Available online at www.sciencedirect.com
SCIENCE DIRECT
Speech Communication 48 (2006) 746–758
SPEECH COMMUNICATION
www.elsevier.com/locate/specom

Scalable distributed speech recognition using Gaussian mixture model-based block quantisation

Stephen So *, Kuldip K. Paliwal

School of Microelectronic Engineering, Griffith University, Brisbane QLD 4111, Australia

Received 6 July 2004; received in revised form 6 July 2005; accepted 20 October 2005

Abstract

In this paper, we investigate the use of block quantisers based on Gaussian mixture models (GMMs) for the coding of Mel frequency-warped cepstral coefficient (MFCC) features in distributed speech recognition (DSR) applications. Specifically, we consider the multi-frame scheme, where temporal correlation across MFCC frames is exploited by the Karhunen–Loève transform of the block quantiser. Compared with vector quantisers, the GMM-based block quantiser has relatively low computational and memory requirements which are independent of bitrate. More importantly, it is bitrate scalable, which means that the bitrate can be adjusted without the need for re-training. Static parameters such as the GMM and transform matrices are stored at the encoder and decoder and bit allocations are calculated 'on-the-fly' without intensive processing. We have evaluated the quantisation scheme on the Aurora-2 database in a DSR framework. We show that jointly quantising more frames and using more mixture components in the GMM leads to higher recognition performance. The multi-frame GMM-based block quantiser achieves a word error rate (WER) of 2.5% at 800 bps, which is less than 1% degradation from the baseline (unquantised) word recognition accuracy, and graceful degradation down to a WER of 7% at 300 bps.

Keywords: Distributed speech recognition; Gaussian mixture models; Block quantisation; Aurora-2

1. Introduction

With the increase in popularity of remote and wireless devices such as personal digital assistants (PDAs) and cellular phones, there has been a growing interest in applying automatic speech recognition (ASR) technology in the context of mobile communication systems. Speech recognition can facilitate consumers in performing common tasks, which have traditionally been accomplished via buttons or pointing devices, such as making a call through voice dialing or entering data into their PDAs via spoken commands and sentences. Some of the issues that arise when implementing ASR on mobile devices include: computational and memory constraints of the mobile device; network bandwidth utilisation; and robustness to noisy operating conditions.

* Corresponding author. Present address: School of Engineering, Griffith University, PMB 50 Gold Coast Mail Centre, QLD 9726, Australia. Tel./fax: +61 7 5552 8496 (8065).
E-mail addresses: s.so@griffith.edu.au (S. So), k.paliwal@griffith.edu.au (K.K. Paliwal).

0167-6393/$ - see front matter
doi:10.1016/j.specom.2005.10.002

Mobile devices generally have limited storage and processing ability which makes implementing a full on-board ASR system impractical. The solution to this problem is to perform the complex speech recognition task on a remote server that is accessible via the network. Various modes of this client–server approach have been proposed and reported in the literature. In the *network speech recognition* (NSR) mode (Kiss, 2000), the user's speech is compressed using conventional speech coders (such as the GSM speech coder) and transmitted to the server which performs the recognition task. In speech-based NSR (Fig. 1(a)), the server calculates ASR features from the decoded speech to perform the recognition. In bitstream-based NSR (Fig. 1(b)), the server uses ASR features that are derived from linear predictive coding (LPC) parameters taken directly from the bitstream. Numerous studies have been reported in the literature evaluating and comparing the performance of these two forms of NSR (Hirsch, 1998; Huerta and Stern, 1998; Kim and Cox, 2001; Lilly and Paliwal, 1996; Raj et al., 2001; Turunen and Vlaj, 2001; Gallardo-Antolin et al., 1998).

In *distributed speech recognition* (DSR), shown in Fig. 1(c), the ASR system is distributed between the client and server. Here, the feature extraction of speech is performed at the client. These ASR features are compressed and transmitted to the server via a dedicated channel, where they are decoded and input into the ASR backend. Studies have shown that DSR generally performs better than NSR (Kiss, 2000) because, in the latter model,

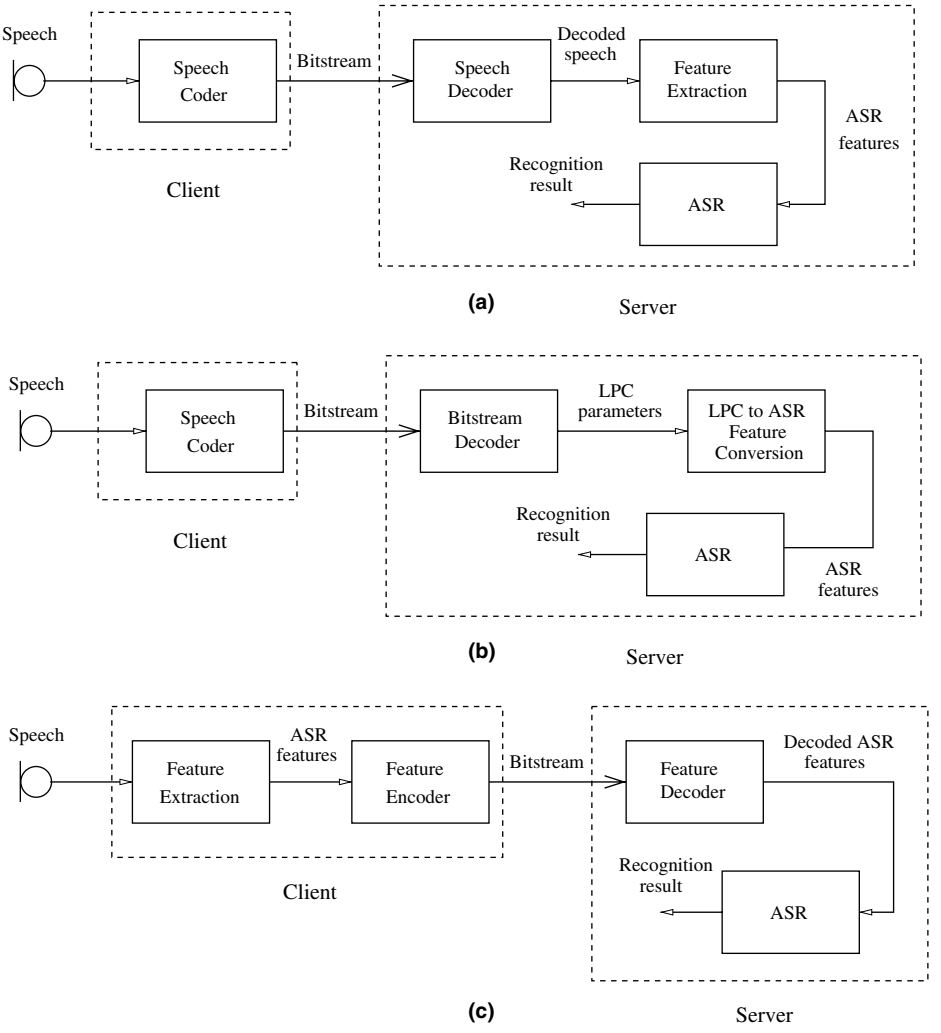

Fig. 1. Client–server-based speech recognition modes: (a) speech-based network speech recognition (NSR), (b) bitstream-based network speech recognition, and (c) distributed speech recognition (DSR).

speech is processed for optimal perceptual quality and this does not necessarily result in optimal recognition performance (Srinivasamurthy et al., 2003).

