Speaker-independent isolated word recognition using multiple hidden Markov models

Y. Zhang
C.J.S. deSilva
R. Togneri
M. Alder
Y. Attikiouzel

Indexing terms: Word recognition, Multiple hidden Markov models, Subrecognisers

Abstract: A multi-HMM speaker-independent isolated word recognition system is described. In this system, three vector quantisation methods, the LBG algorithm, the EM algorithm, and a new MGC algorithm, are used for the classification of the speech space. These quantisations of the speech space are then used to produce three HMMs for each word in the vocabulary. In the recognition step, the Viterbi algorithm is used in the three subrecognisers. The log probabilities of the observation sequences matching the models are multiplied by the weights determined by the recognition accuracies of individual subrecognisers and summed to give the log probability that the utterance is of a particular word in the vocabulary. This multi-HMM system results in a reduction of about 50% in the error rate in comparison with the single model system.

1 Introduction

Currently, one of the most popular approaches to speech recognition is the combination of vector quantisation (VQ) for the encoding of segments of speech with a hidden Markov modelling (HMM) for the classification of sequences of segments [5]. We can consider VQ/HMM to be a two-step modelling technique. The first step, vector quantisation, is used to divide the signal space into a number of cells or subspaces to produce a codebook of vectors. Each vector in the codebook corresponds to a cell and is used to represent all the vectors in that cell. The second step, hidden Markov modelling, is used to produce a set of models which represent possible sequences of codebook vectors which arise from words that the system is to recognise.

The commonest VQ algorithm is the LBG algorithm, which was named from the initials of the three authors [4]. It is an example of K-means clustering and has the advantages of being simple and not requiring excessive computation. The LBG algorithm does not guarantee that the classification of the speech space is globally optimal. This means that some of the codebook vectors may not be typical of the vectors in the cells they represent. The shortcomings of the LBG algorithm lead to an inappropriate classification of the speech space and inadequate matching with the hidden Markov modelling, and consequently a limited recognition accuracy for the whole system.

Observations have shown that different utterances of the same speech sound form a cluster around some centre, which represents some average or fiducial production of the sound. The variations about the mean will occur at random when a large population of speakers is considered, so the points in the cluster may be distributed according to a multidimensional Gaussian probability density function. This view of the speech production process suggests that classification 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. A method now known as the expectation maximisation (EM) algorithm for estimating the parameters of GMMs was described by Wolfe [7]. The EM algorithm is an iterative algorithm for the derivation of maximum likelihood estimates of the parameters of a wide variety of statistical models and can be used as a substitute for the LBG algorithm for quantisation of the speech space. The EM algorithm for GMMs is a means of quantising the speech space in a way that reflects the speech production process more closely than the LBG algorithm. Experiments have shown that the EM algorithm matches the HMM quite well and leads to a better recognition accuracy [6, 9]. However, the EM algorithm is more computation intensive than the LBG algorithm and is sensitive to background noise [8, 9]. Another classification method that we have devised is the multiple Gaussian clustering (MGC) algorithm which is similar to the EM algorithm, but requires less computation in training and produces slightly rougher classification than the EM algorithm.

Further observations have shown that the three different VQ algorithms classify the speech space into different cells or subspaces, resulting in different recognition errors.

Paper 1142K (E5), first received 15th November 1993 and in revised form 2nd February 1994

The authors are with the Centre for Intelligent Information Processing Systems, University of Western Australia, Nedlands, WA 6009, Australia

This work was supported in part by the University Fee-Waiver Scholarship and the University Research Studentship of the University of Western Australia.

Figs. 1 and 2 show how the three different classification methods perform on two-dimensional data. In each Figure, the top left panel depicts a collection of

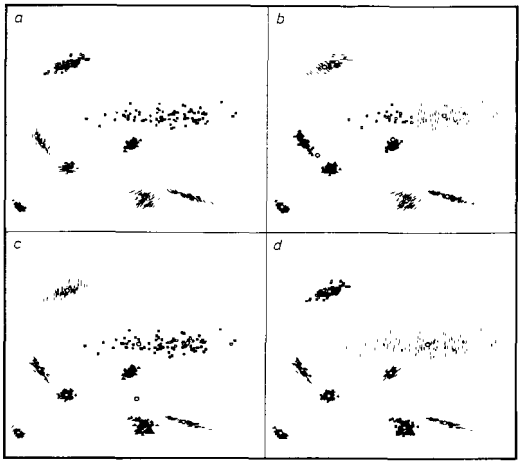

Fig. 1 *Gaussian random data A and classification results*
a Original data
b LBG quantised result
c MGC quantised result
d EM quantised result

