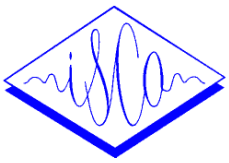

4th European Conference on
Speech Communication and Technology
EUROSPEECH '95
Madrid, Spain, September 18-21, 1995

MULTI-VARIATE MIXTURE PROBABILITY DENSITY MODELLING OF VQ CODEBOOK USING GRADIENT DESCENT ALGORITHM*

S. Dobrišek & F. Mihelič & N. Pavešić
e-mail: simond@fer.uni-lj.si
Faculty of Electrical Engineering & Computer Science
University of Ljubljana
Tržaška cesta 25
SI-61000 Ljubljana
SLOVENIA

ABSTRACT

A vector quantisation codebook can be modelled as a set of probability density functions. The problem of estimating the parameters determining mixture probability density models can be solved using a log-likelihood based reestimation procedure. On the other hand, this problem can be also viewed as a conventional optimisation problem. Consequently, gradient descent techniques may be used to obtain values of the model parameters. The main advantage of these techniques over the reestimation procedure is higher robustness due to an initial estimation of the model parameters. In the paper, we describe a descent algorithm along with a criterion function, we propose. We obtained some promising results by applying this algorithm to one and two-variate pseudo Gaussian mixture probability density functions and further to signal vectors of a continuous speech database.

1. INTRODUCTION

Most of the conventional vector quantisation (VQ) methods partition the feature space into separate regions, according to some distortion measures [4], regardless of the feature vectors' probability distributions [8]. This introduces errors in the partition operations which can not be avoided [5]. As an alternative, we can model a VQ codebook as a set of probability density functions (*pdf*) [5, 9]. In this case the representation of the vector quantisation codeword in the feature space is calculated as a *pdf*.

The problem of estimating the parameters determining a mixture *pdf* model, is to be solved. This can be done using a log-likelihood based reestimation procedure. The most distinctive solution to this problem is the procedure, known as Estimation-maximisation (EM) algorithm [5, 1]. However, it has a significant drawback, as we need to obtain sufficiently good initial estimates for the unknown model parameters [8].

The problem of estimating the model parameters can be also set up as a conventional optimisation problem. Thus, rather standard gradient techniques can be used to obtain the values of the model parameters [3]. In the following sections, we go on describing a descent algorithm, minimising the difference between the current estimation and the moving average estimation of the model parameters. This algorithm has proved to be very robust due to the initial estimations of the model parameters, and the current parameter estimations can be derived promptly at each step of the algorithm. This feature turns out to be of great use when dealing with a huge training set like a feature vectors set, derived from a continuous speech signal.

*This work was partly funded by the Comission of the European Community under COP-94 contract No 01634 (SQEL)

2. MIXTURE PDF MODEL

Suppose that we have a set of N observable feature vectors $\mathcal{X} = \{\mathbf{x}_1, \cdots, \mathbf{x}_N\}$. The total likelihood of the observable vectors is by definition the joint probability

$$p(\mathcal{X}|\boldsymbol{\theta}) = \prod_{j=1}^{N} p(\mathbf{x}_j|\boldsymbol{\theta}), \tag{1}$$

where $\boldsymbol{\theta}$ is the model parameter vector. In the mixture *pdf* with L component densities c_i, the above conditional *pdf* equals

$$p(\mathbf{x}_j|\boldsymbol{\theta}) = \sum_{i=1}^{L} p(c_i)p(\mathbf{x}_j|c_i, \boldsymbol{\theta}_i), \tag{2}$$

where $i = 1, \cdots; L$ denotes a component density label.