Various schemes for compressing the ASR features have been proposed in the literature. Digalakis et al. (1999) evaluated the use of uniform and non-uniform scalar quantisers as well as product code vector quantisers for compressing Mel frequency-warped cepstral coefficients (MFCCs) between 1.2 and 10.4 kbps. They concluded that split vector quantisers achieved word error rates (WER) similar to that of scalar quantisers while requiring less bits. Also, scalar quantisers with non-uniform bit allocation performed better than those with uniform bit allocation. Ramaswamy and Gopalakrishnan (1998) investigated the application of tree-searched multistage vector quantisers with one-step linear prediction operating at 4 kbps. Transform coding, based on the discrete cosine transform (DCT), was investigated in Kiss and Kapanen (1999) at 4.2 kbps and in Zhu and Alwan (2001) which used a two-dimensional DCT. The ETSI DSR standard (STQ, 2000) uses split vector quantisers to compress the MFCC vectors at 4.4 kbps. Srinivasamurthy et al. (2003) exploited correlation across consecutive MFCC features by using a DPCM scheme followed by entropy coding.

Even though vector quantisers generally give better recognition performance using less bits, they are not scalable in bitrate when compared with scalar quantiser-based schemes, such as DPCM and transform coders. In other words, the vector quantiser is designed to operate at a specific bitrate only and will need to be re-trained for other bitrates. *Bitrate scalability* is a desirable feature in DSR applications, since one may need to adjust the bitrate adaptively, depending on the network conditions. For instance, if the communications network is heavily congested, then it may be more acceptable to sacrifice some recognition performance by operating at a lower bitrate in order to offset long response times. In addition to this, the computational complexity of vector quantisers can be quite high, when compared with scalar quantiser-based schemes.

Block quantisation or transform coding,[1] has been used as a less complex alternative to fullsearch vector quantisation in the coding literature. Proposed by Kramer and Mathews (1971) and analysed by Huang et al. (1963), it involves decorrelating the components within a block or vector of samples using a linear transformation before scalar quantising each component independently. When quantising for minimum mean-squared-error (MSE), the Karhunen–Loève transform (KLT)[2] is the best transform for correlated Gaussian sources (Goyal, 2001). However, the probability density functions (PDF) of real life sources are rarely Gaussian and any PDF mismatch with the quantiser will invariably cause a degradation in performance. Rather than assuming the source PDF to be a standard function such as Gaussian, Laplacian, etc., one can design a quantiser that matches the source PDF as close as possible.

There have been numerous studies in the coding literature on source PDF modelling for quantiser design. These can be broadly classified as either *non-parametric or parametric modelling*. Ortega and Vetterli (1996) estimated the source model in a non-parametric fashion using piecewise linear approximation. Similarly, multidimensional histograms were used by Gardner and Rao (1995) to model the PDFs of line spectral frequencies (LSF) in order to evaluate the theoretical bounds of split vector quantisers. In relation to parametric modelling, Su and Mersereau (1996) applied Gaussian mixture models (GMM) in the estimation of the PDF of DC transform coefficients while Archer and Leen (2004) used GMMs to form a probabilistic latent variable model from which transform coding design algorithms were derived. On the speech side, Hedelin and Skoglund (2000) used GMMs with bounded support for designing and evaluating high-rate vector quantisers for LSF coding while Samuelsson and Hedelin (2001) extended this work to recursive spectral coding.

Subramaniam and Rao (2003) incorporated the PDF model into the block quantisation of LSFs via GMMs and in our previous work (Paliwal and So, 2004a), we have extended this scheme to exploit memory across successive frames. Even though this quantisation scheme is not as optimal as vector quantisation, it nevertheless possesses the following advantages (Subramaniam and Rao, 2003):

[1] While these two terms are generally synonymous, the latter term generally encompasses coders that use variable rate entropy coding after the quantisation step, while block quantisers use non-uniform scalar quantisers of fixed rate with no entropy coding (Huang et al., 1963). This paper is focused on block quantisation only.

[2] Often called an eigenvector transform, Hotelling transform, or principal component analysis (PCA) in the pattern recognition literature.

- Compact representation of the source PDF which is independent of bitrate;
- bitrate scalability with 'on-the-fly' bit allocation; and
- low search complexity and memory requirements which are independent of the rate of the system.

In this paper,[3] we investigate the use of the fixed-rate, multi-frame GMM-based block quantisation scheme of Paliwal and So (2004a) for DSR applications[4] that is computationally simpler than vector quantisers, is scalable in bitrate, and leads to a more graceful degradation in recognition performance when compared with other scalar quantiser-based schemes.

The organisation of this paper is as follows. We give a brief description of the multi-frame GMM-based block quantiser, bit allocation, as well as its computational and memory requirements in Section 2. In Section 3, we describe the setup of our recognition experiments on the Aurora-2 database. Following this, in Section 4, we present and discuss the recognition results for the single frame and multi-frame GMM-based block quantiser and compare them with results from the non-uniform scalar quantiser and vector quantiser. Finally, we offer our conclusions in Section 5 as well as further work that needs to be done.

2. Multi-frame GMM-based block quantization

This quantisation scheme is based on the one proposed by Subramaniam and Rao (2003) for the quantising of speech line spectral frequencies (LSF), where a Gaussian mixture model (GMM) is used to parametrically model the probability density function (PDF) of the source and block quantisers are then designed for each Gaussian mixture component. In Paliwal and So (2004a), we proposed a modified scheme which used vectors formed by concatenating p successive frames, in order to exploit interframe correlation. Therefore, if the length of the MFCC frame is n, then the dimensions of the vectors processed will be np. MFCCs are calculated frame-wise from speech and there is considerable overlap between successive frames. Generally, there will be high correlation between consecutive frames (Srinivasamurthy et al., 2003). Therefore, we have chosen to use multi-frame GMM-based block quantisation to exploit this correlation. For more details on this quantisation scheme, the reader is referred to (Subramaniam and Rao, 2003; Paliwal and So, 2004a).

