A COMPARISON OF THE LBG, LVQ, MLP, SOM AND GMM ALGORITHMS FOR VECTOR QUANTISATION AND CLUSTERING ANALYSIS

R. Togneri, D. Farrokhi, Y. Zhang and Y. Attikiouzel

Department of Electrical and Electronic Engineering
The University of Western Australia

ABSTRACT - We compare the performance of five algorithms for vector quantisation and clustering analysis: the Self-Organising Map (SOM) and Learning Vector Quantization (LVQ) algorithms of Kohonen, the Linde-Buzo-Gray (LBG) algorithm, the MultiLayer Perceptron (MLP) and the GMM/EM algorithm for Gaussian Mixture Models (GMM). We propose that the GMM/EM provides a better representation of the speech space and demonstrate this by comparing the GMM with the LBG, LVQ, MLP and SOM algorithms in phoneme classification and digit recognition.

INTRODUCTION

Currently, the most popular approach to speech recognition is the combination of Vector Quantization (VQ) for the encoding of segments of speech with a Hidden Markov Model (HMM) for the classification of sequences of segments. The VQ stage is usually unsupervised since no a priori assumptions on the speech class distribution is made.

The VQ algorithm proposed by Linde *et al.* (1980) , and subsequently referred to as the LBG algorithm is currently the algorithm preferred by most workers in the area of speech recognition.

The process of speech production suggests that different utterances of the same speech sound will form a cluster around some centre. The variations about the mean will occur at random when a large population of speakers is considered, so the points in the cluster will be distributed according to a multidimensional Gaussian probability density function. This suggests that vector quantization of the speech space is better done on the basis of a Gaussian Mixture Model (GMM), in which the points are clustered around the means according to Gaussian distributions, and each cluster is assigned a weight representing the frequency with which points in the cluster occur.

An algorithm for estimating means, covariances and weights for GMMs was described by Wolfe (1970). This algorithm was later described as an Estimation-Maximization (GMM/EM) algorithm by Dempster *et al.* (1977). The GMM/EM algorithm for GMMs is a means of quantizing the speech space in a way that closely reflects the speech production process.

In earlier papers (Zhang *et al.*, 1991, 1992), we have described a comparison of the performance of speaker-independent digit recognition systems using the LBG and GMM/EM algorithms respectively for VQ. The system using the GMM/EM algorithm outperformed the system using the LBG algorithm.

An alternative approach to vector quantization, based on neural network ideas, has been proposed by Kohonen (1990). The Self-Organizing Map (SOM) is a nearest-neighbour classification scheme that uses an unsupervised learning algorithm to determine the positions of the codebook vectors. The SOM does not produce approximations to the density functions of the class samples. Instead, it defines the class borders according to the nearest-neighbour rule. The borders are piecewise linear, being composed of segments of mid-planes between codebook vectors of neighbouring classes.

Another vector quantization approach proposed by Kohonen is the Learning Vector Quantization (LVQ) algorithm. Like the SOM this uses a nearest-neighbour classification scheme but uses a supervised learning algorithm for determining the codebook positions. A collection of programs implementing various LVQ algorithms has been published by Kohonen and his co-workers (Kohonen *et al.*, 1992). In an earlier paper (Togneri *et al.*, 1992), we compared the performance of the GMM/EM algorithm with the LVQ for both phoneme classification and digit recognition. In the latter case the LVQ could not be trained for codebooks smaller than 2048. In this paper we use the SOM for digit recognition as it is more suitable for the vector quantisation of speech for this task.

The classical neural network architecture for supervised learning is the MultiLayer Preceptron (MLP) trained using the BackPropagation algorithm (Rumelhart and McClelland, 1986).

In this paper we compare the performance of the LBG, LVQ, MLP and SOM algorithms with that of the GMM/EM algorithm. We make two comparisons: first, in the classification of phonemes using GMM/EM, LVQ and MLP; and second, in a digit recognition task using GMM/EM, SOM and LBG. In both cases, the GMM/EM gives the best performance, demonstrating that it provides a superior description of the speech space.