Further, the a posteriori probability, that the vector $\mathbf{x}_j$ occurs from the component density c_i is given as

$$p(c_i|\mathbf{x}_j) = \frac{p(c_i)p(\mathbf{x}_j|c_i, \boldsymbol{\theta}_i)}{p(\mathbf{x}_j|\boldsymbol{\theta})}. \tag{3}$$

The mixture *pdf* model is defined by the parameter vector $\boldsymbol{\theta}$ comprising all component density parameter vectors $\boldsymbol{\theta}_i$ and the a priori probabilities $p(c_i)$. The representation of the VQ codeword in the feature space is calculated as the *pdf* according to the Bayesian decision rule [8].

Our task is to estimate all component values of the model parameter vector $\boldsymbol{\theta}$. This problem can be set up as a conventional optimisation problem. Therefore, we decided to use and to evaluate a standard gradient descent technique. In comparison to the log-likelihod based reestimation procedure, we anticipated a higher robustness fo our model, due to the initial estimations of the model parameter vector.

3. GRADIENT DESCENT ALGORITHM

The basic idea of the gradient descent procedure is very simple. We start with an arbitrary estimation of a parameter vector $\boldsymbol{\theta}'(0)$. The next value $\boldsymbol{\theta}'(1)$ is obtained by moving some distance away from $\boldsymbol{\theta}'(0)$ in the direction of the

steepest descent, i.e., along the negative of the gradient. In general, $\boldsymbol{\theta}'(k+1)$ is obtained from $\boldsymbol{\theta}'(k)$ by the algorithm

$$\boldsymbol{\theta}'(k+1) = \boldsymbol{\theta}'(k) - \alpha(k)\nabla J(\boldsymbol{\theta}'(k)), \quad (4)$$

where $\alpha_s(k)$ is a positive scale factor that sets the step size, and $\nabla J(\boldsymbol{\theta}'(k))$ is the gradient vector, derived from a criterion function $J(\boldsymbol{\theta}'(k))$.

3.1. Criterion function for the gradient descent algorithm

The criterion function of the descent algorithm, we propose, is based on the difference between the current estimate and the moving average estimate of the model parameters. In general, this criterion function can be written as

$$J(\boldsymbol{\theta}_s(k)) = f(\boldsymbol{\theta}_s(k)) - f(\widehat{\boldsymbol{\theta}}_s(k)), \quad (5)$$

where $\boldsymbol{\theta}_s(k)$ denotes the current estimate and $\widehat{\boldsymbol{\theta}}_s(k)$ denotes the current moving average estimate of the model parameter vector. Both estimations of the parameter vectors correspond to the component density c_s, which is selected randomly, according to the a posteriori probabilities in equation (3). The moving average estimate $\widehat{\boldsymbol{\theta}}_s(k)$ is derived from a set of $N_s(k)$ recent feature vectors $\mathcal{X}_s = \{\mathbf{x}'_1, \cdots, \mathbf{x}'_{N_s(k)}\}$, which are considered as to be occuring from the component density c_s.

The function f in equation (5) should be defined in a such way that the gradient vector of the criterion function equals

$$\nabla J(\boldsymbol{\theta}_s(k)) = \boldsymbol{\theta}_s(k) - \widehat{\boldsymbol{\theta}}_s(k). \quad (6)$$

Then the gradient descent algorithm can be writen as

$$\boldsymbol{\theta}_s(k+1) = \boldsymbol{\theta}_s(k) - \alpha_s(k)\left(\boldsymbol{\theta}_s(k) - \widehat{\boldsymbol{\theta}}_s(k)\right). \quad (7)$$

3.2. Gaussian mixture probability density function

In a Gaussian mixture *pdf*, we deal with a model parameter vector $\boldsymbol{\theta}$, consisting of L a priori probabilities $p(c_i)$, mean vectors $\boldsymbol{\mu}_i$ and covariance matrices Σ_i. The conditional *pdf* $p(\mathbf{x}_k|c_i, \boldsymbol{\theta}_i)$ from equation (2) equals the Gaussian *pdf* $g(\mathbf{x}_k, \boldsymbol{\mu}_i, \Sigma_i)$. Consequently, the a posteriory probability, that the vector $\mathbf{x}_k$ occurs from the component density c_i is given as

$$p(c_i|\mathbf{x}_k) = \frac{p(c_i)g(\mathbf{x}_k, \boldsymbol{\mu}_i, \Sigma_i)}{\sum_{i=1}^{N} p(c_i)g(\mathbf{x}_k, \boldsymbol{\mu}_i, \Sigma_i)}. \quad (8)$$