The multi-frame GMM-based block quantiser can be broken down into three stages: PDF estimation, bit allocation and minimum distortion block quantisation. Figs. 2 and 3 show the PDF estimation and bit allocation procedure and minimum distortion block quantisation, respectively. These are similar to those that appear in Subramaniam and Rao (2003).

2.1. PDF estimation using Gaussian mixture models

The PDF model and Karhunen–Loève transform (KLT) orthogonal matrices are the only static and bitrate-independent parameters of the GMM-based block quantiser. These only need to be calculated once (training) and stored at the client encoder and server decoder. The bit allocations for different bitrates (described in Section 2.2.1) can be calculated 'on-the-fly' using the common PDF model stored on both the client and server. Hence this scheme is bitrate scalable (Subramaniam and Rao, 2003).

The PDF model, G, as a mixture of multivariate Gaussians, $\mathcal{N}(\boldsymbol{x};\boldsymbol{\mu},\boldsymbol{\Sigma})$, can be expressed as

$$G(\boldsymbol{x}|\boldsymbol{M}) = \sum_{i=1}^{m} c_i \mathcal{N}(\boldsymbol{x};\boldsymbol{\mu}_i,\boldsymbol{\Sigma}_i) \tag{1}$$

$$\boldsymbol{M} = [m, c_i, \ldots, c_m, \boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_m, \boldsymbol{\Sigma}_1, \ldots, \boldsymbol{\Sigma}_m] \tag{2}$$

$$\mathcal{N}(\boldsymbol{x};\boldsymbol{\mu},\boldsymbol{\Sigma}) = \frac{1}{(2\pi)^{\frac{n}{2}}|\boldsymbol{\Sigma}|^{\frac{1}{2}}} e^{-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu})^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}(\boldsymbol{x}-\boldsymbol{\mu})} \tag{3}$$

where $\boldsymbol{x}$ is a source vector, m is the number of mixture components, and n is the dimensionality of the vector space. c_i, $\boldsymbol{\mu}_i$ and $\boldsymbol{\Sigma}_i$ are the weight, mean, and covariance matrix of the ith mixture component, respectively.

The parametric model, M, is initialised by applying the K-means algorithm (Linde et al., 1980) on the training vectors representing the source distribution where m mixture components are produced, each represented by a mean or centroid, $\boldsymbol{\mu}$, a covariance matrix, $\boldsymbol{\Sigma}$, and a mixture component weight, c. These form the initial parameters for the GMM estimation procedure. Using the EM algorithm

[3] This paper is an extended version of Paliwal and So (2004b) and contains more comparative results.

[4] While we are quantising MFCC features for DSR, this quantiser can also be applied in a CELP speech coder to be used in NSR.

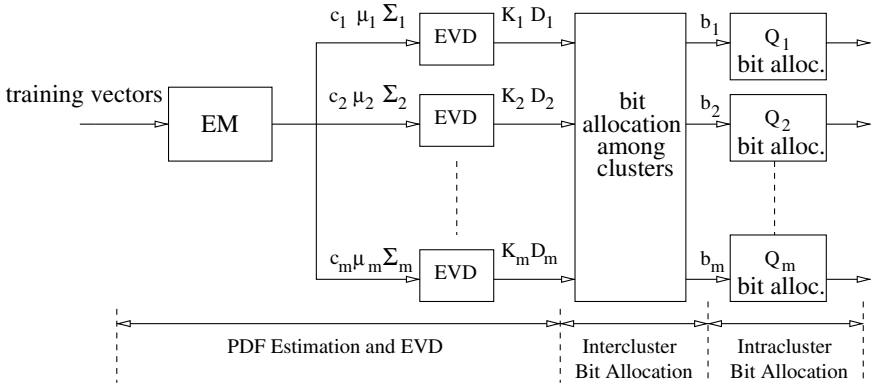

Fig. 2. PDF estimation and bit allocation from training data.

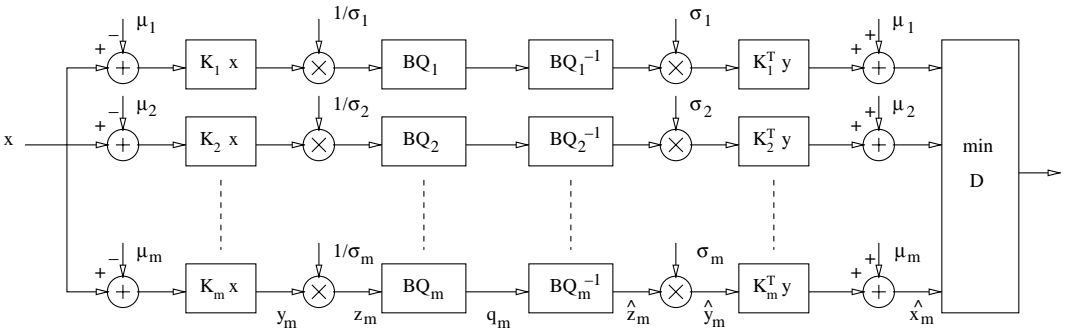

Fig. 3. Block diagram of the GMM-based block quantiser (BQ—block quantiser).

(Dempster et al., 1977), the maximum-likelihood estimate of the parametric model is computed iteratively and a final set of means, covariance matrices, and weights are produced.

An eigenvalue decomposition (EVD) is calculated for each of the m covariance matrices, producing m eigenvalue vectors, i.e. a total of nm eigenvalues; $\{\lambda_{i,j};\ i=1,\ldots,n;\ j=1,\ldots,m\}$ for single frame and npm eigenvalues; $\{\lambda_{i,j};\ i=1,\ldots,np;\ j=1,\ldots,m\}$ for multi-frame. The eigenvectors form the rows of the orthogonal transformation matrix, $\boldsymbol{K}$, of the KLT.

2.2. Bit allocation

2.2.1. Bit allocation among mixture components

Assuming that there are b_{tot} bits available for quantising each vector (for an average bitrate of b_{tot}/np bits per sample, where p is the number of feature frames within each vector), these need to be allocated to each of the block quantisers for each mixture component.

