Compression Algorithms for Distributed Classification with Applications to Distributed Speech Recognition

by

Naveen Srinivasamurthy

A Dissertation Presented to the
FACULTY OF THE GRADUATE SCHOOL
UNIVERSITY OF SOUTHERN CALIFORNIA
In Partial Fulfillment of the
Requirements for the Degree
DOCTOR OF PHILOSOPHY
(Electrical Engineering)

August 2006

Dedication

To my parents for their support throughout my studies.

To Vaishali, without your encouragement this would not have been possible.

Acknowledgments

Although words do not do justice to their contributions, I would like to thank the following people for making this work possible.

Firstly my advisors, Prof. Antonio Ortega and Prof. Shrikant Narayanan, whose continual support and guidance fueled my research over the years. I am thankful for the privilege of working with them and I am most grateful for their guidance, patience and advise during my doctoral research.

I would also like to thank, Prof. Boris Rozovsky for being on my dissertation committee, and to Prof. Keith Chugg and Prof. Keith Jenkins for serving on my Qualification exam committee.

Thanks to all my friends and colleagues, in particular, Raghavendra Singh, Hua Xie, Phoom Sagetong, Abhinav Sethy, Shadi Ganjavi, David Comas, Durai Thriupathi, Robert Wilson, Sang-Yong Lee, Hyukjune Chung, Sankar Ananthakrishnan, Doney Joseph, Baltasar Beferull-Lozano, Krisda Lengwehasatit, Panayiotis Georgiou and Jose Gimeno. They made my life at USC a wonderful experience. I would also like to thank Prof. Abeer Alwan and Qifeng Zhu for a fruitful collaboration.

I would like to thank my mother, for letting me pursue my ambitions. I would also like to thank my brother for his support. Finally, I express immense gratitude to Vaishali

Umredkar, whose support and encouragement made it possible for me start, stay the course and complete my Ph.D. Also, I have to mention Ameya, whose innocence helped me maintain a cheerful outlook at all times.

Contents

List Of Tables

List Of Figures

Abstract

With the wide proliferation of mobile devices coupled with the explosion of new multimedia applications, there is a need for adopting a client-server architecture to enable clients with low complexity/memory to support complex multimedia applications. In these client-server systems compression is vital to minimize the communication channel bandwidth requirements by compressing the transmitted data. Traditionally, compression techniques have been designed to minimize perceptual distortion, i.e., the compressed data was intended to be heard/viewed by humans. However, recently there has been an emergence of applications in which the compressed data is processed by an algorithm. Examples include distributed estimation or classification. In these applications, for best system performance, rather than minimizing perceptual distortion, the compression algorithm should be optimized to have the least effect on the estimation/classification capability of the processing algorithm. In this work novel compression techniques *optimized for classification* are proposed.

The first application considered is remote speech recognition, where the speech recognizer uses compressed data to recognize the spoken utterance. For this application, a *scalable encoder* designed to maximize recognition performance is proposed. The scalable encoder is shown to have superior rate-recognition performance compared to conventional speech encoders. Additionally, a scalable recognition system capable of trading off recognition performance for reduced complexity is also proposed. These are useful in *distributed speech recognition* systems where several clients are accessing a single server and efficient server design becomes important to both reduce the computational complexity and the bandwidth requirement at the server.

The second application considered is distributed classification, where the classifier operates on the compressed and transmitted data to make the class decision. A novel algorithm is proposed which is shown to significant reduce the misclassification penalty with a small sacrifice in distortion performance. The generality of this algorithm is demonstrated by extending it to improve the performance of table-lookup encoders. It is shown that by designing product vector quantizers (PVQ) to approximate a higher dimension vector quantizer (VQ), a significant improvement in PSNR performance over conventional PVQ design is possible while not increasing the encoding time significantly over conventional table-lookup encoding.

Finally, a new distortion metric, *mutual information (MI) loss*, is proposed for designing quantizers in distributed classification applications. It is shown that the MI loss optimized quantizers are able to provide significant improvements in classification performance when compared to mean square error optimized quantizers. Empirical quantizer design and rate allocation algorithms are provided to optimize quantizers for minimizing MI loss. Additionally, it is shown that the MI loss metric can be used to design quantizers operating on low dimension vectors. This is a vital requirement in classification systems employing high dimension classifiers as it enables design of optimal and practical minimum MI loss quantizers implementable on low complexity/memory clients.

Chapter 1

Introduction

1.1 Motivation

The past few decades have seen networking expand from small local area networks (LANs) connecting computers in a university or a research lab to form a global network connecting computers across the entire world. Recent advances in wireless technology have enabled this expansion to include mobile devices into this global network. This combination of portable computing with portable communications is changing the way we think about information processing [69]. LANs were originally deployed for data transfer and remote access. However in the past decade there are an increasing number of applications which are targeting delivery of information and multimedia data content. People want to be able to transmit and receive information independent of computing platform, communication device, and communication bandwidth (a concept sometimes called nomadic computing [30]).

This diversity of computing devices and high user expectation has resulted in new signal processing challenges. Multimedia applications (e.g., speech recognition and video streaming) are beginning to enter our everyday life. Although processor speeds are increasing and memory sizes are decreasing, there still is a wide disparity between the computational/memory capabilities of a mobile device and a desktop computer.

While the aforementioned multimedia applications can be supported for users employing desktops, mobile devices are severely challenged to support them. A feasible approach to enable the mobile devices to support these complex multimedia applications is to adopt a distributed processing paradigm. The mobile devices, although unable to support the multimedia applications locally, can draw upon distributed resources hosted on more powerful networked servers to provide multimedia services to the users. In general, transmission tends to require more power than processing. So, while communicating between the mobile device and the remote server it is essential to use compression in order to maximize the battery life of the mobile device and minimize the bandwidth requirements.

Several compression schemes have been developed for multimedia data. Some of these are FS-10 (CELP), GSM, MELP, AMR, EVRC and the ITU G72x series for voice; mp3, AAC for audio; JPEG-2000, JPEG and GIF for image; and MPEG, H.26x for video. These conventional compression schemes have concentrated on efficient representation (and possibly transmission) of the source data. However, these schemes have been developed with the assumption that the final target is a human who either views or hears the encoded data. Thus the primary objective of these compression schemes is to achieve the best *perceptual* quality.

However, new multimedia applications, such as, distributed speech recognition (DSR) [19, 51, 65, 21, 63, 8, 74], content retrieval from an image database [70], or sensor networks [32, 13], fundamentally differ in that the acquired data is not seen/heard by humans but needs to be processed by algorithms [41].

Unlike before, minimizing perceptual distortion may not be the most efficient criteria in these applications. To ensure better system performance it will be beneficial to tailor the compression scheme to the specific application. To motivate this concept consider the communication system shown in Figure 1.1. The client acquires the data x, encodes it and transmits it to a remote server. The server decodes the received data and estimates a parameter θ from the decoded data. Since the main (or only) objective of the server is to estimate θ, the compression scheme need *not* reliably reproduce x at the server (for many applications, e.g. DSR, the server is not required to generate a reproduction of the original data). Instead the compression scheme should be optimized such that the server can reliably reproduce the parameter θ. Of course if the client had sufficient computational power and higher domain knowledge then it would likely be more efficient to estimate θ at the client and transmit it losslessly to the server [24].

In this thesis we consider the cases when the client acquiring the data has limited computational and memory resources. Hence, it will not be possible to implement the complex parameter estimator at the client. This is a reasonable assumption because with increasing proliferation of wireless connectivity, there is a widespread use of wireless devices with limited computational capabilities. These clients while able to support some processing, will in general be unable to implement complex algorithms requiring significant processing and large memory requirements. Also, part of the application specific knowledge will generally only be available at a centralized location (e.g., complete database might be unknown or might be too large to be stored at the client).

For optimality, the estimator at the server should be modified to take into account the fact that the data it is receiving has been compressed. This might be possible in certain applications. However, in general the same estimator will have to work in conjunction with several different compression schemes, and designing a separate estimator for every possible compression scheme will not be either efficient or practical. So, in most of what follows, we assume that the compression scheme works in conjunction with a fixed remote estimator.

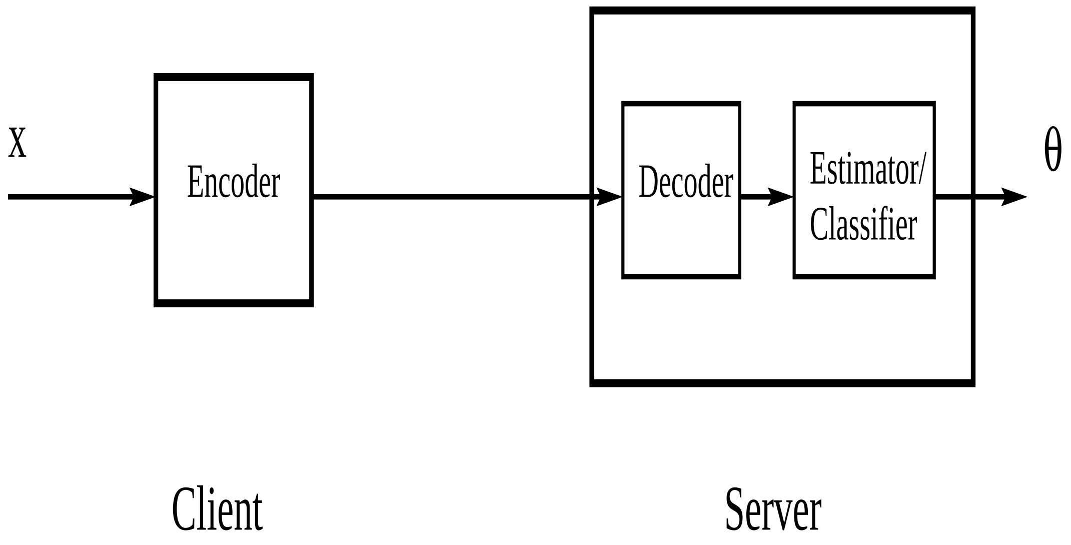

Figure 1.1: Distributed estimation system, the client should encodes x such that the parameter (θ) estimation capability of the server is not degraded.

Consider for example a distributed speech recognition system, where speech is acquired at a (possibly mobile) client (e.g., a PDA or a cell phone) and transmitted to a remote speech recognizer for recognition. While a simple recognizer can be implemented on the client, considering the varying ambient environment at the client and the need to support a large vocabulary continuous speech recognition (LVCSR) it is desirable to implement a complex recognizer capable of handling these requirements. While the client cannot host such a complex recognizer[1], it is possible to implement the recognizer at the server (unlike the client it is assumed that the server is not a mobile device and is hosted on a computer with sufficient computational and memory resources). The straightforward method of communication between the client and the server is to use a standard speech encoder to encode the raw speech. However since the speech will primarily be used for recognition (and not playback, see [12, 49] for methods of reconstructing intelligent speech from features used for recognition), optimizing the compression algorithm to minimize recognition errors rather than perceptual distortion is more meaningful. Also, it is desirable to use a single speech recognizer to handle both compressed (different compression schemes could be used) and uncompressed data (the uncompressed speech could be from land-line telephones). This will enable easy design/maintainance/upgrading of the system. This requirement makes it unattractive to use a speech recognizer optimized for a specific compression scheme.

Similarly when a sensor network is considered, the individual sensors acquire data about a target or environment. To maximize the life of the sensor network (minimize battery power consumption) the sensors do not communicate with each other. Instead the sensors transmit the acquired data to a central node. The central node fuses the received sensor inputs to estimate the required target parameter (e.g., target tracking, target identification, [32] or source localization [13]), or acquire information about the environment (e.g. factory, habitat, etc). The objective of the compression scheme at each

[1]With the current advances in integrated circuits technology it is conceivable that complex recognizers can be implemented in the clients in the future. However advances in automatic speech recognition (ASR) technology for e.g., migration from "directed speech" speech recognition to natural language speech recognition [46] will imply that the computation/memory requirements for speech recognition will also continue to increase.

sensor should be to ensure that compression does not degrade the parameter estimation capability of the central node, rather than to ensure good reconstruction of the sensor outputs at the central node.

Even when computational resources at the client are not constrained, there are applications where distributed processing is preferable. In network based applications it is not practical to distribute the database to every client in cases such as a speech recognizer based call center (e.g., United airlines flight information, E*TRADE telephone investing, Pizza Hut ordering system) or voice portals (Audiopoint, HeyAnita to name a few) where customers dial in to request information. In these examples the database used for information retrieval is constantly changing and it is preferable to maintain this at a centralized location. In addition distribution of the database might not be possible for security reasons, e.g., remote speaker recognition, banking or trading applications.

The main goal of the compression schemes should be to retain the information most relevant for estimation/classification while discarding the redundant information. Unlike conventional compression schemes which aim to minimize distortion at a given bitrate the requirement on the new compression schemes is a trade-off between estimation/classification performance, bitrate and (possibly) complexity. Ideally the compression schemes should be designed such that the performance of the estimator/classifier monotonically increases with bitrate. In addition it is also desirable to have a trade-off between performance and complexity. Extra processing at the client should improve the server performance and/or reduce server complexity. Under heavy load conditions the server should be able to achieve reasonable performance using reduced complexity methods, with these methods possibly requiring fewer bits to be transmitted from the client.

1.2 Distributed Speech Recognition

A block diagram of a speech recognizer is shown in Figure 1.2.

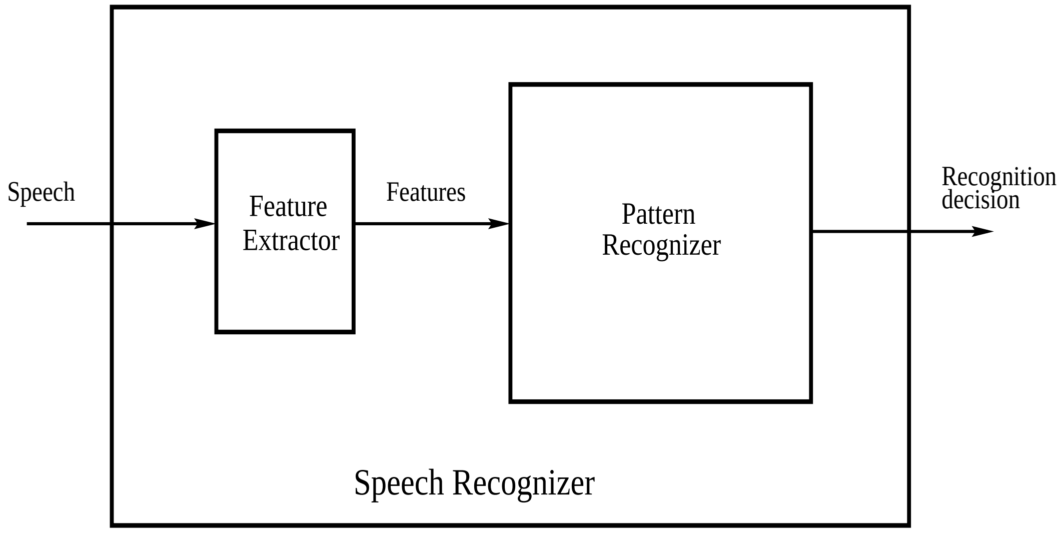

Figure 1.2: A typical speech recognition consists of a feature extractor the output of which is used by a pattern recognizer to make the recognition decision.