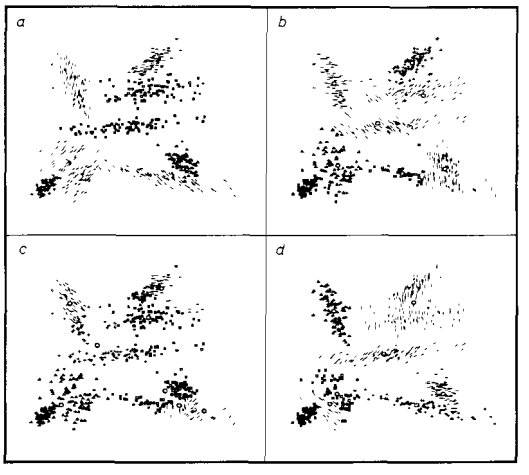

Fig. 2 *Gaussian random data B and classification results*
a Original data
b LBG quantised result
c MGC quantised result
d EM quantised result

points in eight Gaussian clusters. In the top right, bottom left, and bottom right panels are the results of classification using the LBG, MGC, and EM algorithms, respectively. The points in each class have been given the same sign. The difference between the two Figures is that the eight clusters in Fig. 1 are separate but they are overlapped in Fig. 2.

It is clear from these Figures that the three classification methods partition the data space in different ways, and that the clusters arising from the EM algorithm best resemble the original clusters.

Using the three algorithms, three isolated word recognisers, denoted LBG/HMM, MGC/HMM, and EM/HMM, were constructed. The recognisers were tested on a set of 1120 isolated digits. The numbers of errors for the recognisers were 63, 52, and 31 respectively. Fig. 3 is a Venn diagram depicting the pattern of

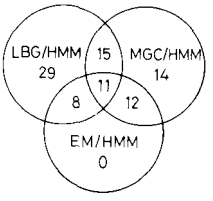

Fig. 3 *Recognition errors generated by various recognisers*

common errors. The different patterns of errors suggest that combining information from all three might improve the recognition performance, since there are only 11 utterances that were erroneously identified by all three recognisers.

In this paper we describe a multi-HMM (MHMM) speaker-independent isolated word recogniser in which the three VQ algorithms mentioned above are used independently of each other. These quantisations of the speech space are then used to produce three HMMs for each word in the vocabulary using the Baum–Welch algorithm. In the recognition step, the Viterbi algorithm is used. The log probabilities of the observation sequences matching the models are multiplied by the weights determined by the individual recognition accuracies and summed to give the log probability that the utterance is of a particular word in the vocabulary. We report the results of comparing this method with the use of a single vector quantisation algorithm. This results in reduction of about 50% in the error rate compared to the best single VQ/HMM system.

2 Vector quantisation

The first practical vector quantiser was proposed by Linde *et al.* in 1980 [4]. The purpose of using VQ is to compress data to reduce the computations in signal processing, compress the signal frequency band in the data transmission, and reduce the use of storage medium in the data store. It was first used for speech and image coding. Rabiner successfully constructed a VQ/HMM system which used VQ combined with HMM for speech recognition in 1983 [3]. In a VQ/HMM system, the vector quantisation works as a signal space classifier and a data compressor. First, it classifies the signal space into a number of subspaces and generates a codebook of vectors which are the centres of, or typical vectors in, the subspace they represent. Secondly, the vector quantiser outputs an index of codebook vector instead of a whole vector for each input vector. This reduces the computation for the Viterbi algorithm.

Three classification algorithms have been used for vector quantisation in our VQ/MHMM isolated word recognition system. We describe them below.

2.1 The LBG algorithm

The LBG algorithm is also known as the clustering and splitting algorithm because each iteration consists of a clustering step and a splitting step. In the splitting step, each vector in the codebook generated in the previous iteration is split into two and a new codebook which has twice as many vectors as before is generated. In the clus-

tering step, the new codebook vectors are used to cluster the all input data points and form the new subspaces. The clustering is repreated until a predetermined threshold is reached. After R iterations, a codebook of size 2^R is generated.

The complete LBG algorithm may be described as follows:

(*a*) Initialisation: Fix N as the codebook size desired, ξ = threshold, set $M = 1$. Given a training sequence x_j, find the centroid vector as a one vector codebook.