Following the derivation given in Subramaniam and Rao (2003), for a fixed-rate quantiser, the total number of quantiser levels is fixed:

$$2^{b_{\text{tot}}} = \sum_{i=1}^{m} 2^{b_i} \tag{4}$$

where b_i is the number of bits assigned to mixture component i, and m is the number of mixture components. Since the mixture component weights can be thought of as probabilities of occurrence of each mixture component (Subramaniam and Rao, 2003), the average distortion is approximated by (assuming there is negligible overlaps between mixture components):

$$D_{\text{tot}} = \sum_{i=1}^{m} c_i D_i(b_i) \tag{5}$$

Using the high resolution approximation for distortion of a single Lloyd–Max scalar quantiser, the total distortion of a block quantiser is (Subramaniam and Rao, 2003):

$$D_i(b_i) = KnpA_i 2^{-2\frac{bi}{np}} \tag{6}$$

$$A_i = \left[\prod_{j=1}^{np} \lambda_{i,j}\right]^{\frac{1}{np}} \quad \text{for } i = 1, 2, \ldots, m \tag{7}$$

where n is the dimension of the vectors, p is the number of concatenated frames, m is the number of mixture components, $\lambda_{i,j}$ is the jth eigenvalue of mixture component i, and K is a constant which is equal to $\frac{\pi\sqrt{3}}{2}$ for Gaussian sources.

Using Lagrangian minimisation, the distortion can be minimised under the fixed rate constraint of (4), and the following bit allocation formula is derived (Subramaniam and Rao, 2003):

$$2^{b_i} = 2^{b_{\text{tot}}} \frac{(c_i A_i)^{\frac{np}{np+2}}}{\sum_{i=1}^{m} (c_i A_i)^{\frac{np}{np+2}}} \quad \text{for } i = 1, 2, \ldots, m \tag{8}$$

where c_i is the weight of mixture component i.

2.2.2. Bit allocation within mixture components

After allocating the bit budget across all mixture components, the bits need to be allocated to each of the n components within each mixture component using existing bit allocation techniques for transform coding (Gersho and Gray, 1992; Subramaniam and Rao, 2003). Following the derivation presented in Huang et al. (1963), the total number of bits is fixed:

$$b_i = \sum_{j=1}^{np} b_{i,j} \quad \text{for } i = 1, 2, \ldots, m \tag{9}$$

where $b_{i,j}$ is the number of bits assigned to component j of mixture component i. Again, using the high resolution approximation for the distortion of a Lloyd–Max scalar quantiser, the average distortion of mixture component i is given by Huang et al. (1963):

$$D_i = \frac{1}{np} \sum_{j=1}^{np} \lambda_{i,j} K 2^{-2b_{i,j}} \quad \text{for } i = 1, 2, \ldots, m \tag{10}$$

Following (Huang et al., 1963), the average distortion is minimised under the fixed rate constraint of (9) using Lagrangian minimisation to give the following bit allocation formula:

$$b_{i,j} = \frac{b_i}{np} + \frac{1}{2}\log_2 \frac{\lambda_{i,j}}{\left[\prod_{j=1}^{np} \lambda_{i,j}\right]^{\frac{1}{np}}} \quad \text{for } i = 1, 2, \ldots, m \tag{11}$$

2.3. Minimum distortion block quantisation

Fig. 3 shows a diagram of minimum distortion block quantisation. It consists of m independent Gaussian block quantisers, BQ_i, each with their own orthogonal matrix, $\boldsymbol{K}_i$, and bit allocations, $\{b_{i,j}\}_{j=1}^{np}$. The following quantisation process is described in Subramaniam and Rao (2003).

To quantise a vector, $\boldsymbol{x}$, using the block quantiser of a particular mixture component i, the mixture component mean, $\boldsymbol{\mu}_i$, is first subtracted and its components decorrelated using the orthogonal matrix, $\boldsymbol{K}_i$, for that mixture component. The variance of each component is then normalized by the standard deviation to produce a decorrelated, mean-subtracted, and normalized-variance vector, $\boldsymbol{z}_i$:

$$\boldsymbol{z}_i = \frac{\boldsymbol{K}_i(\boldsymbol{x} - \boldsymbol{\mu}_i)}{\boldsymbol{\sigma}_i} \tag{12}$$

These are then quantised using a set of np Gaussian Lloyd–Max scalar quantisers as described in Huang et al. (1963) with their respective bit allocations producing indices, $\boldsymbol{q}_i$. These are decoded to give the approximated normalized vector, $\hat{\boldsymbol{z}}_i$, which is multiplied by the standard deviation and correlated again by multiplying with the transpose, $\boldsymbol{K}_i^{\mathrm{T}}$, of the orthogonal matrix. The mixture component mean is then added back to give the reconstructed vector, $\hat{\boldsymbol{x}}_i$.

$$\hat{\boldsymbol{x}}_i = \boldsymbol{K}_i^{\mathrm{T}} \boldsymbol{\sigma}_i \hat{\boldsymbol{z}}_i + \boldsymbol{\mu}_i \tag{13}$$

The distortion between this reconstructed vector and original is then calculated, $d(\boldsymbol{x} - \hat{\boldsymbol{x}}_i)$. The above procedure is performed for all mixture components in the system and the mixture component, k, which gives the *minimum distortion* is chosen:

$$k = \underset{i}{\operatorname{argmin}}\, d(\boldsymbol{x} - \hat{\boldsymbol{x}}_i) \tag{14}$$

In the case of coding MFCC vectors, we use the mean-squared-error (MSE) as the distortion measure for selecting the appropriate block quantiser.

2.4. Quantiser index encoding

Each quantised vector has associated with it, a number identifying which mixture component was used for coding. As proposed in Subramaniam and Rao (2003), this side information can be made inherent in the encoding. For an m mixture component system operating at b bits per vector, $\log_2 m$ bits are required to uniquely identify each mixture

component. Therefore, on average, $b - \log_2 m$ bits are available for quantising each vector which is equivalent to $2^b/m$ quantiser levels. Hence, our range of quantiser indices has been partitioned into m segments.

In effect, this partitioning of the range of quantiser levels allows the mixture component number to be found by determining which partition the block code belongs to. An example of this encoding scheme is shown in Fig. 4 where a total of 3 bits are available to encode each block and the system uses 2 mixture components. Mixture component 1 has been assigned 5 levels while mixture component 2 has the remaining 3 levels. If mixture component 1 was deemed the most suitable for encoding the current block, its quantiser levels would be contained within the range of 011...111. Therefore, the decoder can easily determine which mixture component the block belongs to by working out which partition the code falls into. Hence this removes the need for extra side information to be transmitted.