It consists of two parts

- A feature extractor (typically, Mel frequency cepstral coefficients (MFCC)) and
- A pattern recognizer (typically, hidden Markov models (HMM)).

Most of the computational complexity of the speech recognizer is in the pattern recognizer, while the low complexity feature extractor can be implemented even on simple mobile devices. By moving the feature extractor onto the client we get a distributed speech recognizer (DSR) (where now the recognition computation is distributed between the

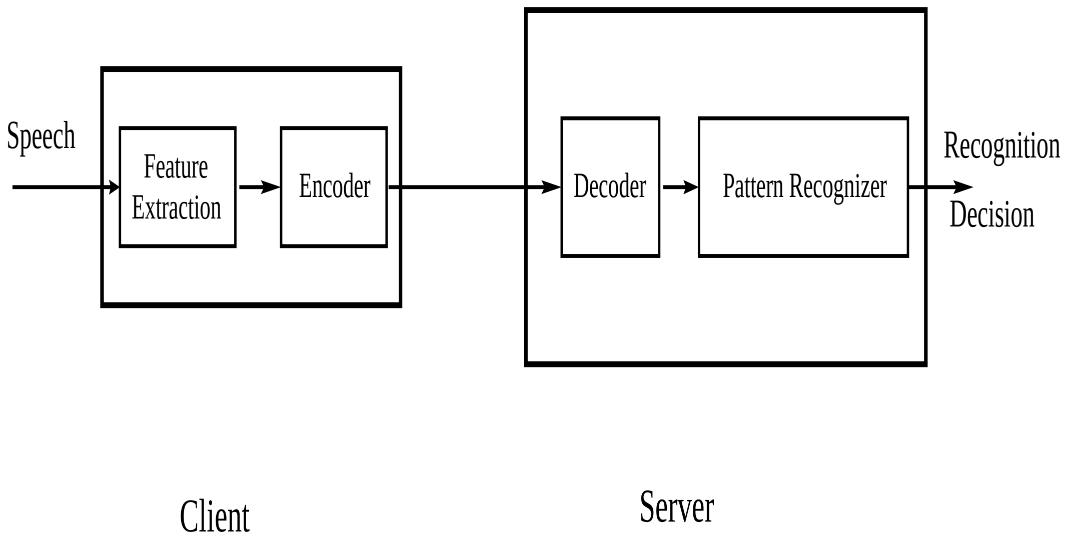

Figure 1.3: A distributed speech recognizer. The client extracts the features encodes and transmits it to the server which decodes it and uses for recognition.

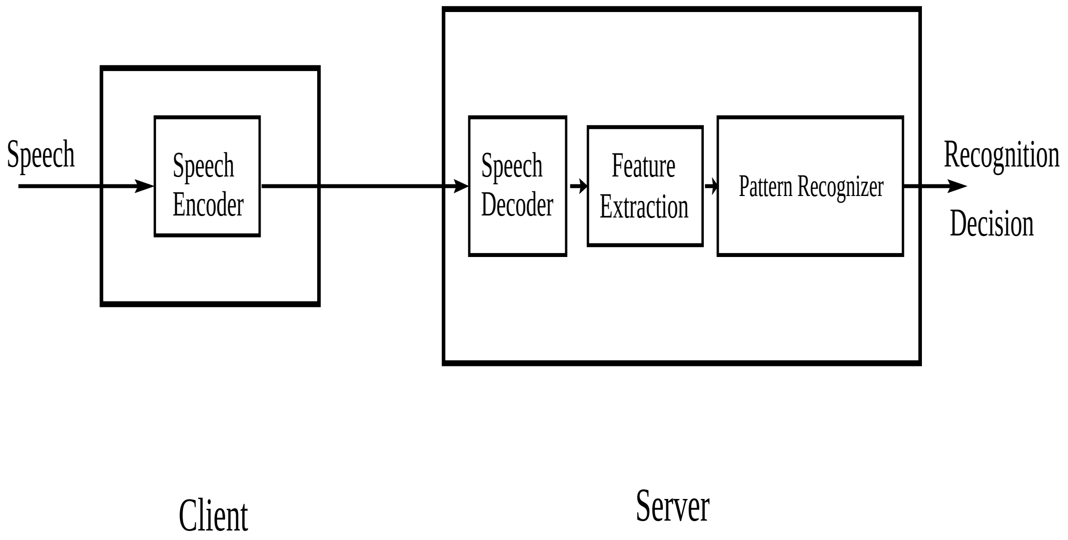

Figure 1.4: A client-server based speech recognition system using a conventional speech encoder to compress the data. Note that unlike a DSR system the feature extraction is done at the server.

client and the server). This is shown in Figure 1.3. There are several potential advantages of using this architecture over the more straightforward method (Figure 1.4) which uses a conventional speech encoder at the client, namely,

- Improved recognition performance
- Robustness to noise
- Scalability (see Section 1.3)
- Distribution of computation between server and client
- Error protected lossless channel for communication

The advantages of using a network based speech recognizer over using a local speech recognizer in the client are

- Complex acoustic models can be used
- Complex language models can be used
- Multi-modal recognition can be implemented
- Easy integration with spoken dialog applications is possible
- Easier maintenance of the system is possible by having a centralized system
- Increased security (database stored at central server)

Compression is vital to minimize the bandwidth requirement between the client and the server. DSR speech encoders operate directly on the features used for recognition. The advantage of operating directly on the features is that only the information relevant to the speech recognizer needs to be encoded, unlike in conventional speech encoders which operate on the entire speech waveform. For example, speech recognition in cars has to overcome noise generated by the vehicle, the radio and (talking by) other occupants of the car. In addition, the recognition system must also deal with channel errors. Part of the difficulty in the conventional (non-DSR) system is that the server has to handle both the environment noise and the channel noise. By using powerful feature level noise compensation techniques [48, 72] at the client we can minimize the effect of environment errors and transmit reliable features to the server which now only will have to deal with the channel impairments. This will ensure that effects of noise are mitigated as soon as they are encountered.

Additionally, operating directly on the features allows the compression techniques to make use of the importance of the features for speech recognition. Furthermore, DSR speech encoders are able to achieve better rate-performance trade-off when compared to conventional speech encoders. Finally, the DSR approach eliminates the need of a speech channel in remote speech recognition and enables the use of an error-protected lossless data channel instead. This results in better robustness to channel errors.

It should be noted that the complexity of the front end processing is comparable to the complexity of a conventional speech encoders, so the computational burden at the client is not adversely affected by moving the front end processing onto the client.

1.3 Scalable Compression for Classification

To motivate the necessity for scalable compression for classification, consider the example of a 2 dimension classifier shown in Figure 1.5. The probability distribution function (*pdf*) of class 1 is $f_1(x_0, x_1) = N(2.0, 1) * N(0.5, 1)$ and the *pdf* of class 2 is $f_2(x_0, x_1) = N(-2.0, 1) * N(-0.5, 1)$. In the figure, the diagonal line represents the optimal Bayes classifier (for equal variance Gaussians, this is the line perpendicular to the line joining the means of the 2 classes) which is the optimal classifier achieving minimum probability of misclassification.

It is apparent that for classification the dimension x_0 has better discrimination power than dimension x_1. For example, if we were to use only x_0, the optimal classification boundary would be the y-axis while using only x_1 implies that the x-axis is the optimal classification boundary. The increase in the probability of error over the optimal Bayes classifier when only one dimension is used is obviously more when we use only x_1 than when we use only x_0.

This unequal discrimination power can be exploited to design a trivial scalable classification system. At first we send only x_0, i.e., the more important (for classification) coefficient and based on the partial classification result at the server we can, if required, request for x_1 from the client. Consider the vector a, we observe that by looking only at a_0 we can say that it is highly likely that a belongs to class 1 and hence there is no necessity for requesting a_1. However, for vector b the class uncertainty is more. Hence we request b_1 in order to make the final classification decision using the entire vector.

This concept of scalability enables us to use a coarse resolution version of the data in making our initial class decision. For vectors that are "easy" to classify the initial decision is highly accurate, however for "difficult" vectors the initial decision can be refined by using more data to get more accurate results.

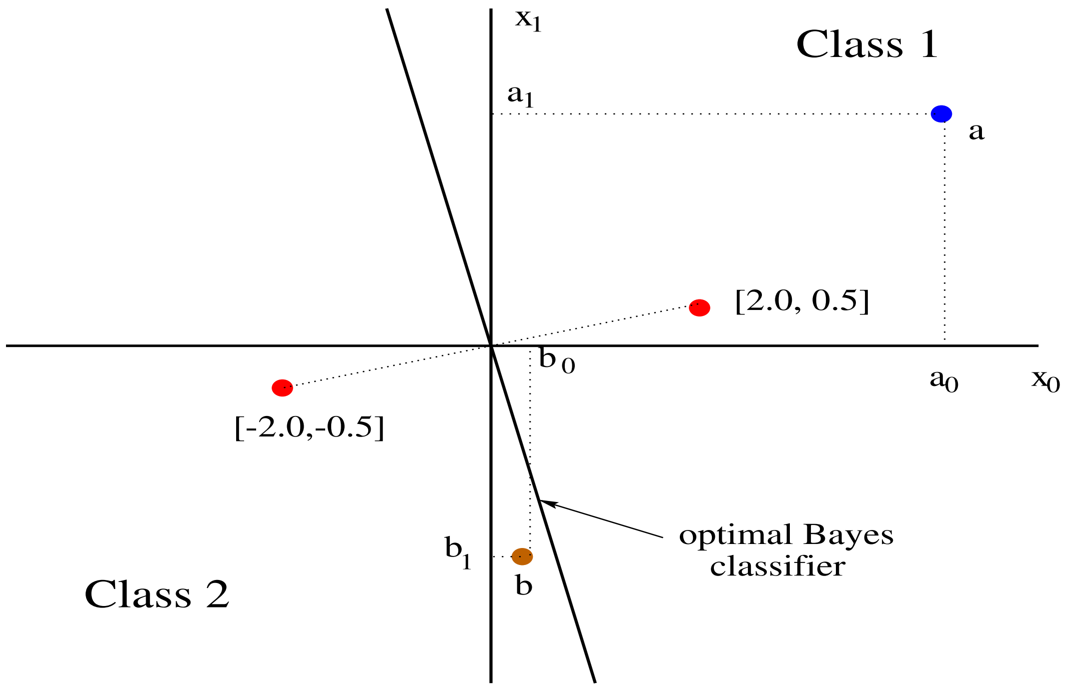

Figure 1.5: The optimal Bayes classifier for a mixture of two sources with means $[2.0, 0.5]$ and $[-2.0, -0.5]$. Notice that x_0 has greater discrimination power than x_1. By using only a_0 we can be almost sure that it belongs to class 1, however for b which lies close to the classification boundary both components are required.

To take full advantage of a scalable classification system, the encoder should be layered [63]. With a non-layered encoder if the current performance is not acceptable a completely new bitstream (at a higher rate) has to be transmitted from the client. However, a layered encoder has the property that to achieve full resolution only incremental bits (lower rate than full resolution) need to be sent and can be combined with the previously received bits to provide a better reconstruction of the data so as to improve the server performance. This can potentially result in a lower average bitrate requirement for a given performance criteria when compared to a non-layered encoder.

1.4 Classifier-Encoder Dimension Mismatch

Typically the classifier at the server either operates on the entire received data or on long source sequences. When the encoder also operates on the entire vector being used for classification it is relatively straightforward to incorporate the effect of encoding on classification [40, 43, 42]. However, the computational complexity of such a high dimension encoder can preclude it from being used by low complexity clients. Consider the case of joint compression and classification. When the encoder and classifier dimensions are the same then the weighted sum of distortion and Bayes risk [40, 43, 42] can be used as the total cost in designing the encoder (Bayes VQ). In these cases, the encoder output corresponds to a reproduction level and a class label, so for some applications the classifier can be replaced by a table lookup. However, since the encoder operates on high dimensional vectors it requires high computation complexity and can become a potential computational bottleneck in the system.

To alleviate the complexity of encoding, in this thesis we consider encoders designed to operate on sub-vectors of the vector. However, the design of these encoders is complicated by the fact that each of the encoders independently operate on the sub-dimensions of the source data while the remote classifier operates on the entire decoded data [61, 62]. For example, in a DSR system the remote speech recognizer uses the quantized features from many frames of the utterance to make its decision. Clearly it is not possible to use an encoder at the client which has the same dimension as the speech recognizer because the number of frames, for say a digit utterance, can be around 50 (typically) and 12 features are extracted for every frame; the resulting encoder operating on a 600 dimension vector is computationally impractical. An alternative solution would be to encode each feature frame individually using a vector quantizer (VQ) of dimension 12 (or use 12 scalar quantizer (SQ), one for each feature), this leads to the scenario where it is required that low dimension encoders have to operate in conjunction with a high dimension classifier.

The crucial difference between designing low dimension encoders to minimize only distortion and designing low dimension encoders to minimize both distortion and misclassification is that: distortion is an additive separable cost and can be optimized separately in each of the sub-dimensions. However, misclassification can be defined only for the entire vector and hence there is no straightforward way of breaking this misclassification cost into elementary components.

The fact that the classifier uses the entire quantized vector for classification, while, each of the encoders independently quantize the sub-dimensions (due to computational constraints), implies that the encoder design in the different sub-dimensions are not independent of each other. To take this into effect we need to jointly design the encoders. Information from other sub-dimensions needs to be used to incorporate the effect of misclassification during the design of each of the individual encoders. This design methodology significantly outperforms an independent encoder design where the encoders are designed separately to only minimize distortion. The low dimension encoders operating on the sub-dimensions of the data can be used to get an approximate classification, which can be used by the classifier to reduce the search space for finding the true class label.

Thus the encoders not only reduce the bitrate but also aid in the classification. To ensure that using a reduced search space will not sacrifice classification performance it is important that the encoders are designed to approximate the higher dimension classifier structure (operation).

1.5 Distortion Metric for Classification

Consider applications where the decoded data at the remote server is used only for classification. In these applications, ideally the quantizer design should be optimized to ensure that quantization does not increase the probability of misclassification as compared to when unquantized data was used for classification. However, probability of misclassification does not have a mathematically tractable form to enable its use in the quantizer design. However, it is not apparent that MSE based quantizer design is the optimal design for classification applications. Hence, there is a necessity to investigate alternative distortion metrics that are more meaningful than MSE for classification applications.

The goal of classification is to estimate a class label from the input data. Thus it seems natural to design encoders which preserve this class information, i.e., design quantizers that aim to preserve the class information present in the unquantized data during the quantization operation. This would ensure that on average the class label estimated from the quantized data will be closer to that estimated from unquantized data.

The main difficulties in designing classification optimal quantizers are

- determining relevant distortion metrics, which succinctly encapsulate the effect of quantization on the class label information

- designing optimal quantizer algorithms for the new distortion metrics
- providing empirical design algorithms which use training data during design, rather than being limited to (generally impractical) situations which require knowledge of probability distributions

In order to address the aforementioned difficulties, in this thesis we consider the information-theoretic measure of mutual information. Mutual information captures information contained in the data about the class labels. Hence, we consider designing a quantizer that can ensure that the loss in mutual information between the data and the class labels is minimized during quantization. It can be expected that this quantizer will perform better than MSE optimized quantizers in classification applications.