(*b*) (splitting): Split each codebook vector y_i into $y_i + \varepsilon$ and $y_i - \varepsilon$, where ε is a fixed perturbation vector. Replace M by $2M$.

(*c*) (clustering): Classify the training sequence using the codebook vectors generated in (*a*) into M cells.

(*d*) Find the centroids of M cells. Using these centroids classify the training sequence and calculate the relative distortion,

$$D = (DIS_l - DIS_t)/DIS_t$$

where DIS_l is the total distortion of the previous iteration and DIS_t is the total distortion of current iteration. The total distortion is defined as

$$DIS = \sum_{i=1}^{s} Min_{j=1}^{M} \sqrt{\left[\sum_{k=1}^{d} (x_i^k - \mu_j^k)^2\right]}$$

where $Min_{j=1}^{M}$ means the minimisation over j from 1 to M, s is the number of vectors in the training sequence, M is the current code book size, x is the input vector, μ is the code book vector, and d is the vector dimension.

(*e*) If $D \leqslant \xi$, continue. Otherwise go to (*c*).

(*f*) If $M = N$, finish and output the final codebook. Otherwise go to (*b*).

For classifying vectors, the following procedure is used:

(*a*) Input a vector x and calculate its Euclidean distance D_i from each codebook vector y_i;

(*b*) Find the minimum distance D_{min} of D_0, D_1, ..., D_{N-1};

(*c*) Output the index of codebook vector which is closest to the input vector x.

2.2 The EM algorithm

The EM algorithm may be used for computing the parameters of a Gaussian mixture model (GMM). A GMM is defined by a probability density function of the form:

$$f(x) = \sum_{i=1}^{N} p_i g(x, \mu_i, \Sigma_i)$$

where $g(x, \mu_i, \Sigma_i)$ is the Gaussian probability density function with mean μ_i and covariance matrix $\Sigma_i = (\sigma_i^{jk})$, x is a random d-dimensional vector, $x = (x^1, x^2, \ldots, x^d)$, and the p_i are weights which describe the relative likelihood of classes being generated from each of the clusters and which must satisfy $\sum_{i=1}^{N} p_i = 1$, where N is the number of classes or size of the codebook.

The Gaussian mixture model is an effective description of data sets comprising clusters of vectors which are both isolated from each other and convex. In the case where the distance between means is large in comparison to the square roots of the variances, this model describes a set of isolated clusters of ellipsoidal shape. As the distances between means decreases, the isolation of the clusters is progressively reduced until they merge into one another.

The EM algorithm is a general statistical procedure in which each iteration consists of an expectation (E) step followed by a maximisation (M) step.

Suppose we have a sample of S points $x_j = (x_j^1, x_j^2, \ldots, x_j^d)$, $j = 1, 2, \ldots, S$, drawn from a set of points which are assumed to lie in N clusters. We initialise N Gaussians with probabilities $p_1 = p_2 = \cdots = p_N = 1/N$, means μ_1, $\mu_2, \ldots, \mu_N$, which can either be random or set equal to N of the data points with a small perturbation, and covariance matrices $\Sigma_1, \Sigma_2, \ldots, \Sigma_N$, set equal to the identity matrix or a multiple thereof.

In the E-step we compute:

the total likelihoods

$$t_j = \sum_{i=1}^{N} p_i g(x_j, \mu_i, \Sigma_i), \quad j = 1, 2, \ldots, S \tag{1}$$

where g is the Gaussian probability density function

the normalised likelihoods

$$n_{ij} = p_i g(x_j, \mu_i, \Sigma_i)/t_j, \quad i = 1, 2, \ldots, N; j = 1, 2, \ldots, S \tag{2}$$

the notional counts

$$C_i = \sum_{j=1}^{S} n_{ij}, \quad i = 1, 2, \ldots, N \tag{3}$$

the notional means

$$\bar{x}_i = \sum_{j=1}^{S} x_j n_{ij}/C_i, \quad i = 1, 2, \ldots, N \tag{4}$$

and the notional sums of squares

$$SS_i^{pq} = \sum_{j=1}^{s} x_j^p x_j^q n_{ij}/C_i \quad i = 1, 2, \ldots, N \quad \text{and} \quad p, q = 1, 2, \ldots, d \tag{5}$$

In the M-step we compute new values of the parameters of the Gaussian model as follows:

$$p_i = C_i/S$$

$$\mu_i = \bar{x}_i$$

$$\Sigma_i^{pq} = SS_i^{pq} - \bar{x}_i^p \bar{x}_i^q$$

where $i = 1, 2, \ldots, N$.

