ON INITIALIZATION OF GAUSSIAN MIXTURES: A HYBRID GENETIC EM ALGORITHM

Franz Pernkopf

Graz University of Technology, Laboratory of Signal Processing and Speech Communication
Inffeldgasse 16, A-8010 Graz, Austria, pernkopf@tugraz.at

ABSTRACT

We propose a genetic-based expectation-maximization (GA-EM) algorithm for learning Gaussian mixture models from multivariate data. This algorithm is capable of selecting the number of components of the model using the *minimum description length* (MDL) criterion. We combine EM and GA into a single procedure. The population-based stochastic search of the GA explores the search space more thoroughly than the EM method. Therefore, our algorithm enables to escape from local optimal solutions since the algorithm becomes less sensitive to its initialization. The GA-EM algorithm is *elitist* which maintains the monotonic convergence property of the EM algorithm. The experiments show that the GA-EM outperforms the EM method since: (i) We have obtained a better MDL score while using exactly the same initialization and termination condition for both algorithms. (ii) Our approach identifies the number of components which were used to generate the underlying data more often as the EM algorithm.

1. INTRODUCTION

Finite mixture models [1] are flexible methods for modeling complex probability distribution functions. These models enable statistical modeling of environments with multimodal behavior where simple parametric models fail to represent adequately the characteristics of the data. The standard approach for learning the parameters of the mixture model is the EM algorithm [2]. We develop a novel algorithm for finding the optimal number of components as well as the parameters determining the components of a mixture model. The MDL criterion is used for selecting the number of components of the model. Our approach embeds the EM algorithm in the framework of the GA so that the properties of both algorithms are utilized. The population-based stochastic search of the GA explores the search space more thoroughly than the EM method. Therefore, our algorithm enables to escape from local optimal solutions since the algorithm becomes less sensitive to its initialization.

2. LEARNING GAUSSIAN MIXTURE MODELS

A finite mixture model $p\left(\mathbf{x}|\mathbf{\Theta}\right)$ is the weighted sum of $M > 1$ components $p\left(\mathbf{x}|\theta_m\right)$ in $\mathbb{R}^d$, $p\left(\mathbf{x}|\mathbf{\Theta}\right) = \sum_{m=1}^{M} \alpha_m p\left(\mathbf{x}|\theta_m\right)$, where $\mathbf{x} = \left[x_1, \ldots, x_d\right]^T$ is the d-dimensional data vector, α_m corresponds to the weight of each component $m = 1, \ldots, M$. These weights are constrained to be positive $\alpha_m \geq 0$ and $\sum_{m=1}^{M} \alpha_m = 1$. For Gaussian mixture models, each component $p\left(\mathbf{x}|\theta_m\right)$ is represented as normal distribution, where each component is denoted by the parameters $\theta_m = \{\mu_m, \mathbf{\Sigma}_m\}$, the mean vector and the covariance matrix. The Gaussian mixture is specified by the set of parameters $\mathbf{\Theta} = \{\alpha_1, \alpha_2, \ldots, \alpha_M, \theta_1, \theta_2, \ldots, \theta_M\}$.

The EM algorithm [2] consists of an *expectation* step (E-step) and an *maximization* step (M-step) which are alternately used until the $\log p\left(\mathcal{X}|\mathbf{\Theta}\right) = \log \prod_{i=1}^{N} p\left(\mathbf{x}^i|\mathbf{\Theta}\right)$ converges to a local optimum, where $\mathcal{X} = \left\{\mathbf{x}^1, \mathbf{x}^2, \ldots, \mathbf{x}^N\right\}$ are N i.i.d. samples. The performance of the EM algorithm depends strongly on the choice of the initial parameters $\mathbf{\Theta}^{t=0}$. Different initialization strategies are given in [1].