The additional advantage of the mutual information metric is that it is also defined for sub-vectors of the entire vector used for classification. Hence design of low dimension encoders, each of which operate on a sub-vector of the vector used for classification, is possible. Therefore, not only is the mutual information metric better suited for classification applications it can also be applied to design low complexity encoders, which operate on low dimension vectors. Finally, it is also possible to provide empirical quantizer design algorithms using labeled training data.

1.6 Outline and contributions of this proposal

The main contributions of this research are

- *Novel speech encoder design for distributed speech recognition applications.* An alternative speech encoding scheme is proposed which instead of minimizing perceptual distortion attempts to reduce speech recognition errors. It is shown that the proposed encoding scheme is able to achieve better rate-recognition performance than conventional speech encoders.

- *Scalable speech encoding and Scalable speech recognition.* The amount of data required for recognition should be dependent on the speech utterance. It is shown that "easy" to recognize utterances use less data, while "hard" to recognize utterances use more data. By using such a data dependent approach the scalable DSR system ensures that while the recognition performance is not adversely affected the bitrate required for data representation is reduced.

- *Joint compression and classification with encoder-classifier dimension mismatch.* A novel algorithm is proposed to design low dimension encoders, the output of which is used for classification by a high dimension classifier. The joint encoder design ensures that independent encoding in each sub-dimension has the least effect on classification performance.

- *Performance improvements in fast table-lookup encoding for image compression applications.* The joint compression and classification algorithm is extended to propose a refine-able fast encoding technique which improves the PSNR performance of table lookup encoders while keeping the encoding time significantly lower than full search VQ algorithms (VQ_{fs}).

- *Sub-dimension based distortion metric for classification applications.* The *information-theoretic* measure of *mutual information* is adopted to propose a sub-dimension based distortion metric. This metric while being defined for sub-vectors of the vector used for classification, succinctly captures the effect quantization has on classification. Both quantizer design and rate-allocation algorithms are proposed which minimize the mutual information loss incurred due to encoding. It is shown that use of this metric provides rate-recognition performance improvements over encoders designed to minimize mean square error.

The speech encoding technique for distributed speech recognition is proposed in Chapter 2. Modifications required to make this encoding technique layered are also presented. A data dependent scalable speech recognizer is introduced which can be combined with the layered speech encoder to achieve improved rate-recognition performance and provides the ability to trade recognition performance for lower computational complexity. Adaptation techniques to improve the speech recognition performance by treating the encoder variability in DSR systems as a train-test mismatch are presented. In Chapter 3 the joint encoder design for low dimension encoders working in conjunction with a high dimension classifier is proposed. Both a parametric design, which can be used when the source *pdf* is known, and an empirical design are proposed. In Chapter 4 the joint encoder design proposed in Chapter 3 is extended to improve the PSNR performance of table lookup VQ encoders. It is shown that by designing product VQs to approximate a higher dimension structure, encoding complexity can be reduced significantly without significantly sacrificing the PSNR performance when compared to full search VQ algorithms. A sub-dimension based distortion (mutual information loss) metric applicable for encoder design in classification applications is presented in Chapter 5. Empirical algorithms are provided for both quantizer design and rate-allocation. The rate allocation algorithm is applied to a distributed speech recognition task to demonstrate the improvements possible with the proposed technique. Finally in Chapter 6 extensions and future work are described.

Chapter 2

Distributed Speech Recognition: Speech Encoding and Scalability

2.1 Introduction

Speech recognition is important to enable mobile devices to support speech driven applications While a simple recognizer with a small grammar can be implemented on typical mobile devices, e.g., mobile phones, PDAs, large vocabulary continuous speech recognition (LVCSR), with vocabulary size $> 10,000$ words, requiring bigram or trigram language models are generally too expensive computationally to be performed at the mobile devices. One method to overcome this computational bottleneck is to adopt a client-server architecture where the mobile device (client) makes use of network resources to support speech driven applications. We consider here the distributed speech recognition (DSR) system shown in Figure 1.3, where speech is acquired at the client and then transmitted to a remote recognition engine. Other examples where a client server architecture is useful include distributed web applications where a single centralized recognition database is kept for security or scalability reasons [4].

Given the reduced channel bit rate available in typical applications (especially mobile ones) compression will be required to transmit speech to the remote recognizer. The choice of encoding algorithms influences both (i) recognition performance and (ii) system operation. It also necessitates the speech recognition system to take into account the encoder variability. These are discussed in detail below.

2.1.1 Recognition Performance

The goal of the speech encoder used in a DSR system should be to have the least effect on recognition performance for a given bitrate. Current speech coding techniques focus on preserving the perceptual quality of the speech. Their design is motivated by the fact that the decoded speech should sound as “similar” as possible to the original speech. It has been observed that speech encoded with conventional speech encoders significantly degrades the recognition performance [19, 65]. Thus when the ultimate (or primary) objective is recognition, rather than playback, it is desirable for the speech encoder to minimize recognition errors rather than perceptual distortion.

In our approach we assume that Hidden Markov Model (HMM) [47] based recognizers are being used with a Mel Frequency Cepstral Coefficient (MFCC) [18] front end. As in previous work [19, 51] we assume the client extracts the MFCCs and then compresses the coefficients to transmit them to the recognition engine. We present a complete compression algorithm based on scalar quantization, linear prediction [51], entropy coding and coefficient pruning. Unlike previous work [19, 51], our coding algorithm is *scalable*, that is, it is possible to reduce the coding rate (for example if the channel conditions worsen and additional channel coding is needed), at the cost of some decrease in the recognition performance. Scalability is achieved by changing the quantization step size to reduce the number of coefficients transmitted. In addition to scalability, the proposed technique achieves similar recognition performance at lower rates, and with significantly lower complexity, than previously proposed approaches.

2.1.2 System Operation

With widespread deployment of DSR we can expect that servers will have to serve significant number of clients. As the number of clients increases, designing efficient servers becomes more important since increasing the number of clients that can be supported by a single server will contribute to lowering the overall cost. Supporting large number of clients not only places computational demands on the server but can also significantly increase the network traffic that the server has to handle. A novel method to tackle both these problems is by using scalable recognizers along with scalable encoding schemes. This scheme is shown in Figure 2.1. The initial low complexity recognizer operating on coarse data can provide the likelihood of each of the possible outcomes. This information can be used to constrain the search in the high complexity recognizer to only the most "likely" outcomes and use the high resolution data to provide the final recognition result. Often the low complexity recognizer itself can make the final decision, i.e., the number of potential classes is only one, implying that the high complexity recognizer need not be used and the server will not request the client for the enhancement data (effectively reducing the network bandwidth at the server).

Multi-pass speech recognizers have been previously proposed [38]. These schemes iteratively use more and more complex recognition schemes (more complex acoustic models,

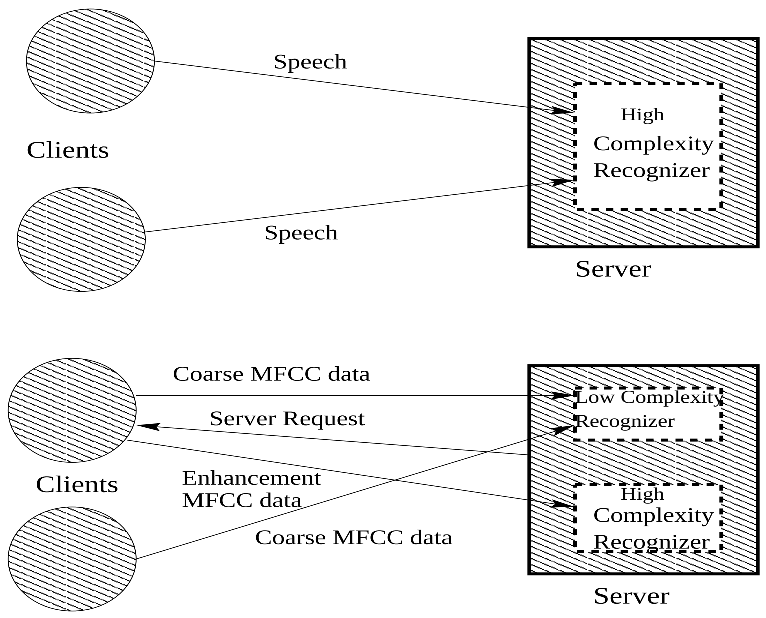

Figure 2.1: Scalable recognition system. The top figure shows the conventional DSR system. The number of clients that the server can handle is limited by the server computational capabilities and the server bandwidth. The bottom figure shows the proposed scalable system wherein the computational requirements as well as the bandwidth at the server are reduced. In the scalable system the enhancement data is requested by the server only if the low complexity recognizer cannot make the recognition decision.

language models, increased knowledge) to refine the word lattices produced by the previous stage. However the successive recognition stages all operate on the same data. In our scheme while the successive stages refine the lattice produced by the previous stage, the successive iterations operate on data at different resolutions, i.e., with each additional iteration more data is requested from the client and successive stages operate on higher resolution data. This multi-resolution representation of the data by the encoder can potentially give insights into the amount of information required to achieve a given recognition performance.

To take full advantage of the recognizer scalability we require a scalable-layered encoder, which will enable reduced bandwidth requirements at the server. Thus the scalable system will have the flexibility of trading off complexity, bitrate and recognition performance. If we were to use a non-layered encoder the bitrate required for the fine layer will be high (since the fine layer encoder does not make use of the fact that the coarse layer has already been transmitted). To reduce the bitrate required for the fine resolution we require a layered encoding scheme where additional refining bits are combined with the already received coarse layer to derive the fine resolution data. Previous work on scalability in predictive coding [55] has addressed the issue of designing a layered coder for video. However this approach requires knowledge (or modeling) of the prediction error *pdf*. We modify our proposed MFCC encoder and propose a novel scalable encoder motivated by the multiple description encoder design (using a fine and coarse DPCM loop) proposed in [59]. Our proposed scheme can easily be combined with the predictive coding design proposed in [55] if the prediction error *pdf* is modeled. In the proposed scalable-layered encoder the fine layer encoder uses the information present in the coarse layer and only transmits additional bits required to achieve the desired quality.

2.1.3 Encoder Variability

The European Telecommunications Standards Institute (ETSI) has standardized the front-end and the encoder for use in DSR systems (AURORA [21]). However, given that processor speeds are increasing constantly and memory size and cost are decreasing, it is to be expected that other encoders will be proposed and used in DSR systems. In addition there are several different compression algorithms that have been developed for traditional speech encoding. These include the traditional waveform based coders: PCM, and AD-PCM; and the linear prediction based coders CELP, MELP, G.723, G.728, G.729, GSM, EVRC and AMR. Scalable encoders further introduce more encoder variability. The choice of speech encoder used by the client may be made dynamically depending on the computational resources/load at the client and the quality of service (QoS) it wishes to provide the user. Hence it is highly unlikely that a single encoder will be used for all DSR applications. The encoding operation can be viewed as a train-test mismatch, i.e., from the speech recognizers perspective the encoder has corrupted the acquired speech waveform. This will obviously result in a degradation of the speech recognizer performance (assuming that the models at the recognizer were trained with uncompressed speech). This mismatch can be compensated by using robust adaptation techniques such as Maximum Likelihood Linear Regression (MLLR) [73] or Bayesian adaptive techniques [25, 31] by modifying the reference models using the observed (compressed) MFCCs as shown in Figure 2.2.

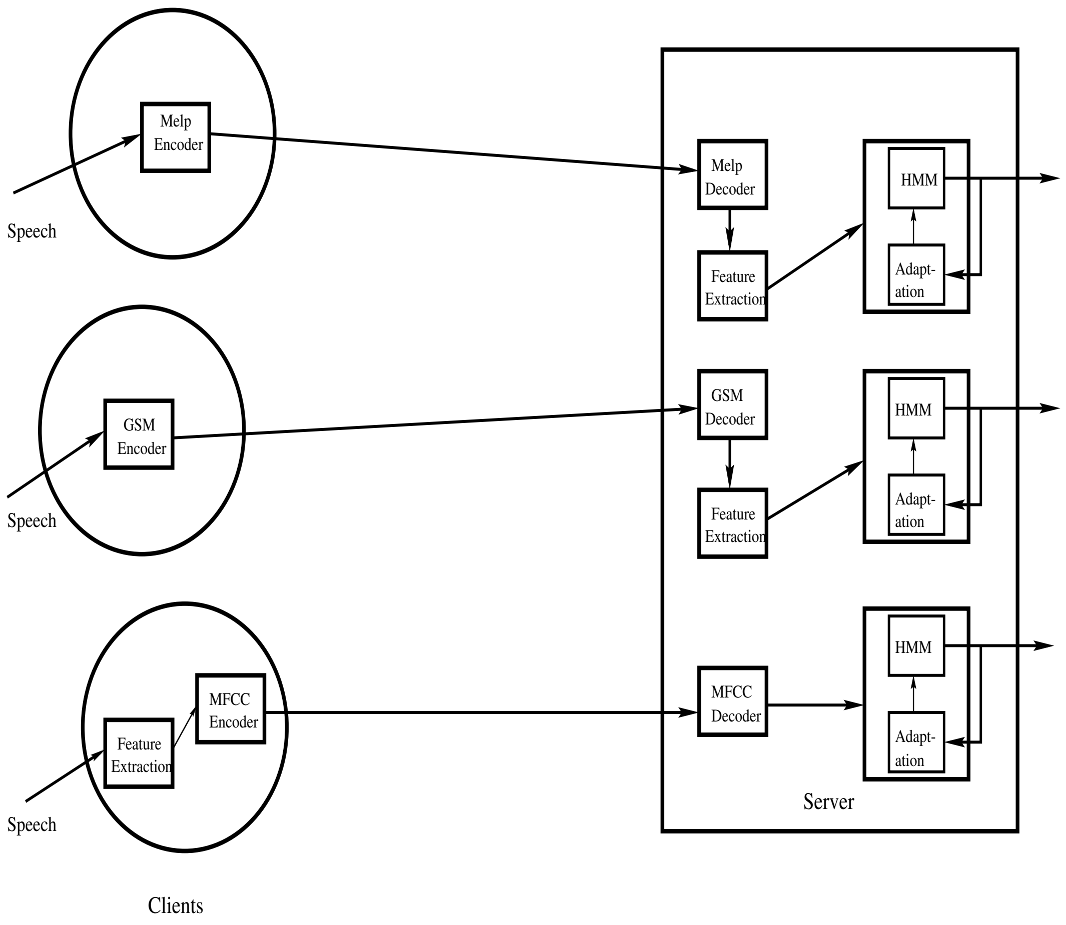

Figure 2.2: A remote speech recognition system. Several clients communicate with a single server. The clients could use different compression schemes to encode the speech data. The models at the server are adapted to ensure that the adapted models are more likely to have produced the observed data.

The rest of chapter is organized as follows. We begin by explaining our proposed speech coder for DSR in Section 2.2. The modifications required to convert this into a layered speech coder are presented in Section 2.3. The details of the scalable DSR system are presented in Section 2.4. Compensation of encoder variability by using model transformation is explained in 2.5. Section 2.6 provides details of our experiments for the non-layered and the layered encoding scheme and the experiments performed to compensate for encoder variability. Results of the different schemes are presented in Section 2.7 and finally conclusions and discussion is in Section 2.8.