3.2.1. *Criterion function for the Gaussian mixture probability density function*

Reffering to section 3.1, it can be easily shown, that the proper choices for criterion functions for parameter vectors $\boldsymbol{\theta}_s(k) = \{\boldsymbol{\mu}_s(k), \Sigma_s(k)\}$ are

$$J(\boldsymbol{\mu}_s(k)) = \frac{\boldsymbol{\mu}_s^T(k)\boldsymbol{\mu}_s(k)}{2} - \boldsymbol{\mu}_s^T(k)\widehat{\boldsymbol{\mu}}_s(k), \quad (9)$$

$$J(\Sigma_s(k)) = trace\left(\frac{\Sigma_s^2(k)}{2}\right) - trace\left(\Sigma_s(k)\widehat{\Sigma}_s(k)\right). \quad (10)$$

Consequently, the gradient descent algorithm can be written as

$$\boldsymbol{\mu}_s(k+1) = \boldsymbol{\mu}_s(k) - \alpha_s(k)\left(\boldsymbol{\mu}_s(k) - \widehat{\boldsymbol{\mu}}_s(k)\right), \quad (11)$$

$$\Sigma_s(k+1) = \Sigma_s(k) - \alpha_s(k)\left(\Sigma_s(k) - \widehat{\Sigma}_s(k)\right). \quad (12)$$

As we mentioned before, the parameter vector $\widehat{\boldsymbol{\theta}}_s(k) = \{\widehat{\boldsymbol{\mu}}_s(k), \widehat{\Sigma}_s(k)\}$ is given as the moving average estimate, derived from a set of recent feature vectors $\mathcal{X}_s = \{\mathbf{x}'_1, \cdots, \mathbf{x}'_{N_s(k)}\}$, which are considered to occur from the component density c_s. We decide randomly, which feature vector $\mathbf{x}_j \in \mathcal{X}$ occurs from which component density c_i, according to the a posteriori probability in equation (3). Since the sets of vectors that correspond to individual component densities change constantly while the current parameters estimates are updated, we should limit $N_s(k)$ to just a few recent or to only the last observed feature vector $\mathbf{x}(k)$. Considering the features of Gaussian *pdf*, the moving average estimates of both parameter vectors are

$$\widehat{\boldsymbol{\mu}}_s(k) = \beta_s(k)\widehat{\boldsymbol{\mu}}(k-1) + \gamma_s(k)\mathbf{x}(k), \quad (13)$$

$$\widehat{\Sigma}_s(k) = \beta_s(k)\widehat{\Sigma}_s(k-1) + \gamma_s(k)(\mathbf{x}_j - \boldsymbol{\mu}_k)^T(\mathbf{x}_j - \boldsymbol{\mu}_k), \quad (14)$$

where $\beta_s(k)$ denotes $N_s(k)/(N_s(k)+1)$ and $\gamma_s(k)$ denotes $1/(N_s(k)+1)$.

The a priori parobabilities $P(c_i)$ can be estimated conventionally from the calculated a posteriori probabilies as

$$p(c_i) = \frac{1}{M}\sum_{j=1}^{M} p(c_i|\mathbf{x}_j), \quad (15)$$

where M denotes the number of the feature vectors $\mathbf{x}_j \in \mathcal{X}$, taken into account. We can also introduce the moving average into the above equation.