2.5. Bitrate scalability, computational complexity and memory requirements

The GMM parameters and Karhunen–Loève transform (KLT) matrices are the only static and bitrate-independent parameters of the multi-frame GMM-based block quantiser. These only need to be calculated once during training and stored at the encoder and decoder. The bit allocation formulas (described in Section 2.2.1) are closed-form expressions, hence they can be evaluated 'on-the-fly' for different bitrates using the static PDF model, by both encoder and decoder. Therefore, this scheme is bitrate scalable, where the bitrate can be changed without quantiser re-training by the encoder and decoder (Subramaniam and Rao, 2003). Vector quantisers, on the other hand, are not bitrate scalable since the codebook is designed only for a specific bitrate.

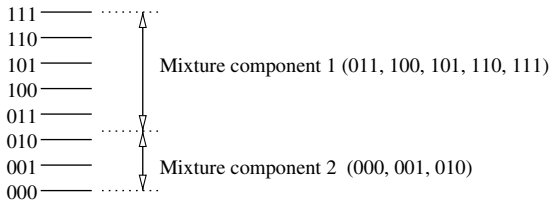

Fig. 4. Example of quantiser level encoding and mixture component number partitioning.

Table 1
Bitrate independent computational complexity (in kflop/frame) and memory requirements (ROM) of the multi-frame GMM-based block quantiser as a function of number of concatenated vectors, p, and number of mixture components, m

m	P	kflop/frame	ROM (floats)
16	1	13.46	3136
	2	22.66	10,624
	3	31.88	22,720
	4	41.09	39,424
	5	50.53	60,736
32	1	26.91	4416
	2	45.33	14,976
	3	63.75	31,936
	4	82.18	55,296
	5	100.6	121,216

One of the salient features of the GMM-based block quantiser scheme is the computational complexity and memory requirements being independent of the operating bitrate (Subramaniam and Rao, 2003). Following the analysis given in Subramaniam and Rao (2003), the computational complexity (in kflop/frame)[5] and memory requirements of the multi-frame GMM-based block quantiser are given in Table 1 for 16 and 32 mixture components. From this table, it can be seen that concatenating more frames to exploit the correlation leads to an increase in computational and memory requirements.

We now approximate the total complexity required for speech coding, MFCC feature extraction, and MFCC quantisation at the client side. Considering the example of a cellular phone as the client, the common speech codec used is the GSM Enhanced Full-Rate (GSM-EFR) codec, which has a computational complexity of approximately 18 Mflops and a static memory requirement of 5900 words (16-bit) (Jarvinen et al., 1997). MFCC feature extraction requires at most 10 Mflops or less (Adami et al., 2002). MFCC features are extracted at a rate of 100 Hz, hence a single-frame GMM-based block quantiser with 16 mixture components would incur a complexity of 1.35 Mflops while a five-frame GMM-based block quantiser with 16 mixture components would require 5.1 Mflops. Therefore, the total complexity at the client side is roughly 29.35 Mflops when using 16 mixture component, single-frame GMM-based block quantisation of

[5] In this study, each addition, multiplication, and comparison is considered one floating point operation (flop).

MFCCs and 33.1 Mflops when using five-frame GMM-based block quantisation.

3. Experimental setup

We have evaluated the recognition performance of various quantisation schemes using the publicly available HTK 3.2 software on the ETSI Aurora-2 database (Hirsch et al., 2000). The purpose of the Aurora-2 database is to provide a common framework for evaluating the DSR-related issues in a connecteddigit recognition task. It consists of a clean (or, noise-free) speech database,[6] a noise database, a standard MFCC-based frontend, and scripts for performing the training and recognition task.

We have limited the focus of this work on recognition performance versus bitrate on clean speech only. Quantiser training was performed on the feature vectors derived from clean speech training data (8440 utterances) and quantisation was performed on the feature vectors derived from the clean speech data of test set A (4004 utterances).[7] Whole word HMMs are used for modelling the digits with the following parameters (Hirsch, 1998):

- 16 states per word (with two dummy states at beginning and end);
- left-to-right topology without skips over states;
- three Gaussian mixtures per state; and
- diagonal covariance matrices.

For more details on the HTK reference recogniser and Aurora frontend, the reader should refer to (Hirsch et al., 2000; STQ, 2000).

The ETSI DSR standard Aurora frontend (STQ, 2000) was used for the MFCC feature extraction. As a slight departure from the ETSI DSR standard, we have used 12 MFCCs (excluding the zeroth cepstral coefficient, c_0, and logarithmic frame energy, $\log E$) as the feature vectors to be quantised. This was done to maintain consistency in the quantisation scheme as c_0 and $\log E$ are sensitive to changes in recording level of a speech utterance and are generally coded independently (STQ, 2000; Kiss and Kapanen, 1999; Ramaswamy and Gopalakrishnan, 1998).

It is well known that lower order cepstral coefficients are particularly sensitive to undesirable variations caused by factors such as transmission, speaker characteristics, and vocal efforts, etc. (Juang et al., 1987). As bits are distributed on the basis of the variance of each MFCC, the bit allocations will be particularly sensitive to these spectral variations. In our scalar quantiser experiments, we have found this to degrade the performance of the recognition as too many bits are given to the lower order MFCCs. Therefore, in order to reduce the effect of these variations on bit allocation, we have applied the cepstral liftering technique of Juang et al. (1987) to the MFCCs using the following window function, $w(n)$ (Juang et al., 1987):

$$w(n) = 1 + \frac{L}{2}\sin\left(\frac{\pi n}{L}\right) \quad \text{where } n = 1, 2, \ldots, L \tag{15}$$

where L is the feature length. The liftered MFCC feature vectors are then encoded and transmitted. At the decoding side, cepstral mean subtraction is applied to the decoded 12 MFCC features, which are concatenated with their corresponding delta and acceleration coefficients, giving the final feature vector dimension of 36 for the ASR system. The HTK parameter type is `MFCC_D_A_Z`. The baseline recognition accuracy using unquantised MFCC features is 98.0%. Assuming each floating point value requires 32 bits, the effective bitrate required for transmitting unquantised MFCC features is 38.4 kbps.

In the training of the single frame and multiframe GMM-based block quantiser, 20 iterations of the EM algorithm were used to generate a 16 mixture component GMM. The unquantised MFCC vectors are used to train the quantiser models, hence they are bitrate independent.

4. Results and discussion

4.1. Recognition performance of the single frame GMM-based block quantiser

Table 2 shows the recognition accuracy for the single frame GMM-based block quantiser at various bitrates and number of mixture components. At 2 kbps, the recognition accuracy is roughly the same as the unquantised scheme. Between 2 kbps and 800 bps, the recognition performance gradually decreases where it can be seen that the

[6] This is based on the TIDigits database, where the 20 kHz speech was downsampled to 8 kHz and filtered using the ITU-T G.712 characteristic.