2.2 Compression of MFCCs

The encoding algorithms presented in this section were developed assuming that the feature vectors used by the speech recognizer are 12 MFCCs derived from every frame of the speech utterance. However, these results can be easily extended to cases where the number of MFCCs in a frame is not 12, or when derivatives are used along with the MFCCs.

MFCCs computed from speech are represented by floating point numbers (32 bits precision is typical). Clearly, it is to be expected that reductions in the precision through quantization will be possible without affecting the recognition performance. In addition, the MFCCs are derived from speech utterances that have been segmented using overlapping Hamming windows. Due to this overlap it is reasonable to expect that MFCC sets corresponding to adjacent frames will exhibit high correlation. We exploit this correlation by using linear prediction, where a given MFCC in a frame is predicted from the corresponding MFCC in one or more past frames. Single-step prediction seems a reasonable choice given that the time overlap occurs only between adjacent frames, and indeed our experiments showed that the gain in applying multi-stage prediction was limited. Thus, in what follows we consider only single step linear prediction, where each MFCC is predicted only from the corresponding MFCC in the previous frame. Note that while quantization of MFCCs for speech recognition has been previously considered in [19] and [51], our approach provides better performance at similar or lower complexity. For example, [19] uses scalar quantization (uniform and non-uniform) and product vector quantization (VQ) techniques but does not use either entropy coding or linear prediction. In [51] a one step linear prediction is used along with a 2-Stage VQ that achieves a fixed rate of 4 kbps. This approach lacks scalability and uses an encoder that is significantly more complex than the approaches we present here. Finally, neither [19] nor [51] employed entropy coding, which, as will be shown, can improve further the compression gains at low bitrates.

To quantize the MFCCs (or the prediction errors after linear prediction) we use two different scalar quantization techniques, namely, entropy constrained scalar quantization (ECSQ) [14] and uniform scalar quantization (USQ). In the ECSQ approach the quantizer is designed by minimizing, for each input in the training set, a cost function of the form $\mathbf{C} = \mathbf{D} + \lambda * \mathbf{R}$ where $\mathbf{D}$ and $\mathbf{R}$ are the distortion and rate, respectively, and λ is a Lagrange multiplier, which is a non-negative real number that is used to control the rate-distortion trade-off. Given that the statistics are different we designed different quantizers for each MFCC. Moreover, different quantizers were used depending on whether the coefficient was coded directly, or the error after linear prediction was coded instead. As a simpler alternative, we used USQ to quantize the prediction errors after linear prediction. The same quantization step size can be used for all the coefficients if the prediction errors are divided first by their corresponding standard deviation, which can be computed during training.

One of our main goals in designing this system was to introduce scalability and thus enable a trade-off between recognition performance and bit-rate. A simple approach to achieve scalability (as shown in Chapter 1, Section 1.3) is to transmit only a subset of the 12 MFCCs. Thus some coefficients are "dropped" at the encoder, and set to zero at the decoder so that the recognition algorithm need not be modified, even if the number of coefficients transmitted varies over time. The relative importance of each MFCC can be determined experimentally by observing the degradation in recognition performance over a large training set when each coefficient is dropped. The coefficient that results in the largest drop in recognition performance when being set to zero is thus deemed the most important coefficient. For the TI46-Word digit database and the TI46-Word alphabet database, the coefficients were ordered, from most important to least important, as 2,3,0,1,6,7,4,8,5,10,11,9 (i.e., 9 would be dropped first, then 11 and so on) and (0,3,2,1,5,4,7,6,9,8,11,10), respectively. This approach is denoted "ad hoc pruning" in our experiments.

Since the speech recognizer has been trained with a full set of coefficients, ad hoc pruning may result in significant loss in recognition, especially when many coefficients are dropped for each frame. In addition, improving the performance of a pruning technique may require selecting a different subset of coefficients to be pruned for each frame, which would then require overhead information to be sent to the decoder. As a simple alternative we use USQ with a dead zone (mid-thread quantizer) to quantize the prediction errors for all coefficients. Scalability is possible by changing the quantization step size: coarser step size results in lower rate and vice versa, while in the ECSQ case a completely different quantizer would have to be designed for each rate. Moreover, whenever a prediction error is quantized to zero we use the predicted value for that coefficient (rather than setting the coefficient to zero), which tends to affect less the recognition performance than pruning the coefficient altogether. Thus, USQ offers the advantages of low complexity encoding, simple design (no training is required) and easy scalability.

It should be noted that different frames can contain different number of non zero coefficients based on the values of the prediction errors. Since the prediction errors are scaled by the pre-computed standard deviations and the higher MFCC prediction errors have larger standard deviations, it is more likely that the prediction errors for the higher MFCCs be quantized to zero. This is desirable because, as mentioned above, the lower MFCCs tend to be more important for recognition. The USQ indices are encoded losslessly with a Huffman entropy coder. Thus the lowest bitrate achievable is 1 kbps (the minimum is 1bit/coefficient, which corresponds to 12bits/frame, with one MFCC frame being computed every 12 ms). To achieve lower bitrates, a bitmap is transmitted to the decoder to indicate the position of the non-zero coefficients in every frame, and the non-zero coefficients can be entropy coded by the Huffman coder. The bitmap is efficiently encoded using run length coding.

2.3 Scalable Predictive Encoding

In the previous section it was shown that by using one step linear prediction and uniform quantization of the MFCCs good recognition performance can be achieved. The coding algorithm was able to trade-off recognition performance for reduction in rate. However, it had the disadvantage of not supporting incremental refinement in recognition performance, i.e., the recognition achieved at a low rate could not be improved by using refinement bits. Instead to improve the recognition the encoded bitstream corresponding to the finer resolution had to be received. To overcome this drawback we propose a layered scheme which uses a coarse and a fine DPCM loop. The output of the coarse DPCM loop corresponds to the base layer and the output of the fine DPCM loop corresponds to the refinement layer. In the most straightforward approach the enhancement layer can be constructed by encoding the quantization error introduced by the coarse DPCM loop, as shown in Figure 2.3. However when this approach is used the bitrate required for the enhancement layer was comparable to that required for an independent fine DPCM loop. The alternative approach which we adopted was to maintain both the coarse and fine DPCM loops and use information from the coarse loop to enable better compression of the fine loop prediction error, as shown in Figure 2.4. We propose a novel scheme to encode the fine layer DPCM prediction error based on the consistency criteria proposed in [59]. In the next section we explain the use of consistency criteria to enable reduction of the bitrate required for the fine DPCM loop. In the subsequent section we describe context dependent entropy coding which can be used to further reduce the bitrate.

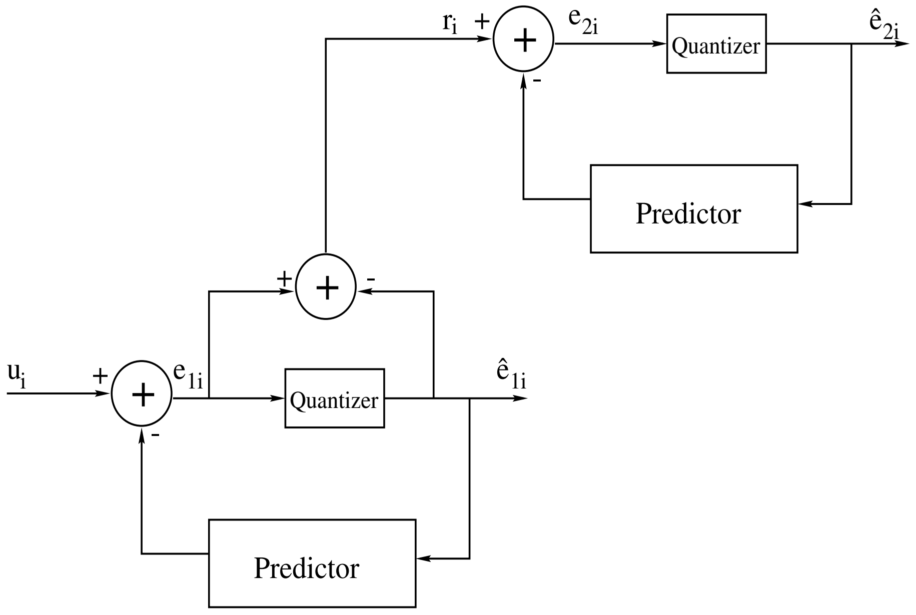

Figure 2.3: A layered DPCM scheme where the top DPCM loops encodes the quantization error of the bottom DPCM loop.

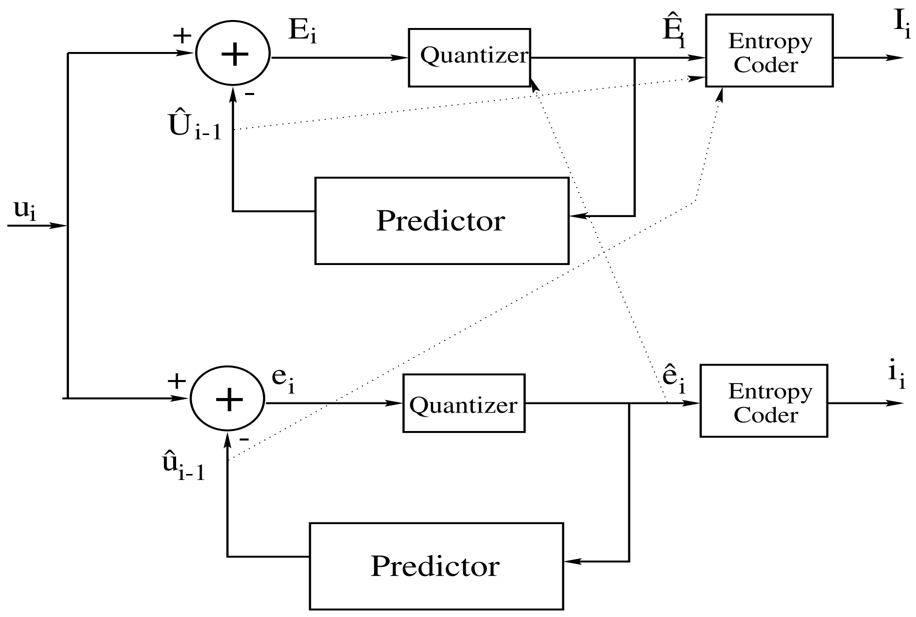

Figure 2.4: The proposed layered DPCM scheme, where two independent DPCM loops are maintained. The quantized prediction error is used by the fine DPCM quantizer to find the valid bins. The entropy coder uses the predictions from both loops to determine the context.

2.3.1 Consistency Criteria for Independent DPCM Loops

For input sample u_i let e_i and E_i be the prediction errors of the coarse and fine loop DPCMs (refer to Figure 2.4). Then

$$e_i = u_i - \alpha\hat{u}_{i-1}, \quad E_i = u_i - \alpha\hat{U}_{i-1}$$

$$\Rightarrow E_i = e_i + \alpha(\hat{u}_{i-1} - \hat{U}_{i-1}) \tag{2.1}$$

where $\hat{u}_{i-1}$ and $\hat{U}_{i-1}$ are the reconstructed samples of the coarse and fine loop DPCMs (as shown in Figure 2.4) and α is the prediction weight. Let $z_i = (\hat{u}_{i-1} - \hat{U}_{i-1})$, and given that $e_i \in [a_k, b_k]$, the interval R_c in which E_i has to lie can be found as

$$E_i \in R_c = [a_k + \alpha z_i, b_k + \alpha z_i] \tag{2.2}$$

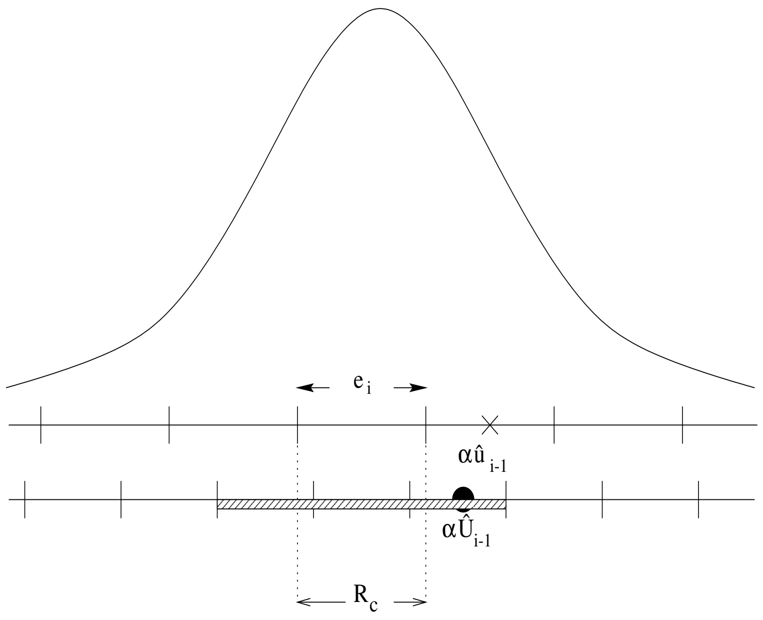

Figure 2.5: Overlapping shifted quantizers. Using information from the coarse reproduction the number of potential bins for Q_f is reduced. Only the bins of Q_f which overlap the region R_c are valid. The valid bins of Q_f are highlighted.

Let Q_f be the fine loop DPCM quantizer with N levels, then only the bins of Q_f that intersect R_c are valid choices for the fine DPCM prediction error. This is illustrated in Figure 2.5 where the region R_c intersects 3 bins (highlighted) of Q_f. Hence, by using knowledge of the interval R_c we can restrict the allowed values in Q_f, and thus enable savings in bitrate. If Δ_c and Δ_f are the step sizes used in the coarse and fine DPCM loop quantizer, then the number of valid bins M is at most $\lceil \frac{\Delta_c}{\Delta_f} \rceil + 1$. If $M << N$ then significant savings in bitrate can be achieved.

2.3.2 Context Dependent Entropy Coding

Let j_i and J_i respectively, be the quantization index of the coarse and fine DPCM loops at time i. Context information from previous coarse and fine reproductions can be used to reduce the number of bits required to encode the current fine DPCM prediction error. We have already used $\hat{U}_{i-1}$ and $\hat{u}_i$ to identify the bins of Q_f that are consistent with the information available (i.e., $\hat{u}_i$, $\hat{u}$ and $\hat{U}_{i-1}$). bins of Q_f. In addition, we can use the previous reconstructed value of the coarse DPCM loop $\hat{u}_{i-1}$ to bias the probability of occurrences for the different bins J_i of the fine DPCM quantizer. While $\hat{u}_{i-1}$ by itself does not provide explicit information, the difference $|\hat{u}_{i-1} - \hat{U}_{i-1}|$ provides information about bins J_i of the fine DPCM quantizer. Consider the case when $|\hat{u}_{i-1} - \hat{U}_{i-1}|$ is small if $j_i = 0$, then it is highly likely that $J_i = 0$ and if $j_i \neq 0$, then it is highly likely that $J_i \neq 0$. On the contrary when $|\hat{u}_{i-1} - \hat{U}_{i-1}|$ is large, then it is highly likely that $J_i \neq 0$. Using this information we can define two different contexts for J_i

C1) $|\hat{u}_{i-1} - \hat{U}_{i-1}| \leq T_q$ and $j_i = 0 \Rightarrow p(J_i = 0) >> p(J_i \neq 0)$

C2) $|\hat{u}_{i-1} - \hat{U}_{i-1}| > T_q$ or $j_i \neq 0 \Rightarrow p(J_i \neq 0) >> p(J_i = 0)$

This information can be exploited by using a bitmap to indicate the more probable event in each context, i.e., in C1 we transmit a "0" if $J_i = 0$ and a "1" otherwise and in C2 we transmit a "0" if $J_i \neq 0$ and a "1" otherwise i.e., a "0" is always transmitted in the bitmap for the more probable event in both contexts. This will ensure that $p(0) >> p(1)$ in the bitmap. An binary arithmetic coder can be used to encode this bitmap efficiently. However the encoding complexity of arithmetic coder is high, so instead we use run length coding to encode the bitmap. This bitmap, can be used by the decoder to find the positions where the prediction error is zero. So now the encoder only has to transmit the non-zero coefficients (in addition to the bitmap). Using the information from the coarse DPCM quantizer to find the valid bins, and the context information from $\hat{u}_{i-1}$ and $\hat{U}_{i-1}$, we were able to reduce the bitrate for the enhancement layer compared to encoding the enhancement layer independently. Specifically, the consistency criteria enabled about 26% reduction in bitrate and the context based entropy coding resulted in an additional 9.5% reduction.