METHOD - PHONEME CLASSIFICATION

Phoneme data was obtained from the TIMIT data base produced by the National Institute of Standards and Technology (U.S.A.). Examples of 56 phonemes were extracted from sentences spoken by three male speakers. The sentences had been recorded in low noise conditions, sampled at 16 kHz and digitized to 16 bit resolution.

The speech data was windowed and 512-point FFT's were computed for each window. The first 256 FFT coefficients were binned into 12 mel-spaced values, to produce 12-dimensional feature vectors. These vectors were assigned to phonemic classes on the basis of the labelling provided by the TIMIT data base.

Four variants of the LVQ algorithm were used in the comparison. *olvq1*, the optimized version of *lvq1*, was used to construct the initial codebook. After this the other versions, *lvq1, lvq2* and *lvq3* were used. They operate in slightly different ways and require different parameters, which will be described in the next section.

METHOD - DIGIT RECOGNITION

Spoken English digits were obtained from the TIDIGITS data base produced by the National Institute of Standards and Technology (U.S.A.). The training set consisted of 112 speakers saying the words 'zero', 'one', 'two', 'three', 'four', 'five', 'six', 'seven', 'eight', 'nine', and 'oh'. The testing set consisted of 113 other speakers saying the same words. The utterances were recorded in a quiet studio, sampled at 20 kHz and digitized to 16 bit resolution.

The speech data was windowed and 512-point FFT's were computed for each window. The window was moved 200 points between frames. The first 256 FFT coefficients were binned into 12 mel-spaced values to produce 12-dimensional feature vectors.

The set of feature vectors was then used to construct VQ codebooks using each of the GMM/EM, SOM and LBG algorithms. The quantized feature vectors were used to construct 5-state discrete Hidden Markov Models for each of the 11 utterances. These were then used for the recognition task.

RESULTS - PHONEME CLASSIFICATION

An LVQ codebook of 800 vectors was used in all the comparisons. Versions of the algorithm require a learning rate control parameter which was always set to 0.05. *lvq3* requires a relative learning rate parameter, which was set to 0.1. *lvq2* and *lvq3* require a window width parameter, which was set to 0.3. The other control parameter is the number of training steps, which was varied as described in the table of results.

The following table summarises the results obtained from a number of trials. It shows the version of the algorithm that was used, the number of training steps, and the percentage of phonemes that were correctly classified.

Version	Number of training steps	Percentage correct
olvq1	32000	28.43 %
lvq1	160000	39.54 %
lvq2	160000	45.34 %
lvq2	320000	48.53 %
lvq2	640000	49.55 %
lvq2	1280000	51.62 %
lvq3	160000	44.65 %
lvq3	640000	45.08 %

Table 1. Phoneme classification results using the LVQ algorithms.

The best results were obtained using *lvq2* and 128000 training steps.

A three-layer (input, one hidden, output) MLP was used with 12 input nodes, corresponding to the 12 mel-spaced input values, 35-55 nodes in the hidden layer and 56 output nodes, corresponding to the 56 different phoneme classes. To avoid saturating the network all input values were normalised by 10. A gain of 0.1 and a momentum factor of 0.01 was used for all trials. The number of iterations and number of nodes in the hidden layer were varied. The results are shown in Table 2.

Hidden Nodes	Iterations	Percentage correct
35	500000	34.57 %
35	700000	35.24 %
35	900000	36.13 %
45	500000	35.98 %
55	500000	35.47 %

Table 2. Phoneme classification results using the MLP.

When constructing codebooks based on Gaussian mixture models using the GMM/EM algorithm, the number of Gaussians per class and the number of iterations need to be specified. In addition, the means of the Gaussians can either be initialized to random positions or they can be placed systematically in positions near clusters of data.