[7] Test set A consists of four subsets of 1001 utterances. The recognition accuracies of each subset are averaged to give a final average recognition accuracy for the specific quantisation scheme.

Table 2
Average word recognition accuracy as a function of bitrate and number of mixture components (MC) for single frame GMM-based block quantiser (baseline accuracy = 98.0%)

Bitrate (kbps)	Recognition accuracy (in %)				
	2 MC	4 MC	8 MC	16 MC	32 MC
0.3	23.5	20.0	16.7	8.1	8.1
0.4	43.5	53.3	57.7	23.3	9.1
0.6	68.7	79.7	85.7	87.6	82.0
0.8	86.2	90.3	91.5	93.7	94.5
1.0	90.5	94.2	95.0	95.5	96.1
1.2	93.9	95.9	95.9	96.4	96.7
1.5	96.0	96.5	97.0	97.2	97.2
1.7	97.0	97.0	97.2	97.3	97.4
2.0	97.3	97.2	97.5	97.6	97.7
2.2	97.6	97.3	97.6	97.7	97.7
2.4	97.6	97.5	97.7	97.9	97.7
3.0	97.9	97.8	97.9	97.8	97.9
4.4	98.0	98.0	98.1	98.0	98.0

schemes which uses a larger number of mixture components maintain a higher accuracy. This may be attributed to the more accurate modelling of the source PDF by using more mixture components. Since it has been shown in other studies that using more mixture components generally will reduce the quantisation distortion incurred at a fixed bitrate, it would be expected to indirectly lead to better recognition.

At bitrates below 800 bps, the recognition performance drops dramatically, where we have the situation of increasing the number of mixture components leading to steeper decreases. For MFCC frames containing no information (all zero), the recognition accuracy is 8.1%. This situation may be explained by the shortage of bits to be allocated to all mixture components. A 16 mixture component quantiser, for instance, requires at least 4 bits in total to be able to uniquely identify each mixture component (assuming a uniform allocation of levels) while a 32 mixture component quantiser requires at least 5 bits.[8] Therefore, the single frame GMM-based block quantiser performs poorly when the number of bits approaches $\log_2 m$, where m is the number of mixture components.

[8] In reality, quantiser levels are non-uniformly allocated in this scheme, so most of the available quantiser levels will be allocated to only a fraction of the mixture component quantisers. The decoder will be able to determine which mixture component quantisers are operational since it performs an identical bit allocation using the same stored models as the encoder.

4.2. Recognition performance of the multi-frame GMM-based block quantiser

Table 3 shows the average word recognition accuracy of the 16 mixture component multi-frame GMM-based block quantiser for different bitrates and number of frames. It can be observed that this quantiser achieves an accuracy close to the unquantised, baseline system at 1 kbps or 10 bits/frame, which is half the bitrate of the single-frame GMM-based block quantiser. For bitrates lower than 600 bps, the performance gradually rolls off.

In terms of quantiser distortion, the multi-frame GMM-based block quantiser generally performs better as more frames are concatenated together because interframe memory can be exploited by the KLT. Also, because the dimensionality of the vectors is high, the block quantisers operate at a higher rate. Comparing Table 3 with the 16 mixture component column of Table 2, it can be observed that there is a trend between using more frames to reduce MFCC frame distortion and improving the recognition accuracy, at low bitrates. In other words, the average recognition accuracy gets progressively better as more and more frames are jointly quantised. At 300 bps, the recognition accuracy of jointly quantising five frames is roughly 14% higher than quantising 2 frames only. However, this comes at the expense of higher delay, computational, and memory requirements.

Compared with the results of the single frame GMM-based block quantiser in Table 2, the multi-frame scheme does not suffer from a dramatic drop in recognition accuracy at low bitrates. Unlike the single frame scheme, where there was a shortage of

Table 3
Average word recognition accuracy as a function of bitrate and number of frames for 16 mixture component multi-frame GMM-based block quantiser (baseline accuracy = 98.0%)

Bitrate (kbps)	Recognition accuracy (in %)			
	2 Frames	3 Frames	4 Frames	5 Frames
0.3	78.3	89.6	91.3	93.0
0.4	91.1	94.3	95.1	95.4
0.6	95.5	96.6	97.1	96.8
0.8	96.9	97.3	97.4	97.5
1.0	97.4	97.6	97.7	97.7
1.2	97.6	97.7	97.8	97.9
1.5	97.8	97.8	97.9	97.8
1.7	97.8	98.0	98.0	98.0
2.0	98.0	97.9	98.1	98.0
2.2	98.0	98.0	97.9	98.0

bits to distribute among the mixture components, the multi-frame GMM-based block quantiser is able to provide enough bits, thanks to the increased dimensionality of the vectors. For example, at 300 bps, a 16 mixture component, single frame GMM-based block quantiser has a total bit budget of 3 bits. On the other hand, a 16 mixture component, 2 frame multi-frame GMM-based block quantiser has 6 bits while a 3 and 4 frame scheme has 9 and 12 bits, respectively. Therefore, the multi-frame GMM-based block quantiser can operate at lower bitrates while maintaining good recognition performance.

Table 4 shows the average word recognition accuracy of a 16 mixture component and 32 mixture component multi-frame GMM-based block quantiser, where the number of frames is fixed at 5. As expected, using more mixture components to reduce the quantised MFCC distortion has led to an improvement in recognition accuracy, at the cost of an increase in complexity and memory.

4.3. Comparison with the recognition performance of the non-uniform scalar quantiser

For the scalar quantisation experiment, each MFCC was quantised using a non-uniform Gaussian Lloyd–Max scalar quantiser whose bit allocation was calculated using the high resolution formula of (11). We have chosen this method over the WER-based greedy algorithm of Digalakis et al. (1999) because of its computational simplicity and this allows us to scale any bitrate with ease.

Table 5 shows the average recognition accuracy of the non-uniform scalar quantiser. It can be seen that the accuracy decreases linearly in the range of 4.4–1.2 kbps and drops rapidly below this range.

Table 4
Average word recognition accuracy as a function of bitrate and number of mixture components (MC) for 5 frame multi-frame GMM-based block quantiser (baseline accuracy = 98.0%)

Bitrate (kbps)	Recognition accuracy (in %)	
	16 MC	32 MC
0.2	83.0	87.7
0.3	93.0	94.2
0.4	95.6	96.0
0.6	96.8	97.1
0.8	97.5	97.6
1.0	97.7	97.6
1.2	97.9	97.9
1.5	97.8	98.0
1.7	98.0	98.0
2.0	98.0	98.0

Table 5
Average word recognition accuracy as a function of bitrate for non-uniform scalar quantiser (baseline accuracy = 98.0%)

Bitrate (kbps)	Recognition accuracy (in %)
0.6	38.2
0.8	72.3
1.0	86.7
1.2	93.3
1.5	95.5
1.7	96.2
2.0	97.0
2.2	97.2
2.4	97.4
3.0	97.8
4.4	98.0

Comparing Table 5 with Tables 2 and 3, the GMM-based block quantisers use less bits than the non-uniform scalar quantiser to achieve a certain level of recognition accuracy. This may be attributed to the effectiveness of the KLT as a decorrelator as well as the better modelling of the source PDF.