3.2.2. *Initial estimations of parameters*

The presented gradient descent algorithm proved to be very robust due to the initial parameters estimation. We can use arbitrary initial estimations, although it is preferable to use a conventional single component Gaussian *pdf* parameters estimation procedure using the whole set of observable vectors. Then we can obtain initial estimates for all parameters, using a random splitting factor, as it is shown in the following section.

3.3. Iterative descent algorithm

The complete gradient descent algorithm, we used and evaluated, is given as it follows:

Step 1: Initialisation. Let us assume that the Gaussian mixture *pdf* consists of L components and the parameter vector $\boldsymbol{\theta}_0 = \{\boldsymbol{\mu}_0, \Sigma_0\}$ consists of the parameters of the Gaussian *pdf*, estimated conventionally, using the entire set of feature vectors $\{\mathbf{x}_j \in \mathcal{X}\}$. Set the initial estimates of all model parameter vectors to

$$\widehat{\boldsymbol{\theta}}_i(1) = \boldsymbol{\theta}_i(0) = \epsilon_i\boldsymbol{\theta}_0 \quad (16)$$

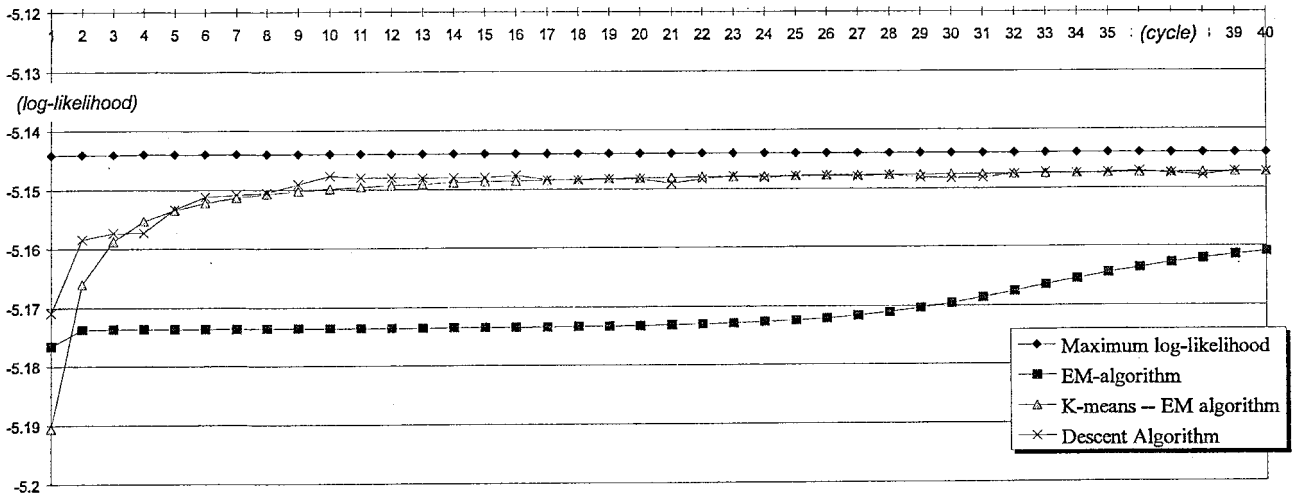

Figure 1: *Comparison of the convergence between the presented descent algorithm and EM reestimation algorithm.*

for $i = 1, \cdots, L$, and set a counter k=1. ϵ_i is a splitting diagonal matrix with random values close to 1.0. When setting the initial values, we have to satisfy all of Gaussian *pdf* covariance matrix properties. The initial estimates of the a priori probabilities can be set to $p(c_i) = 1/L$.

Step 2: Probabilities computation. Compute the a posteriori probabilities $p(c_i|\mathbf{x}(k))$ (8) that the current feature vector $\mathbf{x}(k)$ occurs from the component density c_i for every $i = 1, \cdots, L$.

Step 3: Decision. According to a set of computed a posteriori probabilities $p(c_i|\mathbf{x}(k))$ decide randomly, using a pseudo random generator, from which component density c_i the feature vector $\mathbf{x}(k)$ occurs. Assume now that the pseudo random generator selects the component density c_s.