2.3.3 Packetization

In order to minimize the user latency (especially for continuous speech recognition), speech utterances are accumulated for short durations (e.g., one or two seconds). MFCCs are calculated and compressed for these accumulated speech segments; then they are packetized and transmitted. Transmission over error-prone transport channels will typically result in some packets being lost. To mitigate the effect of these losses, frame concealment techniques, such as insertion, interpolation and regeneration [36, 35] and unequal forward error correction techniques [68, 54] can be used. Now it is required that each packet be independently decodable. Hence, the prediction should be limited to only intra-packet MFCCs (i.e., the first MFCC vector of each packet is not predicted from the last MFCC vector of the previous packet). This will obviously result in some loss in compression efficiency. However, assuming each packet corresponds to two seconds of speech data, this implies the prediction loop is broken (to insert an independently coded frame) once every 80^{th} MFCC. This will lead to a very small loss in compression efficiency.

2.4 Scalable DSR System

In this section we present our proposed scalable DSR system, consisting of a layered encoder (Section 2.3) providing a coarse base layer and an enhancement layer. The base layer is used by the first recognizer to provide an initial "guess" of the final class. In general, the complexity of the first recognizer can be reduced by decreasing the complexity of the acoustic models and/or language models and/or pattern recognition schemes. The final recognizer makes use of the initial decision (e.g., hypotheses, word lattices) and the enhancement bits to provide the final recognition decision.

2.4.1 Scalable Recognition

Speech recognition by HMMs has increasing become popular since they provide good recognition results. However, with complex acoustic and language models, they can become a computational bottleneck at the server, when the server is accessed by many clients. The idea here is to consider scalable recognition where we can trade-off complexity versus accuracy. We consider three different recognition experiments to demonstrate the advantages of the scalable distributed recognition system (i) an isolated digits task where a DTW module is used as a pre-processor for an HMM based recognizer, (ii) a *large* spoken names recognition task where a free phone loop HMM recognizer is used as the first stage and a lattice based HMM recognizer is used in the second stage, and (iii) a continuous speech recognizer (CSR) for the HUB-4 broadcast news task, where a bigram language model recognizer was used as the initial state and a lattice based trigram recognizer as the second state. In all the DSR systems, the client employs a scalable encoder to compress the MFCC features.

2.4.2 DTW-HMM Recognizer example for Isolated Word Recognition

Consider the simple task of isolated digit recognition. Every unknown utterance needs to be scored with 10 models before deciding the best match. One method to reduce the computation would be to speed up the HMM (for example using small model sizes). Another method, which we adopt in this thesis, is to build scalable recognizers to restrict the number of the models the recognizer has to operate upon at any time. Usually when an unknown utterance is scored against the different models, only a few of the model scores will be high; this fact can be used to eliminate some of the models from consideration. A low complexity pre-processor can be used to find the N most likely models and the HMM recognizer can be used to choose the best model from these N models. In our example system we choose to use a DTW recognizer as the pre-processor. Since the DTW finds the distance of the unknown utterance from the known templates and the distance is usually minimum between utterances of the same class, we can use a distance threshold to find the N most likely models. An adaptive threshold is used for every utterance based on the lowest distance obtained after template matching. Using adaptive thresholds instead of fixed ones has the desirable feature of selecting more models when the distance (likelihood) between the best and other models is close, and selecting a few (sometimes only one) models when the distance between the best and other models is far. The procedure for recognizing an unknown utterance using the above system is

Algorithm 1 *(Scalable Recognizer : System A)*

Step 1 : *Find the distance $D(k)$ between the unknown utterance and the L templates using DTW.*

Step 2 : *Select the models with distance $D(k)/D(0) < T$.*

Step 3 : *Use HMM to find the best model among the chosen models in* **Step 2**.

This system is shown on the left in Figure 2.7.

Figure 2.6 shows the average number of models retained after the initial DTW stage and the probability of word error in the N-best list of the DTW as a function of the threshold T. The average number of models monotonically increases with threshold and the probability of word error monotonically decreases with threshold. This fact can be used to trade-off complexity and recognition-performance. A low threshold would imply that the WER would be high but the complexity would be low and vice-versa. We observe that with a threshold of 2.4 we get no word error, however the average number of models is reduced only from 10 to 8 by the use of the initial DTW stage.

To further reduce the complexity we can use 2 stages of DTW before using the HMM recognizer. The initial DTW stage uses the MFC coefficients deemed more important to generate an N_1 best list. The second DTW operates on these N_1 models and the distance

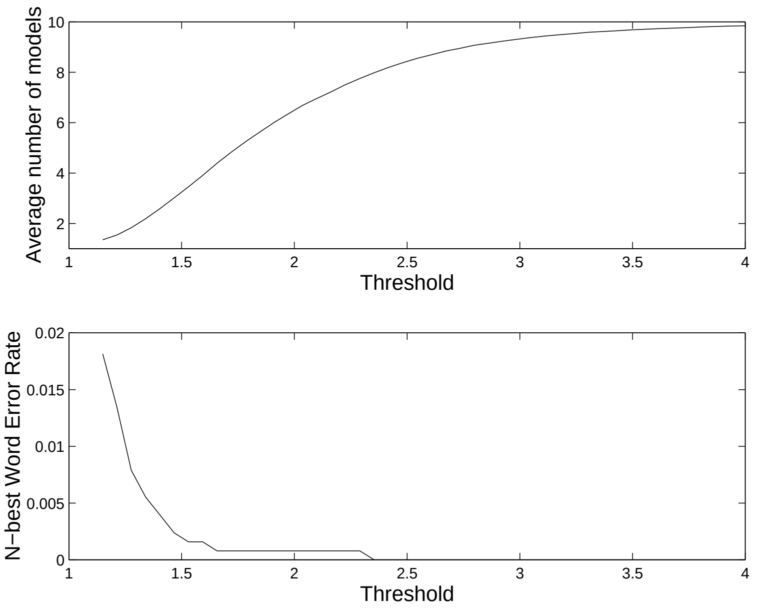

Figure 2.6: Observe that the average number of models increases with threshold and the probability of word error monotonically decreases with threshold. At threshold of 2.4 the probability of word error becomes zero but the average number of models is reduced from 10 to 8, i.e, a 20% reduction in HMM running time with no difference in recognition performance. In general it is not required to use a threshold as high as 2.4, a smaller value between 1.2 to 1.8 will suffice, because utterances very difficult to distinguish by DTW are most likely to be in error for the HMM also.

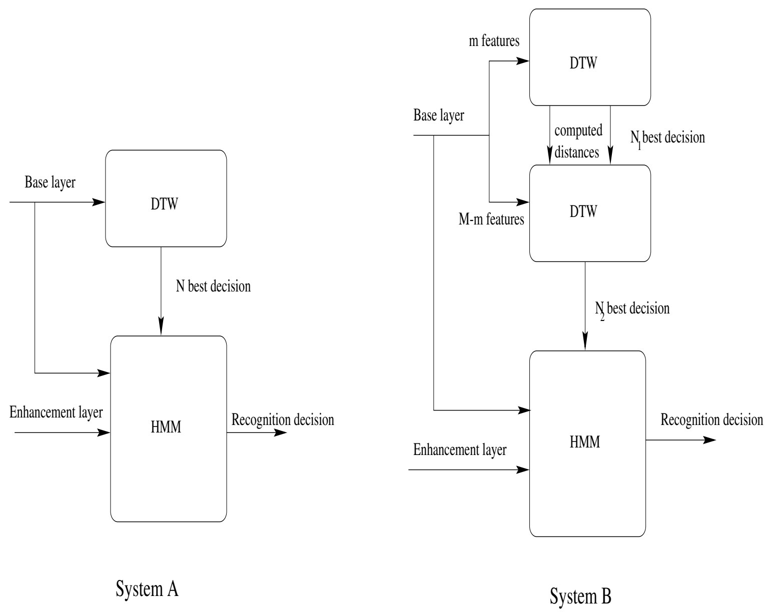

Figure 2.7: The different scalable recognizer schemes used. In System A we use a DTW as the first recognizer and a HMM as the final level recognizer. In System B we use two levels of DTW as the first recognizer and a HMM as the second recognizer.

is refined with the remaining MFCCs to generate an N_2 best list. The HMM makes its decision from these N_2 models. This procedure is summarized below.

Algorithm 2 *(Scalable Recognizer : System B)*

Step 1 : *Find the distance $D(k)$ between the unknown utterance and the L templates using m of the M MFCCs.*

Step 2 : *Select the models with distance $D(k)/D(0) < T_1$.*

Step 3 : *Refine the distances $D(k)$ for the models chosen in* **Step 2** *using the remaining $M - m$ MFCCs.*

Step 4 : *Select the models with distance $D(k)/D(0) < T_2$.*

Step 5 : *Use HMM to find the best model among the chosen models in* **Step 4**.

This system is shown on the right in Figure 2.7.

Importance of the MFCCs can be determined by dropping one MFCC at a time and finding the effect on N best recognition by DTW. The MFCC that introduces the most error in recognition is declared as the most important and so on. From our experiments the coefficients were ordered from most important to least important as [2,3,0,1,6,7,4,8,5,10,11,9]. Note that for both the algorithms the number of models retained at each intermediate step can be variable depending on the unknown utterance. Also, if at any intermediate step the number of models retained is only one, then the subsequent recognizers need not be used and the unknown utterance is classified as the digit corresponding to the retained model. As before, thresholds T_1 and T_2 can be varied to trade-off between complexity and recognition-performance. Since the initial stage(s) is (are) used primarily to speed up the recognition operation, we can use the base layer for the decision process. During refinement by the HMM the enhancement layer can be used to enable more accurate representation of the feature vectors. The DTW computation can be reduced by exploiting the fact that the input data has been predicted and quantized. If the prediction error for the entire frame is quantized to zero, we do not need to find the distance of this frame from all the reference frames, instead the distance computed for the previous frame can be repeated without incurring significant degradation. When the proposed scalable system is used the computational load at the server is reduced. In addition this can be advantageous even from the users perspective whenever the DTW recognizer is able to make the final decision the latency of that specific recognition task is reduced.

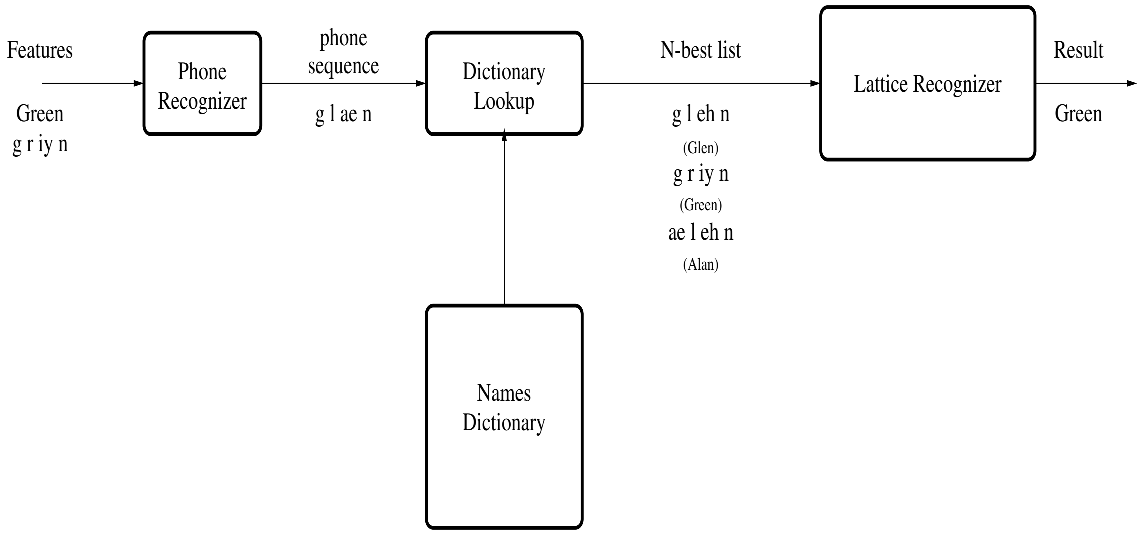

Figure 2.8: A two stage approach using dictionary lookup for the names recognition task. When a Levenshtein distance threshold of 2 is used, although the recognized phone sequence is closer to Glen than Green, the dictionary lookup will ensure that Green is in the final lattice used for rescoring.

Length of phone sequence	Threshold
less than 4	3
4 or 5	4
greater than 5	5

Table 2.1: Threshold used during dictionary lookup is a function of the recognized phone sequence length. Shorter phone sequences are assigned a lower threshold and vice-versa. The thresholds were chosen by experiments on the training data.

2.4.3 Spoken Names Recognition

The isolated digits task is a low perplexity task; we now consider scalable recognition for a high perplexity task. Consider a two stage spoken name recognizer with dictionary lookup (Figure 2.8). Such a two stage approach has been used for spelled name retrieval [29], information retrieval [15], complexity reduction of a phone based continuously speech recognition system [1] and spoken name recognition [57]. In the first stage a low complexity bigram phone loop is used to identify the N-best phone sequence corresponding to the input utterance. The next step involves a string match, where each of the N-best phone sequences is compared to the entries in a dictionary. The utterances corresponding to phone sequences which have a distance less than a given threshold (the threshold is usually chosen as a function of the number of phones in the recognized phone sequence, see Table 2.1) from the recognized N-best phone sequence are selected to generate a lattice. The final stage involves rescoring this generated lattice using more complex acoustic (triphone) models.

Consider a spoken names recognition task [5, 23], which, among other, is used in network based applications e.g., directory assistance, caller identification. In these applications the list of names tends to be quite large, in the order of hundreds of thousands. Variability in pronunciation further increases the perplexity. The traditional approach to name recognition has been to use a finite state grammar (FSG), where all the names (with all possible pronunciation variants) are alternate paths for recognition. For a name utterance the recognizer evaluates all possible paths and selects the name corresponding to the most likely path. As the names list grows it is evident that the computational complexity increases and the recognition accuracy will drop. An alternative approach with reduced computational complexity is to adopt a two stage recognizer with dictionary lookup. Figure 2.8 illustrates this approach. The accuracy obtained by the two stage recognizer for spoken names task is comparable to the conventional single stage FSG based approach (as shown in [57], where the single stage results are compared to the two stage recognizer) but results in significant savings in complexity, since the lattice only consists of a subset of the entire names list. The dictionary used for lookup is a names dictionary which consists of all possible names along with their pronunciations. In our experiments we used the Levenshtein (or edit) distance during dictionary lookup to compute the string match distance between phone sequences. The Levenshtein distance between phone sequences p_1 and p_2, $LD(p_1, p_2)$, is the minimum cost associated in transforming p_1 into p_2 by deletions, insertions and substitutions. This two stage names recognition procedure is summarized below.