E-step: The data $\mathcal{X}$ are assumed to be incomplete and the complete data set $\mathcal{Y} = (\mathcal{X}, \mathcal{Z})$ is determined by estimating the set of variables $\mathcal{Z} = \{\mathbf{z}_1, \mathbf{z}_2, \ldots, \mathbf{z}_M\}$, where each $\mathbf{z}_m$ is an N-dimensional vector $\left[z_m^1, z_m^2, \ldots, z_m^N\right]^T$. The log likelihood of the complete data $\mathcal{Y}$ is

$$\log p\left(\mathcal{Y}|\mathbf{\Theta}\right) = \sum_{i=1}^{N} \sum_{m=1}^{M} z_m^i \log\left[\alpha_m p\left(\mathbf{x}^i|\theta_m\right)\right], \qquad (1)$$

where z_m^i is the posterior probability

$$z_m^i = P\left(m|\mathbf{x}^i, \mathbf{\Theta}^t\right) = \frac{\alpha_m^t p\left(\mathbf{x}^i|\theta_m^t\right)}{\sum_{l=1}^{M} \alpha_l^t p\left(\mathbf{x}^i|\theta_l^t\right)} \qquad (2)$$

and $\mathbf{\Theta}^t$ is the parameter estimate obtained after t iterations.

M-step: In this step the parameters $\mathbf{\Theta}^{t+1}$ are determined according to the estimate of the variables z_m^i. For Gaussian mixture models this corresponds to reestimating the α_m^{t+1},

the μ_m^{t+1}, and $\mathbf{\Sigma}_m^{t+1}$ for each m according to

$$\alpha_m^{t+1} = \frac{1}{N}\sum_{i=1}^{N} z_m^i, \qquad \mu_m^{t+1} = \frac{\sum_{i=1}^{N} z_m^i \mathbf{x}_i}{\sum_{i=1}^{N} z_m^i}, \text{and} \quad (3)$$

$$\mathbf{\Sigma}_m^{t+1} = \frac{\sum_{i=1}^{N} z_m^i \left(\mathbf{x}_i - \mu_m^{t+1}\right)\left(\mathbf{x}_i - \mu_m^{t+1}\right)^T}{\sum_{i=1}^{N} z_m^i}. \quad (4)$$

3. MODEL SELECTION CRITERION: MDL

Different approaches for model selection have been proposed in [1]. The MDL criterion

$$MDL = -\log p\left(\mathcal{X}|\mathbf{\Theta}\right) + \frac{M\left(L+1\right)}{2}\log N \quad (5)$$

is the most commonly used selection criterion, where L is the number of parameters defining each component (for Gaussian mixture models $L = d + d\left(d+1\right)/2$). Equation 5 has the intuitive interpretation that the log likelihood $-\log p\left(\mathcal{X}|\mathbf{\Theta}\right)$ is the code length of the *encoded* data. The term $\frac{M(L+1)}{2}\log N$ models the optimal code length for all parameters.

4. GENETIC-BASED EM ALGORITHM (GA-EM)

The main goal of interweaving GA [3], [4] with the EM algorithm is to utilize the properties of both algorithms. In our GA-EM algorithm, each individual in the population represents a possible solution of the Gaussian mixture model. The MDL criterion (see Section 3) is used as a fitness function for model selection. The best individual is the one that has the *lowest* MDL value. The evaluation of the individuals in the population is two-fold. Firstly, R cycles of the EM algorithm are performed on each individual which results in an update of the set of parameters $\mathbf{\Theta}^t$ (at iteration t) and consequently of the individual which encodes the parameters. Secondly, the MDL value is determined according to Equation 5 from each updated individual to judge the model. Hence, the evaluation process of the individual provides both, a fitness value and an update of the parameters encoded by the individual. To maintain the monotonic convergence property [5], we extended our GA-EM so that it is elitist which means that the best individual of the current generation is copied unaltered to the next generation. Thus, the mixing weights α_m of the best individual have to be saved for the subsequent generation. This mechanism guarantees that the best member of the population at generation $t+1$ does not perform worse than the best individual at generation t. The evolution process of the GA-EM is terminated when the number of components used by the best model does not change within five consecutive generations. Once the evolution is stopped, the EM algorithm is used to improve the best individual $\mathbf{a}_{min}$ found so far until the relative log likelihood of the mixture model $\left|\frac{\log p\left(\mathcal{X}|\mathbf{\Theta}^t\right) - \log p\left(\mathcal{X}|\mathbf{\Theta}^{t+1}\right)}{\log p\left(\mathcal{X}|\mathbf{\Theta}^t\right)}\right|$ drops below a certain threshold ϵ (e.g., $\epsilon = 0.00001$).
In the following, the GA-EM algorithm is presented.