4.4. Comparison with the recognition performance of the vector quantiser

An unconstrained, full-search vector quantiser was used to quantise single MFCC frames. In terms of minimising quantiser distortion, the vector quantiser is considered the optimum coding scheme (Gersho and Gray, 1992), hence it will serve as an informal upper recognition bound for single frame quantisation and highlight the effectiveness of the multi-frame GMM-based block quantiser in exploiting interframe memory. Table 6 shows the average recognition accuracies at several bitrates as well as the computational and memory requirements of the vector quantiser. Comparing this with Table 2, the vector quantisation scheme achieves higher recognition than the single frame

Table 6
Average word recognition accuracy, computational complexity (in kflop/frame), and memory requirements (ROM) as a function of bitrate for vector quantiser (baseline accuracy = 98.0%)

Bitrate (kbps)	Recognition accuracy (in %)	kflop/frame	ROM (in floats)
0.4	77.0	0.77	192
0.6	92.0	3.07	768
0.8	95.7	12.29	3072
1.0	96.9	49.51	12,288
1.2	97.0	196.7	49,152

GMM-based block quantiser for all bitrates, which is consistent with the fact that the vector quantiser will always incur the least distortion of all quantisation schemes for a given dimension. Comparing with the performance of the multi-frame GMM-based block quantiser in Table 3, it can be observed that this scheme gives higher recognition accuracies than the vector quantiser for all bitrates considered. Even the 2 frame, multi-frame GMM-based block quantiser does better than the vector quantiser. Hence this shows that there is a considerable amount of correlation between MFCC frames that can be exploited by quantisation schemes.

As can be seen from Table 6, the computational complexity and memory requirements of the vector quantiser are dependent on the bitrate and can be quite high at medium bitrates like 1.2 kbps. On the other hand, the complexity of the GMM-based block quantiser (shown in Table 1) is constant for bitrates and is dependent only on the number of concatenated feature frames, p, and the number of mixture components, m. Also of note is that, unlike the GMM-based block quantiser, the vector quantiser is *not bitrate scalable*.

4.5. Summary of the recognition performance of all quantisation schemes

Fig. 5 summarises the performance of all quantisation schemes considered in this paper, where the average word recognition accuracy is plotted against bitrate. The top horizontal line represents the baseline (unquantised) recognition accuracy. We can see that the recognition accuracy as a result of using scalar quantisation drops drastically at bitrates lower than 1.2 kbps, while the GMM-based block quantisers and vector quantiser maintain relatively high accuracies. At bitrates between 0.5 and 1 kbps, the vector quantiser maintains a slightly higher accuracy than the single frame GMM-based block quantiser, which is consistent with the well-known rate-distortion superiority of the vector quantiser (for a given dimension). Finally, we note that the multi-frame GMM-based block quantiser is able to achieve reasonable recognition accuracies at bitrates below 0.5 kbps through the exploitation of interframe correlation.

5. Conclusion and further work

In this paper, we have investigated the use of the multi-frame GMM-based block quantiser for the quantising of MFCC features in DSR applications. The strengths of this quantisation scheme are its computational simplicity when compared with vector quantisers, bitrate scalability, and graceful degradation of recognition performance at very low bitrates via effective exploitation of intra-frame and interframe correlation of MFCC frames. Because the PDF model and transformation matrices are independent of bitrate, these static parameters can be stored on the encoder and decoder and

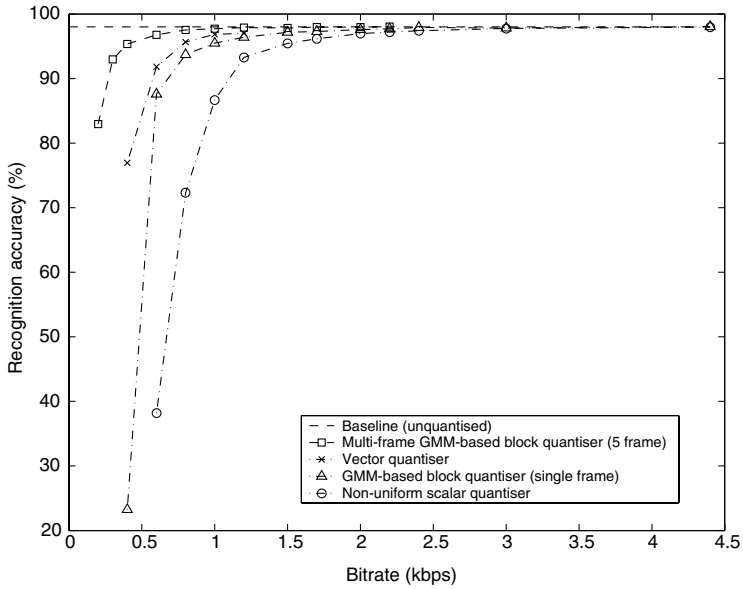

Fig. 5. Summary of average word recognition accuracies for all quantisation schemes considered.

the bitrate can be adjusted 'on-the-fly', depending on network conditions. The recognition performance can improved by increasing the number of the frames to concatenate and using more mixture components, though this comes at the expense of an increase in delay and complexity. Unlike the single-frame version, the multi-frame GMM-based block quantiser does not suffer from a shortage of bits at low bitrates and is able to maintain good recognition performance. On the Aurora-2 database, the 16 mixture component, 5 frame GMM-based block quantiser exhibits a negligible degradation of 1% (WER of 2.5%) in recognition performance over the unquantised, baseline system at 800 bps and 5% (WER of 7%) at 300 bps.

The present work has not taken into account the effect of additive noise, so further experiments need to be performed that utilise the various noises provided by Aurora-2. Furthermore, because the Aurora-2 experiment is essentially a connected-digit recognition task, it has a rather small vocabulary and hence represents a low recognition task difficulty. Further work would involve a large vocabulary continuous speech recognition task on the DARPA Resource Management (RM) database and evaluating the various MFCC quantisation schemes discussed in this paper. In addition to this, because the primary aim of this work was to examine various quantisation schemes for MFCC feature vector coding and to find the best bitrate-versus-recognition performance, other considerations related to the processing and memory demands on the server side, were not discussed. These will also impact on the recognition accuracy of the DSR scheme. Therefore, the issues related to the effects of narrow search beams on the word recognition accuracy and the limitation of the number of ASR channels at the server side are topics for further work.