Algorithm 3 *(Scalable Spoken Names Recognition)*

Step 1 : *Identify the N-best phone sequences p_r^n for the name utterance using a bigram phone loop, $n = 0, 1, \ldots, N-1$.*

Step 2 : *For each of the variable length phone sequences p_r^n find the corresponding threshold T_n from Table 2.1, $n = 0, 1, \ldots, N-1$.*

Step 3a : *Initialize $i = 0$*

Step 3b : *For name i in the names dictionary find the corresponding phone sequence p_i*

Step 3c : *If $LD(p_r^n, p_i) < T_n$, for any $n = 0, 1, \ldots, N-1$, add name i to the names lattice.*

Step 3d : *If there are more names in the dictionary set $i = i + 1$ and go to* **Step 3b**

else go to **Step 4**

Step 4 : *Rescore the names lattice using context-dependent models to get the final result.*

It will be shown in Section 2.6.3 that this two stage names recognition procedure can be efficiently combined with the proposed scalable encoder to reduce the overall latency experienced by the user.

2.4.4 A two-stage continuous speech recognition

As a third example, we consider a more standard continuous speech recognition task using the HUB4 data. Many practical speech recognition applications are impeded from using advanced speech recognition techniques due to their intensive computation/memory complexities. Simpler speech recognition technologies can instead be used. However, they result in recognition performance degradations. Similar to the two stage recognizer for the spoken names task, a multi-pass recognition scheme has been proposed [38] which enables improved speed/accuracy tradeoff by using progressive search techniques. These techniques use an "early-pass" reduced complexity speech recognizer to reduce the search space of a "later-pass" more accurate but complex speech recognizer. This procedure can be repeated iteratively, with each stage result used to constrain the search space for the next stage.

The early-stage recognizer builds a word lattice[1] containing several most likely word sequences using its low complexity models. The latter-stage uses this word lattice to constrain its search space and find the most likely word sequence hypothesis using its more

[1]We have used the early-pass recognizer to build word lattices. However, our proposed system can also use progressive search lattices [38].

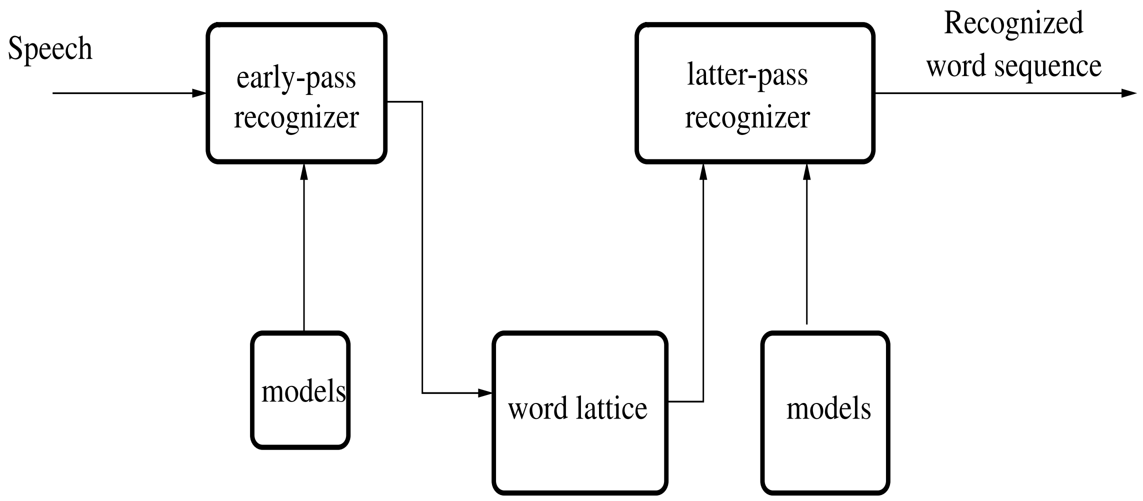

Figure 2.9: A two stage multi-pass continuous speech recognizer. The low complexity early-pass recognizer is used to constrain the search space of a latter-pass more complex recognizer by generating a word lattice.

complex models. Maintaining a reasonable sized word lattice at the early stage typically ensures that the "true" most likely word sequence is included in the word lattice, while only unlikely word sequences are discarded. Thus the word lattice, while constraining the search pass of the later-pass recognizer, does not significantly degrade its recognition performance. Hence, this multi-pass procedure ensures that *simultaneously* good recognition performance and reduced decoding time are achieved, making them ideal for use in practical speech recognition applications. Figure 2.9 illustrates a multi-pass recognition system consisting of two stages.

The multi-pass recognition procedure when a scalable encoder is used at the client is summarized below.

Algorithm 4 *(Multi-pass Distributed Continuous Speech Recognition)*

Step 1: *Use the base layer data with the early-pass low complexity speech recognizer.*

Step 2: *Recognize the utterance and dump the word lattice corresponding to it.*

Step 3: *Use the enhancement layer data and rescore the word lattice with the high complexity latter-pass speech recognizer.*

(a) *Update the acoustic probabilities using the enhancement layer data.*

(b) *Update the language probabilities using the more complex LMs.*

(c) *Find the most likely word sequence.*

If more than 2 stages are used in the multi-pass recognizer, the 2^{nd} stage also provides a (more refined) word lattice. The 3^{rd} stage uses this word lattice along with an additional enhancement layer and so on. It should be noted that the number of recognition stages required depends on the performance requirements for a given application, and the confidence in the recognition results of a given stage such as for example determined by confidence scores. As a consequence, it is possible that just a single stage recognition with base layer data may be adequate in some scenarios.

2.4.5 Scalable system operation

The most important constraints in the DSR system are

- User delay
- Client bandwidth
- Server bandwidth
- Server complexity

Let us consider how the scalable recognition system we propose can be used to work under each of these constraints

2.4.5.1 User delay

When the most important constraint is the time taken between the user speaking and the result of recognition, we can generate both the base and enhancement layers immediately and transmit both to the server. The server uses the base layer with the first stage recognizer and if only one model is present in the N-best list, the result is sent back to the client; if the N-best list contains more than one model, the enhancement layer (which is already available at the server) is used by the second stage recognizer to get the final recognition result. By this method we can reduce the delay experienced by the user, while keeping low the server complexity. However the client and server bandwidth requirements will be increased.

2.4.5.2 Client and Sever bandwidth

In bandwidth constrained situations, initially only the base layer is transmitted to the server. After the first stage (see Figure 2.1), if required, the enhancement layer is requested from the client. As can be seen from this procedure, both client and server bandwidths can be low, and the server complexity can also be kept low, as long as the first stage finds the best candidate from time to time. However the absolute delay experienced by the user can be high (for cases where the first stage is not able to make the final decision).

2.4.5.3 Sever complexity

Irrespective of all other constraints, we can always ensure that the complexity at the server is reduced, as mentioned in the above two cases. However when user delay is a constraint, the memory requirements at the server will be increased since the enhancement layer will have to be stored for future use.

2.5 Model Adaptation

In the previous sections it was assumed that the models used by the speech recognizer which were trained using unquantized features were kept fixed. In this section we investigate the complementary problem of optimizing the speech recognizer to take into account that it is operating on compressed speech.

One of the major problems for robust speech recognition is the mismatch between the training and testing conditions. Speech recognition performance, with speech models trained on clean data, significantly degrades when the test utterances are noisy (channel noise, ambient environment noise). Similarly the performance is also degraded due to long term and short term speaker variations. It is well known that speaker dependent models usually outperform speaker independent models. To improve robustness, techniques proposed involve (i) finding invariant features; (ii) allowing model parameters/feature vectors to vary within a neighborhood specified by the training data; (iii) transforming the models so they are more likely to have produced the observed data; (iv) incorporate newly acquired application specific data into existing models.

Here we consider an additional source for mismatch, namely speech encoding, which arises in DSR systems we study, as well as in situations where speech is encoded with standard speech encoders then remotely recognized. The distortion introduced by speech encoders can also be thought of as a mismatch between the training and testing conditions.

It is relatively easy to remove this mismatch by the use of a family of models each trained with data from different encoding schemes, and choose the one that best matches the unknown test data. However, such schemes are not attractive since it might not be possible to have models trained for all different compression schemes, because the choice of the compression scheme used by the client may be made dynamically depending on the computational resources/load at the client and the quality of service (QoS) it wishes to provide the user. Scalable encoders, which could be combined with scalable recognition schemes (see Section 2.4), wherein the recognition is refined in every pass with more data (and/or better models) until a satisfactory decision (say in the likelihood sense) can be made, further complicates the creation of pre-defined models. Depending on the optimization criteria used for compression (recognition performance or human perception), more variability in the compression schemes used by the different clients can be expected.

This mismatch introduced by the choice of different speech compression schemes can be solved in similar manner as other mismatches. The models at the server can be trained using clean speech (or a particular compression scheme) and we can alleviate the mismatch between testing and training phases by the use of model transformation/adaptation to optimize classification by ensuring that the transformed/adapted models are more likely to have produced the observed data [60]. Note that simple signal processing techniques are not likely to be helpful as the distortion introduced by compression is non-linear. However, adaptation schemes, which operate on the models rather than the input speech are more likely to be able to reduce the mismatch.

Two popular adaptation techniques which have been used previously are Maximum Likelihood Linear Regression (MLLR) [73] and Maximum a posteriori (MAP) [25] estimation. In the MLLR technique a transformation is computed for the means and variances of the different mixture components after observing the new data. Regression classes are defined to facilitate transformation even when a limited amount of data is observed. MAP in contrast assumes that model parameters are random and have a prior distribution. The observed data can be combined with the existing models to obtain new models by maximizing the posterior density of the models given the observed data. Unlike MLLR, in MAP we can modify not only the means and variances of the Gaussian mixtures but can also modify the mixture weights, the initial probabilities and the transition probabilities. For both methods, adaptation can be carried out either in batch mode or in an incremental manner. In batch mode adaptation (or supervised adaptation) the transcription corresponding to the unknown utterance is available. Incremental adaptation (or unsupervised adaptation) does not require the transcription and the result of recognition is used as the "true" transcription of the unknown utterance. We used unsupervised adaptation for MLLR and supervised adaptation for MAP.

2.6 Experimental setup

2.6.1 Experimental conditions

An HMM based recognizer (HTK3.0) was used to test the variable-rate DSR encoder developed in Section 2.2 and the scalable DSR encoder developed in Section 2.3. In all our experiments MFCCs were extracted and encoded at the client and the Δ and $\Delta\Delta$

coefficients were derived at the server from the decoded MFCCs and the HMM speech recognizer used was HTK 3.0..

2.6.2 Isolated digits and alphabet recognizer

The speech utterance was segmented using overlapping Hamming window of length 24 ms, with adjacent windows separated by 12 ms. 12 MFCCs derived from each segment of the speech utterance were used as features. A left to right HMM with five states and two Gaussian mixtures was trained with unquantized MFCC front-end data, for each utterance. Diagonal covariance matrices were used. A five state silence HMM was used before and after every digit HMM. The baseline performance was determined by recognizing speech with unquantized MFCCs. The MFCC encoders proposed were tested with recognition experiments using encoded MFCCs. For comparison, experiments were also performed on speech which had been coded by different low-bit rate speech encoders MELP (2.4 kb/s) and FS-10 (CELP, 4.8 kb/s). In this case, MFCCs were computed from the decoded speech. Better recognition performance may be possible by using waveform based speech coders (PCM, ADPCM), but these would require much higher bitrates (64 kb/s, 24-40 kb/s). So these high rate speech coders are not considered in this thesis. Comparison was also done using methods reported in [19] and [51].

The variable-rate MFCC encoder was tested with recognition experiments for two databases, the TI46-Word digit database, which contains discrete utterances of digits and the TI46-Word alphabet database, which consists of discrete utterances of letters. The MFCC encoders proposed were designed based on the front-end data from the digit database and the same encoders were used for both databases. The HMM training was done with 80 utterances from 4 male and 4 female speakers (80 utterances from 8 male speakers) and the total number of test utterances used was 3320 (1260) for the alphabet (digit) experiment. Test data were from the same speakers as the training data, but comprised of different utterances.

The HMM used for the scalable system was the same as above. The DTW templates were trained using digit utterances from the TI46-Word digit database. One template per digit was used. The first DTW stage used the 6 most important MFCCs to generate a N_1-best list and the second DTW stage refined this result using the other 6 MFCCs to generate a N_2-best list. The output of the last DTW stage was refined by the HMM.

2.6.3 Spoken names recognizer

For the names recognition task 12 MFCCs and the zeroth cepstral coefficient were extracted using an overlapping Hamming window of length 25 ms, with adjacent windows separated by 10 ms. For the context-independent (CI) models 3 stage left to right HMMs with 8 mixtures per state were trained, and for context-dependent (CD) models 3 stage left to right HMMs with 4 mixtures per state were trained. The speech corpus used for testing was the OGI NAMES corpus [17]. This is a collection of name utterances spoken by different speakers over the telephone. For our experiments we used 4619 names of which 3498 were unique (i.e., the 1121 names had multiple pronunciations). 6356 spoken names were used for training and 3000 different spoken names were used for testing. The top two phone sequences from the phone recognizer were used by the dictionary lookup to generate the lattice of names for rescoring.

2.6.4 Continuous speech recognizer for the HUB-4 broadcast news task

The acoustic models were 1128 context-dependent models trained as 3 stage left to right HMMs with 8 Gaussian mixtures per state. The speech corpus used for training was the 1994 HUB-4 speech corpus. 2200 sec of speech from the 1995 HUB-5 speech corpus were used for testing[2]. Both the train and test speech utterances contain significant variability (background music, different recording conditions and speakers), making this a difficult recognition task. The vocabulary size was 1300 words. The language models were the bigram and trigram broadcast news LMs provided by CMU. The number of bigrams and trigrams were $427 * 10^3$ and $1.8 * 10^6$ respectively. The early-pass recognizer employed a bigram LM and used the base layer data. The word lattices were generated as HTK SLF files by retaining the 20 best word sequences and using 5 tokens at every state [73]. The word lattice was used as the network for recognition by the latter-pass recognizer. The acoustic probabilities were updated using the enhancement layer data and the language probabilities were updated by rescoring the word lattice with a trigram LM. The average number of arcs in the word lattice was 21300. An unconstrained SLF generated from trigram LMs would have had more than $1.8 * 10^6$ arcs, indicating significant reduction in decoding time by using the multi-pass recognition system.