Step 4: Parameters updating. Update the component density c_s model parameters as it is given in equations (11,12). Parameter $N_s(k)$ corresponds to the expected number of feature vectors that occur from the individual component density. The a priori probability $p(c_i)$ (15) is updated when the counter k reaches M. Parameter M can be set to the number of all observable feture vectors in $\mathcal{X}$ or to some arbirtary value.

Step 5: Stopping rule. If k reaches k_{max}, stop; otherwise proceed to Step 2 for the next feature vector, and set $k = k + 1$.

The above algorithm allows adjustments in the scale factor ($\alpha_s(k)$), the numbers of considered feature vectors ($N_s(k)$ and M), and maximum number of steps (k_{max}). In our experiments, it has been shown, that a fairly good choice for $\alpha_s(k)$ should equal the constant value within the range $[0.05 - 0.8]$, $N_s(k)$ should equal the constant value within the range $[10 - 1000]$, and M should equal N or some value lower than N if we deal with a huge set of observable feature vectors. We can also use a monotonically decreasing scale factor $\alpha_s(k)$. The maximum number of steps k_{max} should be larger than N.

4. ONE AND TWO-VARIATE GAUSSIAN MIXTURE PDF MODELLING EXAMPLE

We evaluated the presented algorithm on one and two-variate Gaussian mixture *pdfs*, generated using Gaussian pseudo random number generators. Figure 2 shows graphically the one-variate experimental results of the presented algorithm. In particular, it gives an example, where we assume, that we know the number ($L = 5$) of "hidden" component densities. The initial estimations were set as it is descibed in the previous section. Experimental results show that the estimations of the component densities are fairly close to the "hidden" component densities.

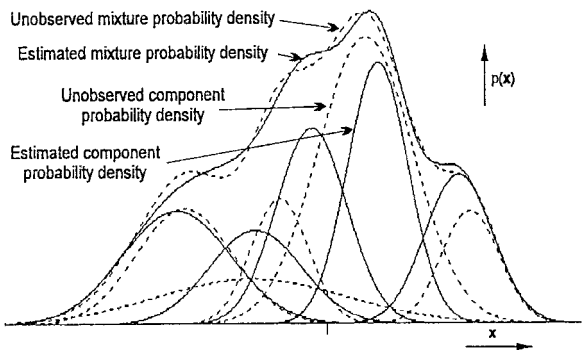

Figure 2: *A one-variate example of estimating the five "hidden" components densities using the presented descent algorithm.*

4.1. Convergence tests

The presented descent algorithm, which critically depends on taking small steps in the direction of the gradient, often does not produce any monotonic improvement as it is guaranteed by some reestimation algorithms. Thus, we compared the convergence of the presented algorithm to the one of the well known EM reestimation algorithm, using the logarithm of the total likelihood (1) as the criterion function. When using the EM reestimation algorithm, convergence occurs, when the gradient is zero. However, similarly to the presented descent algorithm, the EM algorithm does not guarantee to yield a global maximum.

We performed several convergence tests using two-variate pseudo Gaussian mixture *pdf* with five relatively overlapped component densities. Figure 1 shows the logarithm of total likelihoods after each cycle through the set of two-variate pseudo random feature vectors. It can be seen that the convergence of the presented algorithm is much faster than the convergence of the EM algorithm, especially if the EM algorithm is initialised in the same manner as the presented algorithm. The main disadvantage of the presented algorithm remains in its non-monotonic improvement.

5. PHONES RECOGNITION EXAMPLE

For additional experiments, we used labelled continuous speech signal frames cepstral feature vectors. The continuous speech signal database incorporates 50 training and 30 testing sentences, spoken by a male speaker. This represents approximately 5 minutes of speech and about 42000 feature vectors. Speech signal is labelled by 31 phone classes. This relatively small speech signal database is described more precisely in [2, 7].

