategy. We experiment the proposed algorithm on two chromosomal aberration patterns data in cancer genomics showing that the our strategy produces plausible results. The proposed strategy is computationally efficient considering the well known pitfalls of methods using backward search strategy.

Acknowledgements Helsinki Doctoral Programme in Computer Science—Advanced Computing and Intelligent Systems (Hecse), and Finnish Center of Excellence for Algorithmic Data Analysis (ALGODAN) funds the current research.

References

Adhikari, P.R., & Hollmén, J. (2010a). Patterns from multi-resolution 0–1 data. In B. Goethals, N. Tatti, J. Vreeken (Eds.) *Proceedings of the ACM SIGKDD workshop on useful patterns (UP'10)* (pp. 8–12). ACM.

Adhikari, P.R., & Hollmén, J. (2010b). Preservation of statistically significant patterns in multiresolution 0–1 data. In T. Dijkstra, E. Tsivtsivadze, E. Marchiori, T. Heskes (Eds.) Pattern recognition in bioinformatics. *Lecture notes in computer science* (Vol. 6282, pp. 86–97). Berlin/Heidelberg: Springer.

Adhikari, P.R., & Hollmén, J. (2012). Fast progressive training of mixture models for model selection. In J.-G. Ganascia, P. Lenca, J.-M. Petit (Eds.) *Proceedings of fifteenth international conference on discovery science (DS 2012). LNAI* (Vol. 7569, pp. 194–208). Springer-Verlag.

Akaike, H. (1974). A new look at the statistical model identification. *IEEE Transactions on Automatic Control, 19*(6), 716–723.

Bache, K., & Lichman, M. (2013). *UCI machine learning repository*. University of California, Irvine, School of Information and Computer Science. http://archive.ics.uci.edu/ml.

Baudis, M. (2007). Genomic imbalances in 5918 malignant epithelial tumors: an explorative meta-analysis of chromosomal CGH data. *BMC Cancer, 7*, 226.

Beeferman, D., & Berger, A. (2000). Agglomerative clustering of a search engine query log. In *Proceedings of the ACM KDD '00, New York, USA* (pp. 407–416).

Blekas, K., & Lagaris, I.E. (2007). Split-merge incremental learning (SMILE) of mixture models. In *Proceedings of the ICANN'07* (pp. 291–300). Springer-Verlag.

Cai, H., Kulkarni, S.R., Verdú, S. (2006). Universal divergence estimation for finite-alphabet sources. *IEEE Transactions on Information Theory, 52*(8), 3456–3475.

Cover, T.M., & Thomas, J.A. (1991). *Elements of information theory*. New York: Wiley-Interscience.

Dempster, A.P., Laird, N.M., Rubin, D.B. (1977). Maximum likelihood from incomplete data via the EM algorithm. *Journal Of The Royal Statistical Society, Series B, 39*(1), 1–38.

Donoho, D.L. (2000) High-dimensional data analysis: the curses and blessings of dimensionality. Aide–Memoire of a lecture. In *AMS conference on math challenges of the 21st century*.

Everitt, B.S., & Hand, D.J. (1981). *Finite mixture distributions*. London, New York: Chapman and Hall.

Figueiredo, M.A.T, & Jain, A.K. (2002). Unsupervised learning of finite mixture models. *IEEE Transactions on Pattern Analysis Machicne Intelligence, 24*(3), 381–396.

Goldberger, J., Gordon, S., Greenspan, H. (2003). An efficient image similarity measure based on approximations of KL-divergence between two Gaussian mixtures. In *Proceedings of the ICCV '03, Washington DC, USA* (pp. 487–493).

Hershey, J.R., & Olsen, P.A. (2007). Approximating the Kullback Leibler divergence between Gaussian mixture models. In *IEEE. ICASSP 2007* (Vol. 4, pp. 317–320).

Juang, B.H., & Rabiner, L.R. (1985). A probabilistic distance measure for Hidden Markov models. *AT&T Technical Journal, 64*(2), 391–408.