2.6.5 Model Adaptation

The model adaptation experiments were carried out on the TIDIGITS corpus using HTK 3.0 speech recognizer, again with MFCCs as the front end. The speech utterance was

[2]Utterances with significant speech and music overlap were eliminated from the test set. A few examples of the test utterances are available at *http://biron.usc.edu/˜ snaveen/speech_examples.*

segmented using overlapping Hamming window of length 25 ms, with adjacent windows separated by 10 ms. The database consists of variable length connected digit utterances (1 to 7 digits per utterance). The "train" part of the database consisted of 8623 utterances spoken by 55 male and 57 female speakers and the "test" part of the database consisted of 8700 utterances spoken by 56 male and 57 female speakers (the train and test speakers were different). For every digit context-independent digit models with ten states and two Gaussian mixtures were initially trained (on the server) using clean speech from the "train" part of the database. A silence model was used before and after the digit utterance to take care of the pre and post utterance silence. In addition a short pause model was used to account for inter-digit short pauses. The testing (using utterances from the "test" part of the database) was carried out using MELP compressed speech, GSM compressed speech and the non-layered MFCC encoder. The MFCC encoder was used at two different rates 2.07 kbps (denoted MFCC-HR) and 1.22 kbps (denoted MFCC-LR). The baseline performance was determined by using "matched" models for the different compression schemes, i.e., the training was done using speech encoded by the same method as that used during the testing phase.

The experiments were carried out for two different settings (i) unsupervised MLLR adaptation and (ii) supervised MAP adaptation. For the unsupervised MLLR adaptation, the models were adapted once every 20 utterances. For the supervised MAP adaptation the original models were adapted for each speaker individually, i.e., part of the testing data from each speaker was used to adapt the original models to that particular speaker and the adapted models were used to recognize the test utterances of that speaker.

The original databases TI46-Word and TIDIGITS contained speech sampled at 12.5 kHz and 20 kHz respectively. However MELP, FS-10 and GSM require the input speech to be sampled at 8 kHz. One method to overcome this would be to downsample the speech to 8 kHz, encode the speech and then upsample the decoded speech back to 12.5/20 kHz, however when this method was used the performance obtained was poor. The reason for this could be that the spectrum of the upsampled speech is flat from 4 kHz to 6.25/10 kHz while the spectrum of the original speech was not. To overcome this we can downsample all the speech (training and testing) to 8 kHz and perform all our experiments using this downsampled data. Now the training phase also uses downsampled speech to build the initial models. For consistency the MFCC encoder also was used with the downsampled speech data.

2.7 Results

2.7.1 MFCC encoder

Tables 2.2 and 2.3 compare the results of the proposed USQ technique to MELP and FS-10(CELP). It can be observed that while the increase in WER by encoding the MFCCs is small there are substantial savings in the bitrate. The advantage obtained by encoding the MFCCs as opposed to encoding speech and extracting the MFCCs from the decoded speech can be seen by comparing the WER obtained by the proposed USQ technique to the WER achieved by MELP and FS-10(CELP). It is apparent that although the proposed USQ technique uses lower bitrate than these speech codecs, it achieves lower WER than them. This gain in recognition performance and reduction in bitrate required is not surprising because the speech coders have been optimized to preserve the perceptual quality of the speech, while the MFCC encoders are designed to maximize recognition performance.

Encoding technique	WER	Bitrate (kbps)
Unquantized	0.21%	-
Proposed USQ technique	1.26%	1.02
MELP	1.15%	2.4
FS-10(CELP)	4.75%	4.8

Table 2.2: Comparative results for the digits database for different encoding techniques

Encoding technique	WER	Bitrate (kbps)
Unquantized	17.34%	-
Proposed USQ technique	18.82%	1.02
MELP	24.85%	2.4
FS-10(CELP)	25.69%	4.8

Table 2.3: Comparative results for the alphabets database for different encoding techniques

Comparing the results obtained with previously proposed techniques, it can be observed from Figures 2.10, 2.11 and 2.12 that the methods proposed in this thesis outperformed the algorithm reported in [19]. While the recognition performance of a VQ based technique proposed in [51], is good, this method has the disadvantages of higher bitrate and higher encoding complexity, when compared with the methods proposed in this thesis.

It is also evident from Figures 2.10 and 2.11 that the combined recognition-rate performance of USQ is better than the recognition-rate performance of ECSQ. This indicates that implicit pruning (by increasing the step size of the quantizers) of the MFCCs gives better results when compared to ad hoc pruning. The scalability of the proposed methods can be seen from Figures 2.10 and 2.11, which show the trade off in bitrate vs. WER. By accepting higher WER, we can operate at a lower bitrate (with more bits available for channel coding); this feature will be useful in situations such as in wireless communications where the channel conditions are varying.

The reduction in complexity at the client side by using our method can be seen from Table 2.4. Running the speech recognizer locally would require significantly more computation and memory than quantization. For example isolated digits recognition requires almost a factor of 3 more computation time than USQ.

Speech recognition	ECSQ	USQ
0.156 s	0.067 s	0.047 s

Table 2.4: Average CPU time (in seconds, on a sun workstation) required to recognize a utterance from the digit database, and time required to encode it. The times shown for ECSQ and USQ also include the time for entropy coding, as well as the time required to compute the MFCCs (also included in speech recognition). The encoding and recognition times can be expected to be much higher for a portable device.

2.7.2 Scalable DSR system

2.7.2.1 Isolated digits recognition

Figure 2.13 shows the WER as a function of average rate for Systems A & B. For System B the results for two different Δ_c/Δ_f ratios are shown. The bitrate for the base layer

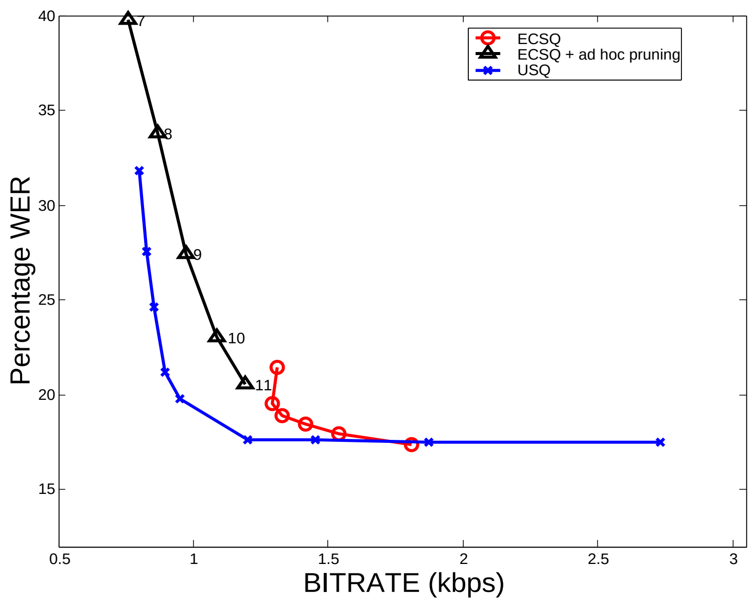

Figure 2.10: Recognition performance of the different encoders for alphabet database. Numbers next to the points indicate the number of coefficients retained. The scalability of the encoders can be seen by the Bitrate/WER tradeoff. Recognition performance with MELP was 24.85% and with FS-10 was 25.69%. The recognition performance with unquantized MFCCs was 17.34%.

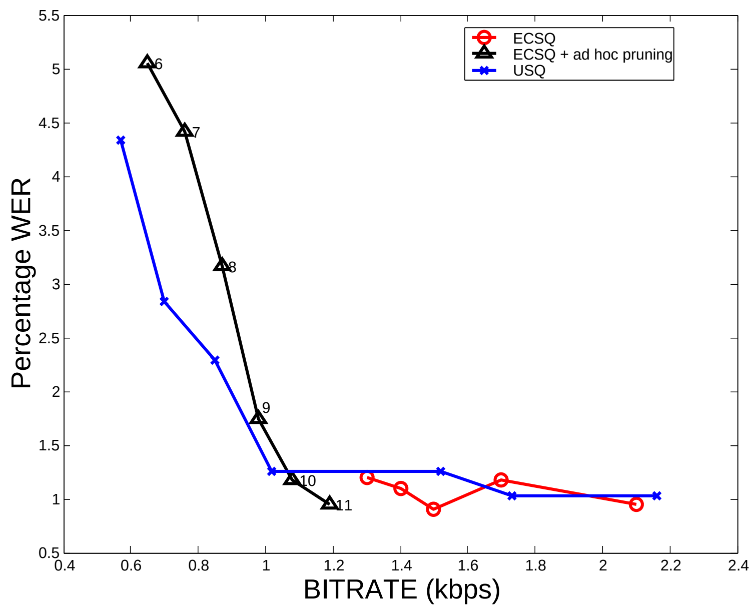

Figure 2.11: Same information as in Figure 2.10 except that the database is the digit database. WER with MELP was 1.15% and with FS-10 was 4.75%. The recognition performance with unquantized MFCCs was 0.21%.

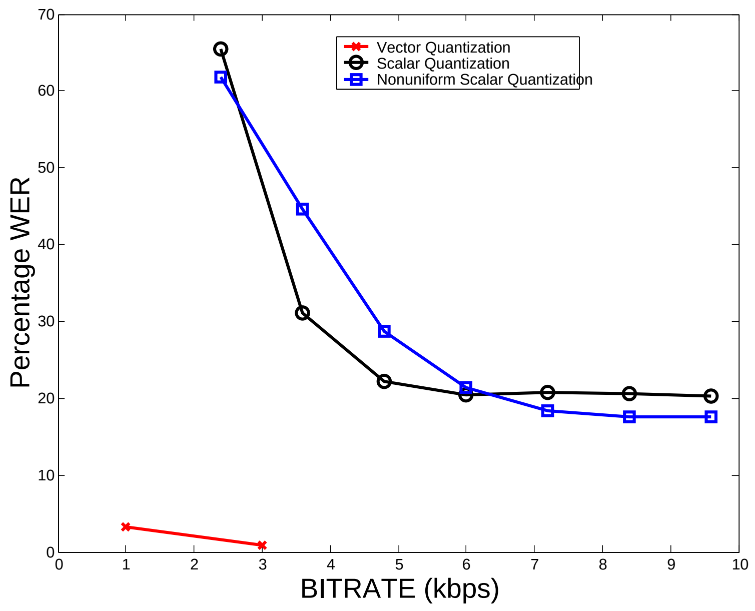

Figure 2.12: Recognition results with techniques based on scalar quantization [19] on the alphabet database and linear prediction and vector quantization [51] on the digit database.

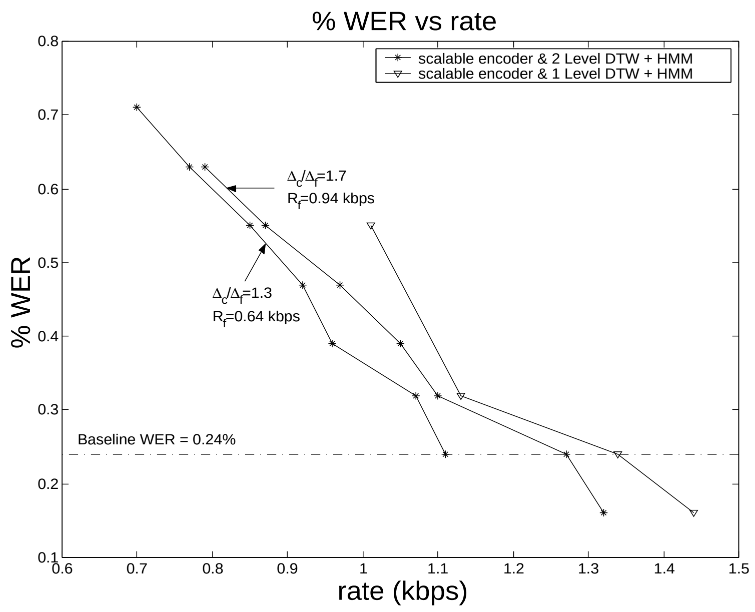

Figure 2.13: Recognition performance for the scalable recognition schemes. The coarse bitrate is 0.72 kbps. The fine bitrate (R_f) is indicated in the figure. The baseline WER of 0.24% is shown as a dotted line.

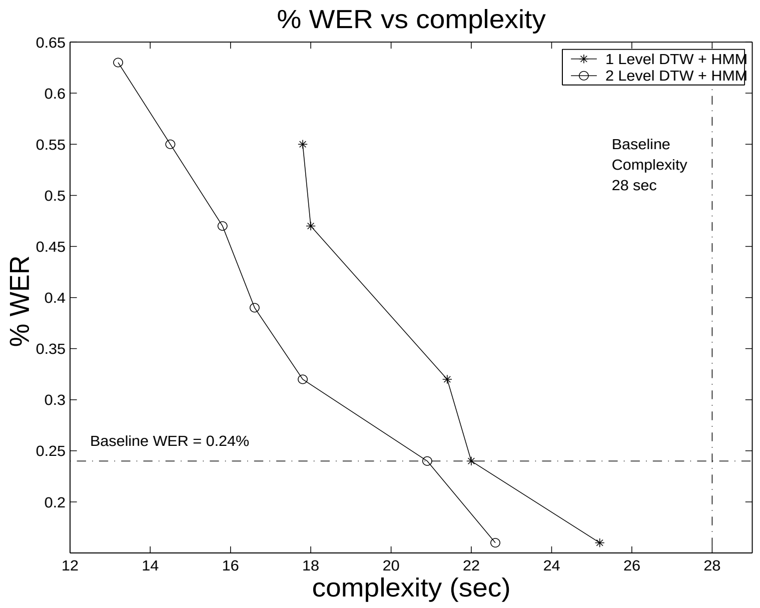

Figure 2.14: The time required for the scalable recognition schemes. The baseline computational time of 28 sec is shown as a dotted line. Notice the computational scalability wherein reduced complexity can be achieved at the expense of WER.

was 0.72 kbps for both systems. Since the first DTW of System B uses only the first 6 MFCCs of the base layer, initially we only need to transmit these, which requires a bitrate of 0.38 kbps while the other 6 MFCCs require 0.34 kbps. We observe that the WER is 0.32% and 0.24% at 1.13 kbps and 1.11 kbps for System A and System B respectively, when compared to the baseline WER of 0.24%. For System B we observe that the recognition-rate trade-off is better when Δ_c/Δ_f is smaller, however larger Δ_c/Δ_f provides a superior reconstruction of the enhancement layer enabling lower WER at a higher average bitrate. Figure 2.14 shows the trade-off in complexity and WER for both the Systems. Using only the HMM approach we need about 28 sec to recognize 1267 digit utterances (approximately 1400 sec of speech). We reduced the complexity by 21% and 25% by using System A and System B respectively with no degradation in WER. However if we are willing to tolerate more error we can reduce the running time by 53%

(i.e., more than halve the running time) by using System B while incurring a WER of 0.63% (163% increase over the baseline performance).

From Figures 2.13 and 2.14 it is apparent that with the scalable DSR system we can trade-off either bitrate or complexity vs. WER. The delay constraints, client/server bandwidth and server computational load will determine the optimal trade-off. The flexibility of the scalable system is that trade-offs are controlled only by the threshold of the initial DTW stage(s). By increasing the threshold, we reduce the average bitrate and the server complexity while incurring a higher WER. If recognition performance is important then we simply need to increase the threshold.