We built three VQ codebooks. The first one was obtained using a conventional K-means clustering and an additional Learning Vector Quantisation algorithm [6, 8]. Here we used 9 centroid vectors per phone class. The second one was obtained using a conventional K-means clustering [8] for the initialisation step and the mentioned EM reestimation algorithm. In this case, we used only 5 component densities per phone class.

	LVQ	EM	GD
Vowels	76.1%	78.2%	79.8%
Nasals	83.3%	83.9%	84.9%
Sonorants	60.5%	62.2%	65.6%
Nonsonorants	61.5%	70.1%	74.3%
Total	70.9%	75.4%	77.3%

Table 1: *Classification results for all tested VQ modelling methods.*

The last VQ codebook wa obtained using the presented algorithm under equal conditions as in the previous case. In all three cases, the maximum number of iterations was selected to be 500000. In first two cases, there were additional approximately 100000 steps for the K-means clustering initialisation algorithm. The results that the algorithms achieved are given in Table 1. It can be seen that the presented descent algorithm yields fairly good results. When using less iterations, the descent algorithm achieves even better results then the other two. This proves the already mentioned rapid convergence of this algorithm. The results of the LVQ algorithm exibit a major disadvantage of the algorithm, which is in its concept, based on the Euclidian distance distortion measure.

6. DISCUSSION

The paper presents an alternative approach to mixture *pdf* modeling of a VQ codebook, using a rather specific gradient descent algorithm. First experimental results of Gaussian mixture *pdf* modelling, derived from sets of normally distributed one and two-dimensional pseudo random vectors, are given. Additional experiments on continuous speech signal frames feature vectors give comparable results to the well known Learning Vector Quantisation algorithm and the EM reestimation algoritm. The presented algorithm proved to be very robust due to the initial estimations of the model parameter vector. Further, it converges fast in comparison to the EM reestimation algorithm. We intend to perform some additional experiments on a much larger continuous speech signal database and to incorporate the presented algorithm into the Hidden Markov Model (HMM) concept [5].

REFERENCES

[1] A.P. Dempster, N.M. Laird, D.B.Rubin: *Maximisation Likelihood from Incomplete data via the EM algorithm.* Proc. R. Stat. Soc. B, pp. 1-38, 1977

[2] S. Dobrišek: *Introducing Context Dependency of Patterns in Phoneme Recognition of Slovenian Speech, M.Sc Thesis (In Slovene).* Faculty of Electrical and Computer Engineering, University of Ljubljana, Ljubljana, 1994.

[3] R.O. Duda, P.E. Hart: *Pattern Classification and Scene Analysis.* A Wiley-Interscience Publication, New York, London, Sydney, Toronto, 1973.

[4] R.M. Gray: *Vector Quantisation.* IEEE ASSP Magazine, Vol. 1(2), pp. 4-29, 1984.

[5] X.D. Huang, Y. Ariki, M.A. Jack: *Hidden Markov models for speech recognition.* Edinburg University Press, Edinburg, 1990.

[6] T. Kohonen: *Some Practical Aspect of the Self Organizing Maps.* IJCMTI, Vol 2., pp. 2.253-2.256, Washington DC, 1990.

[7] F. Mihelič, I. Ipšić, S. Dobrišek, N. Pavešić: *Feature Representations and Classification Procedures for Slovene Phone Recognition.* Pattern Recognition Letters, no.12, vol.13, pp. 879-891, Elsevier North-Holland, 1992.

[8] L. Rabiner, B.H. Juang: *Fundamentals of Speech Recognition.* PTR Prentice Hall, New Jersey, 1993

[9] Y. Zhang, M. Alder, R. Togneri: *Using Gaussian Mixture Modeling in Speech Recognition.* ICASSP'94 Proceedings, vol.I, pp. 613-616, Aldelaide, South Australia, 1994.