Acknowledgements

We would like to thank the anonymous reviewers for their detailed comments, which have been extremely helpful in improving the clarity and quality of this paper.

References

Adami, A., Burget, L., Dupont, S., Garudadri, H., Grezl, F., Hermansky, H., Jain, P., Kajarekar, S., Morgan, N., Sivadas, S., 2002. Qualcomm-ICSI-OGI features for ASR. In: Proc. Int. Conf. Spoken Language Processing.

Archer, C., Leen, T.K., 2004. A generalized Lloyd-type algorithm for adaptive transform coder design. IEEE Trans. Signal Process. 52 (1), 255–264.

Dempster, A.P., Laird, N.M., Rubin, D.B., 1977. Maximum likelihood from incomplete data via the EM algorithm. J. Roy. Stat. Soc. 39, 1–38.

Digalakis, V.V., Neumeyer, L.G., Perakakis, M., 1999. Quantization of cepstral parameters for speech recognition over the world wide web. IEEE J. Select. Areas Commun. 17 (1), 82–90.

Gallardo-Antolin, A., Diaz-de-Maria, F., Valverde-Albacete, F., 1998. Recognition from GSM digital speech. In: Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing pp. 1443–1446.

Gardner, W.R., Rao, B.D., 1995. Theoretical analysis of the high-rate vector quantization of LPC parameters. IEEE Trans. Speech Audio Process. 3 (Sept.), 367–381.

Gersho, A., Gray, R.M., 1992. Vector Quantization and Signal Compression. Kluwer Academic Publishers, Massachusetts.

Goyal, V.K., 2001. Therotical foundations of transform coding. IEEE Signal Process. Mag. 18 (5).

Hedelin, P., Skoglund, J., 2000. Vector quantization based on Gaussian mixture models. IEEE Trans. Speech Audio Process. 8 (4), 385–401.

Hirsch, H.G., 1998. The influence of speech coding on recognition performance in telecommunication networks. In: Proc. ICSLP, Denver, USA, September 1998.

Hirsch, H.G., Pearce, D., 2000. The Aurora experimental framework for the performance evaluation of speech recognition systems under noisy conditions. In: ISCA ITRW ASR2000, Paris, France, September 2000.

Huang, J.J.Y., Schultheiss, P.M., 1963. Block quantization of correlated Gaussian random variables. IEEE Trans. Commun. Syst. CS-11 (Sept.), 289–296.

Huerta, J.M., Stern, R.M., 1998. Speech recognition from GSM codec parameters. In: Proc. ICSLP 4, pp. 1463–1466.

Jarvinen, K., Vainio, J., Kapanen, P., Honkanen, T., Haavisto, P., 1997. GSM enhanced full rate speech codec. In: Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing 2, pp. 771–774.

Juang, B.H., Rabiner, L.R., Wilpon, J.G., 1987. On the use of bandpass liftering in speech recognition. IEEE Trans. Acoust., Speech, Signal Process. 35 (July), 947–954.

Kim, H.K., Cox, R.V., 2001. A bitstream-based front-end for wireless speech recognition on IS-136 communications system. IEEE Trans. Speech Audio Process. 9 (5), 558–568.

Kiss, I., 2000. A comparison of distributed and network speech recognition for mobile communication systems. In: Proc. Int. Conf. Spoken Language Processing.

Kiss, I., Kapanen, P., 1999. Robust feature vector compression algorithm for distributed speech recognition. In: Proc. Eurospeech, pp. 2183–2186.

Kramer, K.P., Mathews, M.V., 1971. A linear coding for transmitting a set of correlated signals. IRE Trans. Inform. Theory (Corresp) IT-17 (Nov), 751–752.

Lilly, B.T., Paliwal, K.K., 1996. Effect of speech coders on speech recognition performance. In: Proc. Int. Conf. Spoken Language Processing, vol. 4, pp. 2344–2347.

Linde, Y., Buzo, A., Gray, R.M., 1980. An algorithm for vector quantizer design. IEEE Trans. Commun. 28 (1), 84–95.

Ortega, A., Vetterli, M., 1996. Adaptive scalar quantization without side information. IEEE Trans. Image Proc.

Paliwal, K.K., So, S., 2004a. Multiple frame block quantisation of line spectral frequencies using Gaussian mixture models. In: Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing, Montreal, 1(May), pp. 125–128.

Paliwal, K.K., So, S., 2004b. Scalable distributed speech recognition using multi-frame GMM-based block quantization. In: Proc. Int. Conf. Spoken Language Processing, Jeju, Korea, October 2004.

Ramaswamy, G.N., Gopalakrishnan, P.S., 1998. Compression of acoustic features for speech recognition in network environments. In: Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing, pp. 977–980.

Raj, B., Migdal, J., Singh, R., 2001. Distributed speech recognition with codec parameters. In: Proc. ASRU, Trento, Italy, December 2001.

Samuelsson, J., Hedelin, P., 2001. Recursive coding of spectrum parameters. IEEE Trans. Speech Audio Process. 9 (5), 492–503.

Srinivasamurthy, N., Ortega, A., Narayanan, S., 2003. Efficient scalable encoding for distributed speech recognition, submitted to IEEE Trans. Speech and Audio Processing. Available from: <http://biron.usc.edu/~snaveen/papers/Scalable_DSR.pdf>.

Speech processing, Transmission and Quality aspects (STQ) 2000. Distributed speech recognition; Front-end feature extraction algorithm; Compression algorithms, Tech. Rep. Standard ES 201 108, European Telecommunications Standards Institute (ETSI), April 11, 2000.

Su, J.K., Mersereau, R.M., 1996. Coding using Gaussian mixture and generalized Gaussian models. In: IEEE Int. Conf. Image Processing, Lausanne, Switzerland, pp. 217–220.

Subramaniam, A.D., Rao, B.D., 2003. PDF optimized parametric vector quantization of speech line spectral frequencies. IEEE Trans. Speech Audio Process. 11 (2), 130–142.

Turunen, J., Vlaj, D., 2001. A study of speech coding parameters in speech recognition. In: Proc. Eurospeech, pp. 2363–2366.

Zhu, Q., Alwan, A., 2001. An efficient and scalable 2D DCT-based feature coding scheme for remote speech recognition. In: Proc. IEEE Int. Conf. Acoust., Speech, Signal Processing 1(August), pp. 113–116.