Dempster *et al.* [1] have proved that the EM algorithm is convergent and that the convergence rate is quadratic. In most cases, according to our observations, ten iterations are sufficient to yield useful estimates of the parameters of the Gaussian mixture model.

For classifying vectors, the following procedure is used:

(*a*) Input a vector x and calculate the weighted likelihood of the input vector with respect to each Gaussian in the codebook. The weights of likelihood are p_i described above.

(*b*) Find the maximum weighted likelihood.

(*c*) Output the index of the Gaussian which gives the maximum likelihood.

2.3 The MGC algorithm

The multiple Gaussian clustering algorithm (MGC) was devised by the second author as a simple alternative to the EM algorithm for the construction of a GMM. It is described here for the first time.

As before, we have a sample of S points, $x_m = (x_m^1, x_m^2, \ldots, x_m^d)$, $m = 1, 2, \ldots, s$, drawn from a set of points which are assumed to lie in N clusters. We initialize N mean vectors $\mu_1, \mu_2, \ldots, \mu_N$ either to random positions in the speech space or to positions which are considered good estimates of the means of the clusters. We also initialise counts C_i, sums S_i^j, and products P_i^{jk}, to zero, where

$i = 1, 2, \ldots, N; j = 1, 2, \ldots, d; k = 1, 2, \ldots, d$. We then go through the data sample, one vector at a time, carrying out the following steps:

(*a*) For each mean $\mu_i = (\mu_i^1, \mu_i^2, \ldots, \mu_i^d)$, we compute the Euclidean distance between the data point and the mean,

$$\sqrt{\left[\sum_{j=1}^{d} (x_m^j - \mu_i^j)^2\right]} \tag{6}$$

(*b*) We find the closest mean μ_n to the data point x_m;

(*c*) We update the counts, sums and products for the cluster to which this mean belongs;

$$C_n' = C_n + 1$$

$$S_n'^j = S_n^j + x_m^j$$

$$P_n'^{jk} = P_n + x_m^j x_m^k$$

(*d*) We compute new values for the cluster means,

$$\mu_n^j = S_n'^j / C_n'$$

The new value for the mean is used in the first step for the next data point.

After all the points have been used, we calculate the remaining parameters of the Gaussian distributions as follows:

$$p_i = C_i / S$$

$$\Sigma_i^{jk} = P_i'^{jk} / C_i - \mu_i^j \mu_i^k \tag{7}$$

For classifying vectors, the following procedure is used:

(*a*) Input a vector x and calculate its Mahalanobis distance D_i from each Gaussian in the codebook;

(*b*) Find the minimum distance D_{min} of $D_0, D_1, \ldots, D_{M-1}$;

(*c*) Output the index of the Gaussian which is closest to the input vector x.

The MGC algorithm works reasonably well where the clusters are well separated and the initial values of the means are close to the clusters. It is fast in comparison to the LBG and EM algorithms. Considering the Mahalanobis distance is associated with the mean and covariance matrix of a Gaussian, we can say that the MGC algorithm clusters the speech data using a geometrical method concerned with the statistical features of the data.

3 System construction

The recognition accuracy of VQ/HMM speech recognisers is limited. One reason for this is that the classification of the speech space using vector quantisation is not perfect. As indicated above, the three different classification methods classify the speech space into different cells or subspaces, leading to different errors. In other words, the errors produced by the three different VQ methods do not always overlap.

We therefore construct a multiple hidden Markov model (MHMM) speaker-independent isolated word recogniser. Fig. 4 is a block diagram of a VQ/MHMM recogniser. The system is composed of three subrecognisers, each of which uses one vector quantisation method for the first step modeling.

In the training step, the three classification algorithms mentioned above are employed for a parallel vector quantisation and a codebook is generated for every subrecogniser. Then three models for each word of the vocabulary are produced by the hidden Markov modelling (Baum–Welch) algorithm. In the recognition step, the Viterbi algorithm is used in parallel with the three subrecognisers. The log probabilities of the observation sequences matching the models are multiplied by weights determined by the individual recognition accuracies. It is reasonable to assume that the three observation sequences from the vector quantisers are independent. Therefore the weighted log probabilities for each word to be recognised are summed. Then the model which produces the highest probability is the output.