The following table summarises the results from four trials. It shows the number of Gaussians used for each of the 56 classes, the number of iterations, whether initialization of the means was random or systematic, and the percentage of phonemes that were correctly classified.

Gaussians per class	Number of iterations	Initialization	Percentage correct
1	5	Random	40.10 %
1	5	Systematic	40.10 %
2	10	Random	39.46 %
2	10	Systematic	44.81 %

Table 3. Phoneme classification results using the GMM/EM algorithm.

The best results were obtained using 2 Gaussians per class and systematic initialization. Systematic initialization is clearly important when there are more Gaussians to train per class. Both the LVQ, MLP and GMM/EM results are comparable and equally poor with worse results than random classification of the data. These results further highlight the difficulty of performing general phoneme recognition except for some restricted cases.

RESULTS - DIGIT RECOGNITION

The SOM network was arranged as a two-dimensional grid of neural units (each neural unit is a codebook vector). Grid sizes of 4x8, 8x8, 8x16 and 16x16 were trained and tested corresponding to codebook sizes of 32, 64, 128 and 256 respectively. The training to produce the codebook vectors was performed in two stages. The ordering stage comprised 500000 iterations using a linearly decreasing adaptation gain of 0.5 and a neighbourhood size which linearly decreased from half the largest grid length to a size of 1. The convergence phase comprised 2000000 iterations using a linearly decreasing adaptation gain of 0.05 and a fixed neighbourhood size of 1.

By quantizing the speech data using the SOM codebook vectors digit recognition was performed using the HMM. Table 4 gives the average percentage correct recognition rate for each of the different codebook sizes. The results of the SOM/HMM combination were obtained on a set of digits spoken by speakers other than those used for training.

Codebook size	32	64	128	256
SOM	82.0%	87.8%	92.8%	94.6%

Table 4. Digit recognition results using the SOM algorithm

For the GMM/EM algorithm, codebooks comprising 32, 64, 128 and 256 Gaussians respectively were produced, using both random and systematic initialization of the means of the Gaussians. Table 5 summarises the results of testing the GMM/EM and HMM combination on the set of digits. The same training and test data was used for all digit recognition results.

Codebook size	32	64	128	256
GMM/EM with random initialization	96.5%	97.4%	96.1%	97.4%
GMM/EM with systematic initialization	95.2%	96.7%	97.3%	97.7%

Table 5. Digit recognition results using the GMM/EM algorithm.

Finally the results using the LBG algorithm are shown using codebook sizes of 32, 64, 128 and 256. Table 6 summarises the digit recognition results using the LBG/HMM combination.

Codebook size	32	64	128	256
LBG	92.8%	93.8%	94.3%	95.1%

Table 6. Digit recognition results using the LBG algorithm

The best results were obtained with the codebook comprising 256 Gaussians with systematic initialization of the means. In all but one case, the GMM/EM and HMM combination achieved greater recognition than both the LVQ/HMM combinations. It should be noted, however, that the number of codebook vectors used for LVQ was eight times the number of codebook Gaussians used for GMM/EM.

CONCLUSION

We have argued that the GMM provides a better description of the speech space than either the LBG, LVQ, MLP or SOM procedures. This has been borne out by the comparable performance of the former in phoneme classification and superior performance in isolated digit recognition.

REFERENCES

Dempster, A. P., Laird, N. M., and Rubin, D. B. (1977) 'Maximum Likelihood from Incomplete Data via the EM Algorithm', *Proc. R. Stat. Soc. B,* 39(1), 1–38.

Kohonen, T. (1990) 'The self-organizing map', *Proceedings of the IEEE,* **78**(9), 1464–1480.

Kohonen, T., Kangas, J., Laaksonen, J. and Torkkola, K. (1992) 'LVQ_PAK: The Learning Vector Quantization Program Package', on-line documentation prepared by the LVQ Programming Team of the Laboratory of Computer and Information Science, Helsinki University of Technology, Espoo, Finland.