procedure GA-EM
begin
  $t \longleftarrow 0$
  $OldSize \longleftarrow 0$
  $c_{end} \longleftarrow 0$
  Initialize $P\left(t\right)$
  while $(c_{end} \neq 5)$
    $P\left(t\right)' \longleftarrow$ perform R EM steps on $\left(P\left(t\right)\right)$
    $MDL' \longleftarrow$ evaluate $\left(P\left(t\right)'\right)$
    $P\left(t\right)'' \longleftarrow$ recombine $\left(P\left(t\right)'\right)$
    $P\left(t\right)''' \longleftarrow$ perform R EM steps on $\left(P\left(t\right)''\right)$
    $MDL'' \longleftarrow$ evaluate $\left(P\left(t\right)'''\right)$
    $\left[P\left(t\right)'''', MDL\right] \longleftarrow$ select
      $\left[\left(P\left(t\right)''', MDL''\right) \cup \left(P\left(t\right)', MDL'\right)\right]$
    $MDL_{min} \longleftarrow \min\left(MDL\right)$
    $\mathbf{a}_{min} \longleftarrow \arg\min_{MDL}\left(P\left(t\right)''''\right)$
    if $(|\mathbf{a}_{min}| \neq OldSize)$ **then**
      $c_{end} \longleftarrow 0$
      $OldSize = |\mathbf{a}_{min}|$
    else
      $c_{end} \longleftarrow c_{end} + 1$
    end
    $P\left(t\right)''''' \longleftarrow$ enforce mutation $\left(P\left(t\right)''''\right)$
    $P\left(t+1\right) \longleftarrow$ mutate $\left(P\left(t\right)'''''\right)$
    $t \longleftarrow t+1$
  end
  EM$(\mathbf{a}_{min})$ until convergence of the log likelihood.
end

The best evaluation value achieved during the evolution process is stored in MDL_{min} and the corresponding individual in $\mathbf{a}_{min}$, where $|\mathbf{a}_{min}|$ denotes the number of components used for this model. $P\left(t\right)$ denotes a population of K individuals at generation t and $P\left(t\right)'$ is the resulting population after performing R EM steps. $P\left(t\right)''$ is an offspring population of $P\left(t\right)'$ with size H. Performing the EM steps and evaluation of the offspring population delivers $P\left(t\right)'''$ and MDL''. In the following, the parameters and operators of the GA-EM are discussed in more detail.

Encoding: Each individual is composed of two parts. The first part (Part A) uses binary encoding, where the length of this part is determined by the maximal number of allowed components M_{max}. Each of these bits is related to a particular component. If a bit is set to zero, then its associated component is omitted for modeling the mixture, while setting the bit to one includes the component. The second part (Part B) uses floating point value encoding to encode the

mean μ_m and covariance $\mathbf{\Sigma}_m$ parameters of M_{max} components. Each component uses $L = d + d\,(d+1)\,/2$ parameters. Due to the switching mechanism of the components among the individuals during evolution of the GA-EM, the component weight α_m cannot be encoded. Except for the best individual, these weights are assumed to be uniformly distributed.

Recombination: The crossover operator selects two parent individuals randomly from the population $P\,(t)'$ and recombines them to form two offsprings. The crossover probability p_c determines the number of offsprings H ($H = p_c K$). We use the *Single-point crossover* [3], [4] which chooses randomly a crossover position $\chi \in \{1, \ldots, M_{max}\}$ within part A of the individual and exchanges the value of the genes to the right of this position between both individuals for part A with its associated parameters in part B.

Selection: For selection the (K, H)-strategy [6] is used. This approach refers to both the parent population $P\,(t)'$ and the offspring population $P\,(t)'''$ containing K and H individuals, respectively. After both populations have been evaluated the K best individuals are selected to form the population $P\,(t)''''$ for the next generation.

Enforced Mutation: If more components model the data points in a similar manner some of their parameters are forced to mutate. This similarity is measured using the correlation coefficient r_{jk} which is computed pairwise between the components j and k ($1 \leq j, k \leq M, j > k$) from the posterior probability $\mathbf{z}_j$ and $\mathbf{z}_k$. If the correlation coefficient is above the threshold $t_{Correlation} < |r_{jk}|$, one of both components is randomly selected and added to the candidate set for mutation. Once the candidate set for enforced mutation is complete, a binary value is sampled from an uniform distribution for each candidate. According to this value, either the candidate component is removed by resetting the corresponding bit in part A of the individual or a randomly chosen data point is assigned as new mean value.