Table 1 demonstrates the recognition improvement of the VQ/MHMM system. The numbers in the table are the output scales of individual VQ/HMM and VQ/MHMM recognisers when the input word is 'ONE'. The highest scale is the system output. The whole system

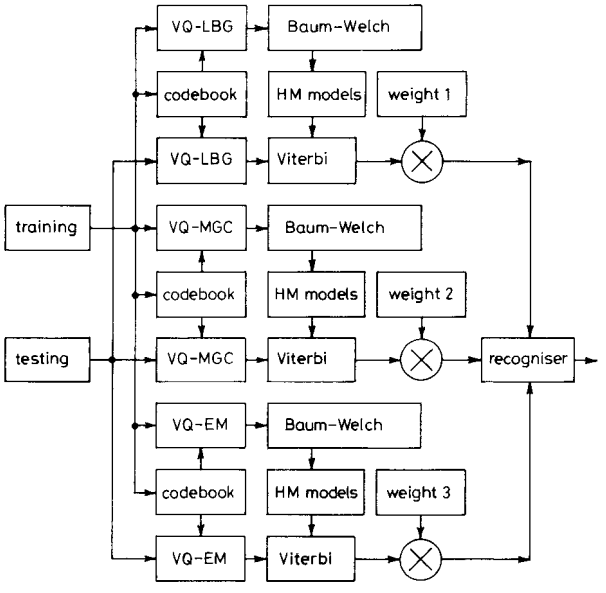

Fig. 4 *Block diagram of the multiple hidden Markov model speaker-independent isolated word recognition system*

Table 1: Scales of hidden Markov models matching the input word 'ONE' with three vector quantisation algorithms

	LBG/HMM	MGC/HMM	EM/HMM	VQ/MHMM
One	−134	−122	−111	−539
Two	−191	−188	−184	−841
Three	−173	−165	−169	−759
Four	−156	−148	−151	−680
Five	−166	−169	−163	−746
Six	−189	−197	−200	−885
Seven	−178	−202	−191	−803
Eight	−195	−186	−183	−840
Nine	−129	−131	−140	−606
Zero	−177	−173	−177	−791
Oh	−141	−119	−144	−608
Output	Nine	OH	ONE	ONE

gave a correct output even when two of three single systems, LBG and MGC, gave incorrect recognition.

4 Experiment and results

A set of evaluation tests was performed on the VQ/MHMM system. Our data base was the Studio Quality Speaker-Independent Connected-Digits Corpus (TIDIGITS) published by the National Institute of Standards and Technology in the USA [2]. The training data comprised a small vocabulary of eleven isolated digits (from zero to nine and oh) spoken by 112 speakers (55 male and 57 female). The testing data comprised the same digits spoken by 113 different speakers (56 male and 57 female). The data was recorded in studio conditions and digitised at 20000 samples per second.

For preparing the input for the VQ system, the speech data was windowed and feature vectors were constructed for each window. The first preprocessing step was the computation of the power spectrum of the windowed signal using a FFT routine, followed by summation of the components of the power spectrum to simulate a bank of 12 mel-spaced band-pass filters. The FFT analysis frame was about 25 ms and had a 15 ms advance. The 12 band-pass filters bank covered the range from 0 to 5000 Hz.

In our experiments, a constrained left-to-right HMM structure with five states, as described by Rabiner [3], was used. The model is shown in Fig. 5. The number

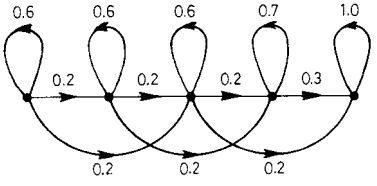

Fig. 5 *The initial left-to-right Markov model with 5 states*

associated with each edge is the transition probability used to initialise each model before training.

A set of preliminary investigations was performed using the individual VQ/HMM systems. Different codebook sizes were used and Table 2 shows the results.

Table 2 shows that among the individual recognisers the EM/HMM gave the best recognition accuracies, and the LBG/HMM the worst. This is consistent with our expectation that the Gaussian mixture model is a good description of speech features which matches hidden Markov modelling very well.