Linde, Y., Buzo, A., and Gray, R. M. (1980) 'An Algorithm for Vector Quantizer Design', *IEEE Trans. on Communications,* COM-28(1), 84–95.

Rumelhart D.E. and McClelland J.L. (eds.), 'Parallel Distributed Processing, Explorations in the Microstructure of Cognition, Volume 1: Foundations', MIT Press, Cambridge, Mass., 1986.

Togneri R., Zhang Y., deSilva C.J.S. and Attikiouzel Y. (1992) 'A Comparison of the LVQ and EM algorithms for Vector Quantization in a Speaker-Independent Digit Recognition Task', Proceedings of The Third International Symposium on Signal Processing and its Applications, Gold Coast, Australia, 2, 384–387.

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

Zhang, Y. and deSilva, C. J. S. (1991) 'An Isolated Word Recognizer Using the EM Algorithm for Vector Quantization', IREECON 1991, (Sydney, Australia), 289–292.

Zhang, Y., deSilva, C. J. S., Attikiouzel, Y. and Alder, M. D. (1992) 'A HMM/EM Speaker-Independent Isolated Word Recognizer', accepted for publication by *The Journal of Electrical and Electronic Engineers, Australia,*.

TWO NOVEL SPEECH CODING TECHNIQUES BASED ON MULTIPULSE REPRESENTATION

O. A. Alim, E. A. Youssef and A. G. Mokhtar

Department of Electrical Engineering
Faculty of Engineering
University of Alexandria

ABSTRACT - Two new coding techniques based on multipulse representation of the excitation signal in the time and frequency domains are presented. Algorithms, illustrations, modelling processes as well as bit rates are discussed. Real speech, English as well as arabic mono and di syllabic words were used to test and evaluate the coding techniques subjectively and objectively. Both coding techniques produced good quality speech but the time domain method which operates at bit rates in the range of 8-16 kbps, was found to be superior to the frequency domain method which operates at a bit rate of 5740 bit/s.

INTRODUCTION

One of the techniques which is most promising in the area of speech coding is Linear Predictive Coding (LPC) . This technique when applied, produced synthetic speech quality at a bit rate of 2.4 kbits/s [2]. Slight quality improvements were possible as bit rates were increased above 2.4 kbits/s, but the degradations inherent in the simplistic LPC model remain. The major impediment to toll or communications quality speech lies in the single pulse periodic excitation, which adds a mechanical aspect to the synthetic speech. As a result, it was clear that the key for an improved LPC system performance was excitation improvement. A breakthrough was made by B. S. Atal and J. R. Remede in 1982 [1] when they introduced the multipulse excitation in which they used more than one pulse per pitch period and adjusting the individual pulse positions and amplitudes to minimize a certain error criteria. In 1989 J. V. Schalkwyk and J.V. Der Linde [4] used a frequency domain multipulse coding technique at 2.4 kbits/s promising to have much more natural sounding speech than the conventional LPC system.

Inspired by the work done in the excitation improvement area, two new coding techniques based on multipulse representation of the excitation signal are presented in this paper. The first technique is based on the multipulse excitation in the time domain. The second technique is based on the multipulse excitation in the frequency domain.

TIME DOMAIN METHOD

The algorithm presented here uses the LPC error sequence generated by passing the original speech through the LPC inverse filter, namely

$$e(n) = S(n) - \sum_{k=1}^{p} a_k S(n-k) \tag{1}$$

where a_k's are the coefficients resulting from LPC analysis and p is the order of the LPC filter. The objective is to obtain another excitation sequence which contains a smaller number of pulses, so when passed through the synthesis filter, the produced speech resembles the original one. Equ (1) when rewritten as

$$S(n) = e(n) + \sum_{k=1}^{p} a_k S(n-k) \tag{2}$$

shows that higher values of e(n) affect the output speech more than lower ones. Also the energy of the error sequence is given by Equ. (3), with w(n-m) being the window used.