Mutation: The mutation operator inverts the binary value of each gene in part A of the individuals with the mutation probability p_m. For part B of the individual an uniform distributed random number sampled within an upper and lower bound is assigned to genes that are mutated. These bounds are determined from the data set. The mutation rate for value encoding is scaled down by a factor of L, i.e. $\frac{p_m}{L}$. The mutation for the value encoded part of the individual is restricted to the mean values. Since our GA-EM is elitist, there are no mutations performed on the best individual.

5. EXPERIMENTS: EM VERSUS GA-EM

We use two initialization methods in the experiments: (i) A variant with random starting values: The covariance matrix is initialized in a similar manner as in [7]. The mean values of the components $\mu_m^{t=0}$ are set to randomly selected data points. The weights $\alpha_m^{t=0}$ of the components are assumed to be uniformly distributed. (ii) k-means clustering [8]: The parameters of the selected components are initialized by the k-means algorithm. All unselected components are initialized to random starting values as described above.

For the GA-EM the start population is comprised of a set of individuals, where each individual has a different number of selected components. Hence, the start population $P\,(0)$ consists of $\max\{M_{max}, K\}$ individuals. The number of individuals in subsequent populations is restricted to K.

If a component m is not supported by the data, the component is annihilated. This is the case when the sum of the posterior probability z_m^i over all data points is below a threshold expressed as $\sum_{i=1}^{N} z_m^i < t_{Annihilate}$. A reasonable threshold depends on the dimension d of the data.

Data sets with a dimension of $d \in \{2, 5, 10\}$ have been generated, whereby the sample size N varies with the number of components M according to $N = 300M$. The weight of each component α_m is selected randomly, whereby it is guaranteed that $\alpha_m > \frac{1}{2M}, \forall m = 1, \ldots, M$. The data were drawn from a mixture of Gaussian distribution with a different number of components $M \in \{3, 5, 9, 12\}$. Additionally, the minimum separation between the components were determined to be $c \in \{0.8, 1.0, 1.2, 1.4, 1.6, 1.8, 2.0\}$. Dasgupta [9] defines that two Gaussians are c-separated if $\| \mu_1 - \mu_2 \|_2 \geq c\sqrt{d \max\left(\lambda_{max}\left(\mathbf{\Sigma}_1\right), \lambda_{max}\left(\mathbf{\Sigma}_2\right)\right)}$, where $\lambda_{max}\left(\mathbf{\Sigma}\right)$ denotes the largest eigenvalue of $\mathbf{\Sigma}$. A mixture of components is considered to be c-separated if the components are pairwise c-separated. We generated 50 data sets for each configuration of M, c, and d. The maximum number of Gaussian components in the data is assumed to be $M_{max} = 15$ for the EM and the GA-EM algorithm. The parameter setting for the GA-EM is $p_m = 0.02$ for the mutation probability, $p_c = 0.8$ for the recombination probability, $K = 6$ for the population size, $R = 3$ for the number of EM steps within one GA generation, and $t_{Correlate} = 0.95$ for the component correlation threshold. The EM algorithm is executed for 2 to M_{max} components. The selected model is the one that achieves the lowest MDL value within the set of obtained candidate models. It is assumed that the proper number of components lies in the given range of $[2..M_{max}]$.

In Figure 1 both algorithms are compared with respect to the achieved average MDL criterion (see column (a)), the average number of EM steps used to establish the model (see column (b)), and the percentage of the correctly identified number of components (see column (c)) which were used to generate the data set. The x-axis represents the value of c-separation. The rows of the figure correspond to the different number of components used for generating the data. GA-EM1 and EM1 use random starting values and GA-EM2 and EM2 are initialized using the k-means algorithm. Figure 1 shows the results only for the dimension