The codebook size is important for the overall system performance. A small codebook will yield a low recognition accuracy because of the high vector quantisation distortion while a large codebook leads a better system performance. On the other hand, the codebook size is

Table 2: Results of individual VQ/HMM recognisers

Codebook size		32	64	128	256	512
Error rate	LBG	13.85%	6.89%	5.67%	4.86%	4.13%
	MGC	12.22%	6.80%	4.62%	4.53%	5.10%
	EM	8.91%	6.40%	2.75%	2.67%	3.16%

limited because the speech signal space is not infinitely divisible. An excessively large codebook leads to the excessive classification of the signal space, so that several codebook vectors will correspond to the same subspace which describes part of an utterance, say, a phoneme [10, 12]. This will cause confusion for the finite state hidden Markov model and degrade system performance eventually. We can see from Table 2 that the results of MGC and EM in the codebook size 512 case are worse than that in codebook size 256 case. This indicates that the speech space is adequately classified using GMM when the codebook size is 256. When we increase the codebook size to 512, the quantisation is excessive. This probably is why the codebook sizes of most discrete HMM systems are limited in the range of 64 to 256. In the LBG case, larger codebooks require greater amount of computation, but yield comparatively small gains in recognition accuracy. Rabiner [3] suggests a codebook size of 64 for the isolated digits recognition task. In our experience a codebook with 128 members has been found to represent a reasonable balance between the amount of computation required and the resulting recognition accuracy.

The weights of three individual recognisers were determined according to the following formula:

$$W_i = 1/E_i^{\Theta} \tag{8}$$

where W_i is the weight of ith subrecogniser, E_i is the recognition error rate of ith subrecogniser obtained from the individual experiements and Θ is a constant between 2 and 3 which was chosen by experiment to give the best result. In our system, $\Theta = 2.3$ gives best performance.

Table 3 shows the results achieved by the two-model systems which were composed by combining any two of the three subrecognisers and Table 4 shows the results

Table 3: Results of two-model recognisers

Codebook size		32	64	128
Error rate	LBG/MGC	10.04%	5.67%	4.37%
	LBG/EM	7.13%	4.86%	2.19%
	MGC/EM	7.77%	4.94%	2.51%

Table 4: Results of three-model recogniser

Codebook size	32	64	128
Error rate	5.47%	2.89%	1.30%

achieved by the three-model system. In the two model cases the LBG/EM combination gives a better result than the MGC/EM; however, the individual LBG/HMM recogniser gives worse results than the MGC/HMM recogniser. The reason for this is that both the EM and MGC algorithms cluster the speech space using Gaussian mixtures and will generate many similar wrong recognitions. The LBG algorithm, however, does a tessellation of the speech space and many of its wrong recognitions will be different from those of either EM or MGC. Thus when

the EM and LBG algorithms are combined and weighted properly, the performance was expected to be better than the MGC/EM combination. Compared with the best single recogniser, EM/HMM, the three-model system obtained 38.8%, 54.9%, and 53.1% reduction in the recognition error rates for the codebook sizes 32, 64, and 128 respectively.

The different speech features, LPC coefficients, LPC derived cepstral coefficients, and differenced cepstral coefficients, were also investigated [8, 9]. The filter bank features gave the best performance.

5 Conclusion and discussion

Consideration of the speech production process suggests that Gaussian mixture models offer a good description, and the EM algorithm is an effective classification method for the first step modelling in a VQ/HMM system.

The multiple hidden Markov model speech recogniser gave better recognition results. This was shown by the performance of combinations of any two of three subrecognisers and combination of three of them together. The best results achieved by VQ/MHMM recogniser represent a reduction in the error rate of about 50% in comparison to the EM/HMM recogniser, the best single recogniser.

The weights need to be carefully chosen on the basis of the recognition accuracies of the subrecogniser obtained from separate experiments. Inappropriate weights could lead to an insignificant improvement of the recognition results.

This system requires more computation and more recognition time when implemented in software than the individual recognisers. However, for a hardware system, we can implement the subrecognisers in parallel to improve the recognition results without increasing recognition time. Compared with other methods which employ different speech features and form larger feature vectors to improve the recognition accuracy [3, 11, 13, 14], this VQ/MHMM system looks more effective for building a real-time hardware speech recognition system.

To conclude, we compare the performance of our VQ/MHMM system with some other results in the literature.