Hollmén, J., & Tikka, J. (2007). Compact and understandable descriptions of mixture of Bernoulli distributions. In M.R. Berthold, J. Shawe-Taylor, N. Lavrač (Eds.) *Proceedings of the IDA 2007. LNCS* (Vol. 4723, pp. 1–12).

Kittler, J. (1986). *Feature selection and extraction. Handbook of pattern recognition and image processing.*. Academic Press.

Kullback, S., & Leibler, R.A. (1951). On information and sufficiency. *Annals of Mathematical Statistics, 22*(1), 79–86.

Lee, Y.K., & Park, B.U. (2006). Estimation of Kullback–Leibler divergence by local likelihood. *Annals of the Institute of Statistical Mathematics, 58*, 327–340.

Leonenko, N., Pronzato, L., Savani, V. (2008). A class of Rényi information estimators for multidimensional densities. *Annals of Statistics, 36*(5), 2153–2182.

Li, T. (2005). A general model for clustering binary data. In *Proceedings of the eleventh ACM SIGKDD international conference on knowledge discovery in data mining, KDD '05* (pp. 188–197). ACM: New York.

Li, Y., & Li, L. (2009). A novel split and merge EM algorithm for gaussian mixture model. In *Fifth international conference on natural computation, 2009. ICNC '09* (Vol. 6, pp. 479–483).

Li, Y., & Li, L. (2009). A split and merge EM algorithm for color image segmentation. In *IEEE ICIS 2009* (Vol. 4, pp. 395–399).

Mclachlan, G.J., & Krishnan, T. (1996). *The EM algorithm and extensions* (1st ed.). Wiley-Interscience.

McLachlan, G.J., & Peel, D. (2000). *Finite mixture models*. New York: Wiley.

Myllykangas, S., Tikka, J., Böhling, T., Knuutila, S., Hollmén, J. (2008). Classification of human cancers based on DNA copy number amplification modeling. *BMC Medical Genomics, 1*(15), 1–18.

Perez-Cruz, F. (2008). Kullback–Leibler divergence estimation of continuous distributions. In *IEEE international symposium on information theory, ISIT 2008* (pp. 1666–1670).

Smyth, P. (2000). Model selection for probabilistic clustering using cross-validated likelihood. *Statistics and Computing, 10*, 63–72.

Tikka, J., & Hollmén, J. (2008). A sequential input selection algorithm for long-term prediction of time series. *Neurocomputing, 71*(13–15), 2604–2615.

Tikka, J., Hollmén, J., Myllykangas, S. (2007). Mixture modeling of DNA copy number amplification patterns in cancer. In F. Sandoval, A. Prieto, J. Cabestany, M. Graña (Eds.) *Proceedings of the IWANN 2007. Lecture notes in computer science* (Vol. 4507, pp. 972–979). San Sebastián, Spain: Springer-Verlag.

Ueda, N., Nakano, R., Ghahramani, Z., Hinton, G.E. (2000). SMEM algorithm for mixture models. *Neural Computation, 12*(9), 2109–2128.

Wang, Q., Kulkarni, S.R., Verdú, S. (2005). Universal estimation of divergence for continuous distributions via data-dependent partitions. In *Proceedings international symposium on information theory, ISIT 2005* (pp. 152–156).

Windham, M.P., & Cutler, A. (1992). Information ratios for validating mixture analyses. *Journal of the American Statistical Association, 87*(420), 1188–1192.

Wolfe, J.H. (1970). Pattern clustering by multivariate mixture analysis. *Multivariate Behavioral Research, 5*, 329–350.

Zhang, B., Zhang, C., Yi, X. (2004). Competitive EM algorithm for finite mixture models. *Pattern Recognition, 37*(1), 131–144.

Zhang, Z., Chen, C., Sun, J., Chan, K.L. (2003). EM algorithms for Gaussian mixtures with split-and-merge operation. *Pattern Recognition, 36*(9), 1973–1983.