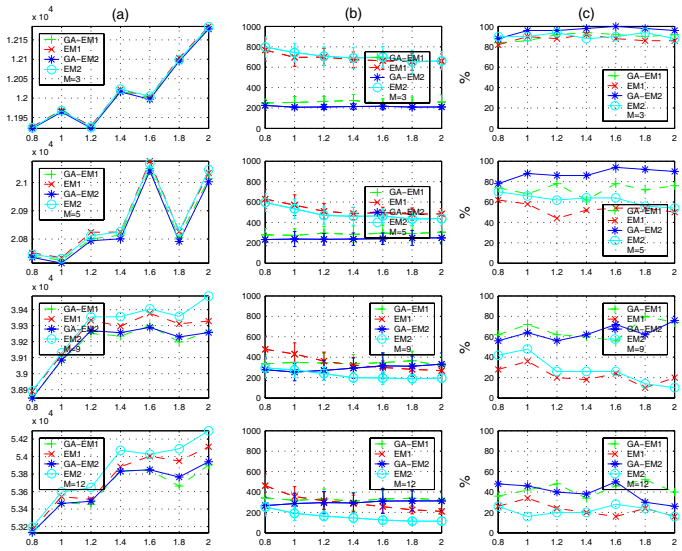

Fig. 1. Comparison of EM and GA-EM: EM1 and GA-EM1 use random starting values and EM2 and GA-EM2 use the k-means algorithm for initialization. Column (a): Average achieved MDL, Column (b): Average number of required EM steps, Column (c): Percentage of correctly identified number of components.

$d = 5$. The performance for $d = 2$ and $d = 10$ is similar. Further experiments on simulated and real data are in [10].

Average MDL score (see column (a)): Since both algorithms use the same termination condition, the obtained MDL score for selecting the finite mixture model is similar. However, especially for larger number of components M the GA-EM algorithm yields a better score. This fact is accredited to the dependency of the EM to the initialization. The population-based stochastic search behavior of the GA-EM explores the search space more thoroughly. This enables to escape from local optimal solutions since the algorithm becomes less sensitive to its initialization.

Average number of required EM steps (see column (b)): The GA-EM converges faster than the EM algorithm for a small number of components M. The EM converges faster by increasing the separation of the components, whereby, the GA-EM is almost independent to this change. For $M = 9$ both algorithms require approximately the same number of EM steps when initialized with random starting values. Using k-means initialization speeds up the convergence of the EM algorithm, especially for a large number of components in the underlying data. Note that for the GA-EM algorithm the computational costs required for the genetic operators such as recombination, mutation and selection are neglected.

Correctly identified number of components (see column (c)): The GA-EM is more often identifying the correct number of components which were used for sampling the data. For a small number of components $M = 3$ both algorithms work well. However, for an increasing number of components the GA-EM is able to identify the correct number of producing components more often.

6. CONCLUSION

This paper proposes a genetic-based EM algorithm for learning Gaussian mixture models from multivariate data. This algorithm is capable of selecting the number of components based on the MDL criterion. Our approach is less sensitive to the initialization compared to the standard EM algorithm. This is attributed to population-based search behavior of the GA-EM which explores the parameter space more thoroughly. Since the GA-EM is elitist it maintains the monotonic convergence property of the EM algorithm. The experiments demonstrated that our algorithm outperforms the EM algorithm. In fact, we have obtained a better MDL score while using exactly the same initialization and termination condition for both algorithms. Additionally, the number of components which were used to generate the underlying data were more often correctly identified compared to the EM algorithm. However, one drawback of the GA-EM algorithm is that it requires additional parameters.

7. REFERENCES

[1] G. McLachlan and D. Peel, *Finite mixture models*, John Wiley & Sons, 2000.

[2] A. Dempster, N. Laird, and D. Rubin, "Maximum likelihood estimation from incomplete data via the EM algorithm," *Journal of the Royal Statistic Society*, vol. 30, no. B, pp. 1–38, 1977.

[3] T. Bäck, *Evolutionary algorithms in theory and practice*, Oxford University Press, 1996.

[4] Z. Michalewicz and D.B. Fogel, *How to solve it: Modern heuristics*, Springer Verlag, 2000.

[5] L. Xu and M.I. Jordan, "On convergence properties of the EM algorithm for Gaussian Mixtures," *Neural Computation*, vol. 8, pp. 129–151, 1996.

[6] T. Bäck and H. Schwefel, "Evolutionary computation: An overview," in *IEEE Conference on Evolutionary Computation*, 1996, pp. 20–29.

[7] M.A.T. Figueiredo and A.K. Jain, "Unsupervised learning of finite mixture models," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 3, pp. 1–16, 2002.

[8] R.O. Duda, P.E. Hart, and D.G Stork, *Pattern classification*, John Wiley & Sons, 2000.

[9] S. Dasgupta, "Learning mixtures of Gaussian," in *IEEE Symp. on Foundations of Computer Science*, 1999, pp. 634–644.

[10] F. Pernkopf, "Genetic-based EM algorithm for component selection and parameter estimation of gaussian mixture models," Tech. Rep., Graz University of Technology, 2004.