The SPHINX system uses multiple codebooks and a single HMM to improve the system performance [11]. The codebooks are generated from three features, bilinear transformed LPC cepstrum coefficients, differenced bilinear transformed LPC cepstrum coefficients, and a weighted combination of the power and the differenced power. In that system three different indices from three codebooks are used to form a vector and a modified HMM observes that vector rather than a single index of the codewords from the codebook. This system requires more computation, not only for the separate calculation of the coefficients, but also for the Viterbi beam-search when the observation is a vector. If implemented in parallel, our MHMM system requires about same computation time as a normal single HMM system, both for the computation of coefficients and the Viterbi search.

Gregory *et al.* [13] used the same data base (TIDIGITS) and obtained 97.2% recognition accuracy. They employed a feature maps neural network and a very big feature vector, 17 dimensional filter bank coefficients plus zero crossing rate and log RMS energy. Another interesting result is from Yadong *et al.* [14]. They used the dynamic time warping system and with neural network trained templates. They obtained 99.6% recognition accuracy but only used the utterances from female speakers in the TIDIGITS data base and employed a 30 dimensional feature vector. With a feature vector of dimension 15, their best result was 98.33% for female speakers only.

6 References

1 DEMPSTER, A.P., LAIRD, N.M., and RUBIN, D.B.: 'Maximum likelihood from incomplete data via the EM algorithm', *J. Roy. Stat. Soc. B*, 1977, **39**, (1), pp. 1–38

2 LEONARD, R.G.: 'A database for speaker-independent digit recognition'. Proceedings of the IEEE ICASSP Conference, 1984, Vol. 3, pp. 42.11.1–4

3 RABINER, L.R., LEVINSON, S.E., and SONDHI, M.M.: 'On the application of vector quantisation and hidden Markov models to speaker-independent, isolated word recognition', *Bell Syst. Tech. J.*, 1983, **62**, (4), pp. 1075–1105

4 LINDE, Y., BUZO, A., and GRAY, R.M.: 'An algorithm for vector quantiser design', *IEEE Trans.*, 1980, **COM-28**, (1), pp. 84–95

5 RABINER, L.R.: 'A tutorial on hidden Markov models and selected applications in speech recognition', *Proc. IEEE*, 1989, **77**, (2), pp. 257–286

6 TOGNERI, R., ZHANG, Y., DESILVA, C., and ATTIKIOUZEL, Y.: 'A comparison of the LVQ and EM algorithm for vector quantisation in a speaker-independent digit recognition task'. Proceedings of ISSPA 92 (The third international symposium on signal processing and its applications, Gold Coast, Australia), 1992, pp. 384–387

7 WOLFE, J.H.: 'Pattern clustering by multivariate mixture analysis', *Multivariate Behav. Res.*, 1970, **5**, pp. 329–350

8 ZHANG, Y., and DESILVA, C.: 'An isolated word recogniser using the EM algorithm for vector quantisation'. Proceedings of IREECON 91 (Australia's Electrical and Electronic Convention 91, Sydney, Australia), 1991, pp. 289–292

9 ZHANG, Y., DESILVA, C., ATTIKIOUZEL, Y., and ALDER, M.: 'A HMM/EM speaker-independent isolated word recognizer', *J. Elect. Electr. Eng. Aust.*, 1992, **12**, pp. 334–339

10 ZHANG, Y., TOGNERI, R., and ALDER, M.: 'Using Gaussian mixture modelling for phoneme classification'. Proceedings of ANZIIS-93 (first Australian and New Zealand conference on intelligent information systems, Perth), 1993, pp. 649–652

11 LEE, K.-F., HON, H.-W., and REDDY, R.: 'An overview of the SPHINX speech recognition system', *IEEE Trans.*, 1990, **ASSP-38**, (1), pp. 35–45

12 PIJPERS, M., ALDER, M., and TOGNERI, R.: 'Finding structure in the vowel space'. Proceedings of ANZIIS-93 (First Australian and New Zealand conference on intelligent information systems, Perth), 1993, pp. 658–662

13 DE HAAN, G.R., and ECECIOGLU, O.: 'Feature maps for input normalisation and feature integration in a speaker-independent isolated digit recognition system'. Proceedings of international joint conference on neural networks, Baltimore, MD, USA, 1992, pp. 685–690

14 LIN, Y., LEE, Y.-C., CHEN, H.-H., and SUN, G.-Z.: 'Speech recognition using dynamic time warping with neural network trained templates'. Proceedings of international joint conference on neural networks, Baltimore, MD, USA, 1992, pp. 326–331