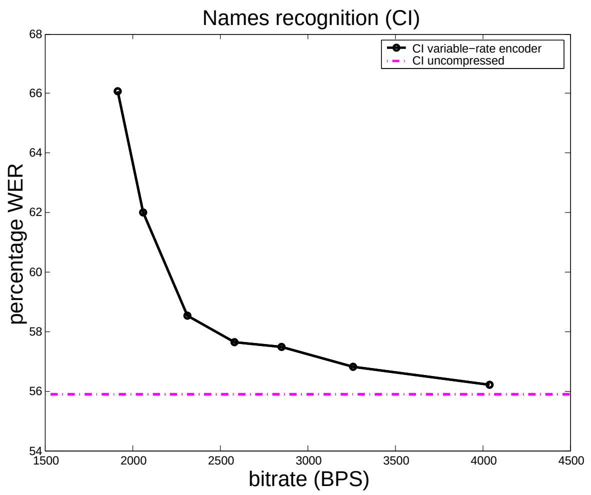

Figure 2.15: Names recognition results when CI models are used in the lattice recognizer. Notice that there is small degradation in performance when the bitrate is above 2500 b/s. However it is apparent that with the proposed variable-rate encoder we can trade-off recognition performance and bitrate.

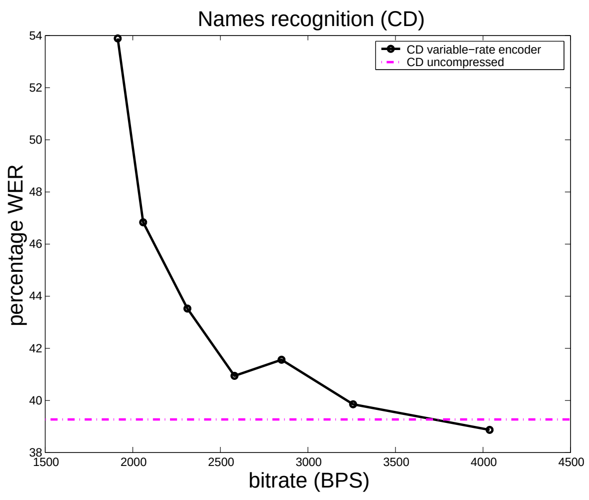

Figure 2.16: Names recognition results when CD models are used in the lattice recognizer. Notice that there is small degradation in performance when the bitrate is above 2500 b/s. However it is apparent that with the proposed variable-rate encoder we can trade-off recognition performance and bitrate.

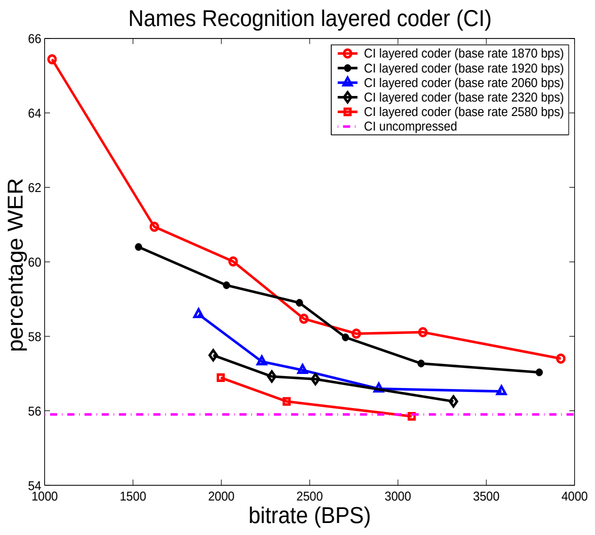

Figure 2.17: Results of the names recognition when the proposed scalable DSR encoder is used at the client. The results are shown for different base layer rates.

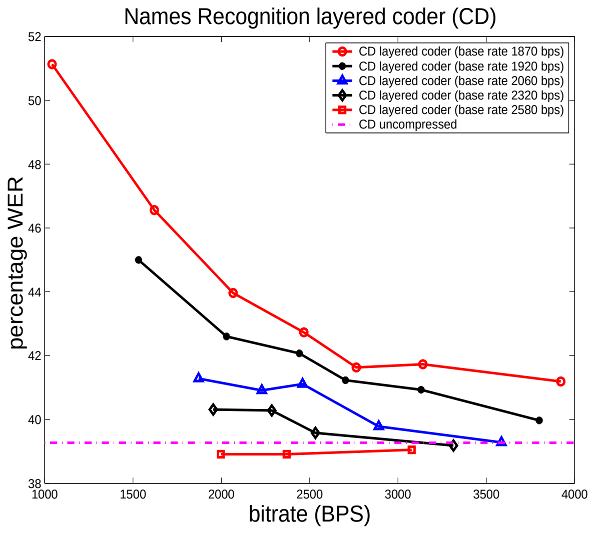

Figure 2.18: Results of the names recognition when the proposed scalable DSR encoder is used at the client. The results are shown for different base layer rates. When the base layer rate is 2580 b/s and the enhancement layer rate is 2000 b/s the recognition result for CD models is the same as that obtained with a variable-rate encoder at a bitrate of 4040 b/s. However since the base layer rate is lower than the full rate of the variable-rate encoder, the recognition latency experienced by the user is reduced.

2.7.3 Spoken Names Recognition

2.7.3.1 Variable-rate DSR encoder

When the spoken names recognition system is used in a DSR system with a variable-rate non-scalable encoder, both recognition stages, i.e., the phone loop and the lattice recognizer operate on the same compressed data. The phone loop is a bigram CI phone loop. The resulting lattice after dictionary lookup can be refined using either CI or CD models. The results obtained for these two cases for different bitrates are shown in Figures 2.15 and 2.16. We observe that in both cases there is small degradation in performance when the bitrate is greater than 2500 b/s. However it is clear that with the proposed encoder we can trade-off bitrate for recognition performance. The average number of names in the lattice when compressed data was used was approximately 1140, which is almost the same as when uncompressed data was used, i.e., compression did not increase the lattice size. Table 2.5 compares the degradation due to compression with the proposed encoder and *Aurora* encoder [21]. Notice that the proposed encoder provides both a lower rate and has a lower WER degradation than *Aurora*.

Encoding technique	CI	CD	Rate (b/s)
Aurora	0.86	0.77	4400
Proposed variable-rate encoder	0.33	0.13	4050
	0.36	0.47	3600

Table 2.5: Absolute percentage increase in WER for the proposed encoder and *Aurora*. Observe that even when the proposed encoder operates at 3600 b/s it is superior to *Aurora*.

2.7.3.2 Scalable DSR encoder

When the proposed scalable DSR encoder is used at the client, a base layer and an enhancement layer are transmitted to the server for every name utterance. Now the bigram CI phone loop uses the base layer to generate the N-best phone sequence. This is used by the dictionary lookup to build the list of names for the lattice recognizer. The lattice recognizer rescores the names list using the enhancement layer data to get the final recognized name result. Note that the phone recognizer and the dictionary lookup need not wait for the enhancement layer data to be received.

The recognition results obtained with the above procedure for the names task are shown in Figures 2.17 and 2.18. Transparent recognition performance (i.e., the recognition performance was the same as that achieved with uncompressed data) with CD models was achieved when the base layer rate was 2580 b/s and the enhancement layer was 2000 b/s. To achieve transparent recognition with our proposed encoder operating with a single layer (i.e., in a non-scalable mode) the required bitrate is 4040 b/s (Figure 2.16). In this case this single layer of encoded data at 4040 b/s is used by both recognition stages. Let the total user latency be defined as the sum of transmission time and recognition time. The use of scalable coding enables us to lower the user latency. Assume each name utterance is put into a single packet (non-scalable codec) or two packets (scalable codec with two layers). Table 2.6 shows the transmission times of the packets for different utterance lengths when using an 8800 b/s transmission link (8800 b/s is the maximum bitrate used by a speech codec on the fundamental channel (FCH) in a cdma2000$^{©}$ mobile network). It can be seen that the transmission time required for the base layer of the scalable codec is lower than the time required for the non-scalable codec. Hence, the first stage recognizer can be started earlier in the scalable case. This ensures that the second stage (lattice recognizer) can also complete faster in systems employing a scalable coder while achieving the same recognition performance. To illustrate the reduction in user latency, consider the transmission times shown in Table 2.6. Specifically consider the case when we are using a multi-pass scheme for both scalable and non-scalable encoding cases, and the length of the name utterance is 1 sec. For systems employing a non-scalable encoder, the time required to transmit the data from the client to server is 0.5 sec. Hence, the first stage recognizer can only be started 0.5 sec after the user finishes speaking. If we assume, without loss of generality, the speech recognizer works in real time (i.e., it takes 1 sec to recognize an utterance of length 1 sec), then the recognizer completes recognizing the utterance 1 sec after reception of the data. Hence, the total delay as experienced by the user is 1.5 sec. However, when we use the scalable codec, the base layer is transmitted in 0.33 sec. Now the first stage recognizer can be started 0.33 sec after the user finishes speaking. As the first stage recognizer is working the server can receive the enhancement layer. Again assuming the same real time recognition, the total delay for the user is now 1.33 sec. Hence, by using a scalable codec we could reduce the user latency by 0.17 sec. For different lengths of the name utterance it was observed that on average around 11% reduction in user latency can be achieved. The successive transmissions in the scalable case, can be pipelined, if needed. Thus systems employing a scalable coder are faster, thus reducing the latency perceived by the user, but are capable of achieving the same recognition performance (albeit at the cost of a small increase in rate).

Table 2.6: Transmission times for scalable and non-scalable codec on an 8800 b/s transmission link. The recognizer is assumed to operate in real-time. Hence time required for recognition is equal to the length of the name utterance. The user latency is the sum of the transmission and recognition time. Reduction in user latency for the proposed scalable encoder is shown in the last row. Around 11% user latency reduction is possible. Each packet is assumed to have 40 bytes of RTP/UDP/IP headers. If RoHC (Robust Header Compression) were used the header size could be reduced to 5 bytes.

Length of name utterance	0.5 sec	1 sec	2 sec
non-scalable codec	0.27	0.50	0.95
scalable codec (base layer)	0.18	0.33	0.62
percentage reduction in user latency	10.8%	11.1%	11.2%

2.7.4 Continuous Speech Recognition

When the proposed scalable DSR encoder is used at the client, the bigram recognizer uses the base layer to generate a word lattice. This word lattice is rescored by a trigram recognizer which in addition uses the enhancement layer to update the acoustic probabilities. As in the spoken names task, the initial bigram recognizer does not have to wait for the enhancement layer data to be received.

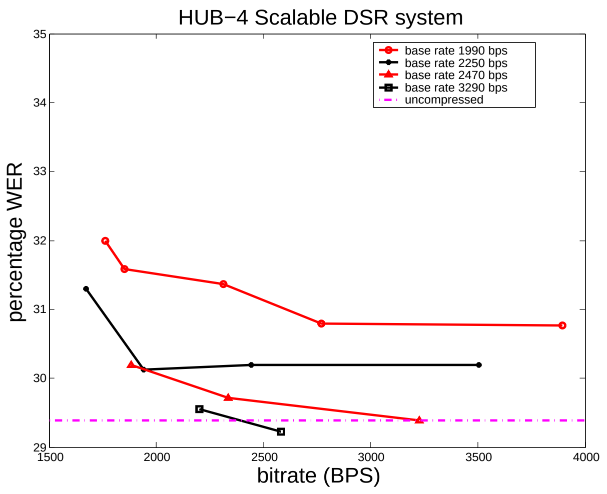

Figure 2.19: Recognition results of the multi-pass DSR for the HUB-4 broadcast news task. Observe that with a base layer rate of 2470 b/s and an enhancement layer of 3230 b/s we achieve the same recognition performance as with uncompressed data. Also the recognition performance versus bitrate trade-off is clear from the results.

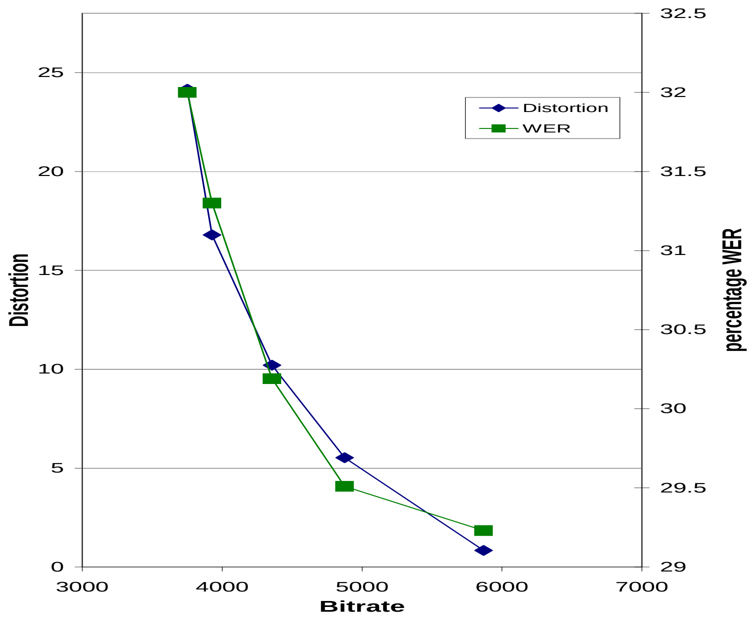

Figure 2.20: Comparison of trade-off between distortion and percentage WER versus bitrate. Distortion scale is shown on the left y-axis and percentage WER scale is shown on the right y-axis. The correlation between distortion and recognition performance is clearly illustrated.

The recognition performance achieved for the two stage multi-pass recognizer for several different base layer rates is shown in Figure 2.19. Observe that with a base layer rate of 2470 b/s and an enhancement layer rate of 3230 b/s, the recognition performance achieved is the same as that achieved with uncompressed data. To achieve transparent recognition with our proposed encoder operating with a single layer (i.e., in a non-scalable mode) the required bitrate is 5240 b/s. Hence we observer that, while using a scalable encoder provides more flexibility in the overall system design it only results in an 8.7% increase in bitrate.

The trade-off between bitrate and recognition performance is also clear from the figure. For the base layer rate of 2470 b/s, reducing the bitrate from 3230 b/s to 1880 b/s (42% reduction) results in only a 2.7% relative increase in WER. In Figures 2.18 and 2.19, the recognition performance with compressed data is sometimes better than with uncompressed data. The difference in recognition performance for these points from the uncompressed recognition performance is not statistically significant.

An interesting observation from the results is that the recognition performance plateaus at different WERs for different base layer rates (with the plateau being higher for lower base layer rates). This indicates that a certain minimum data fidelity is required at the initial recognition stage, below which improving the data fidelity only at the later stages does not enable the system to achieve the same recognition performance as that achieved by uncompressed data, no matter how high the enhancement layer bitrate is made. This provides a guideline for selection of bitrates for the base and enhancement layers. Given that a particular recognition performance is required for the task, this implicitly decides the minimal bitrate that can be used for the base layer. For example in the CSR task if the desired recognition performance of the overall system has to be as good as that achievable with uncompressed features then, from Figure 2.19, we observe that the base layer rate has to be greater than 2470 bps. However, this also implies that given the constraint that a particular rate has to be used for the base layer (this can be due to application/channel constraints) then there is no necessity to use a (high) rate for the enhancement layer which lies beyond the knee of the recognition performance curve corresponding to that particular base layer rate. For example, at a base layer rate of 2250 b/s there is no gain in increasing the enhancement layer rate beyond 1940 b/s.

Figure 2.20 shows both the distortion due to compression and the percentage WER for different bitrates. It is clear that as the bitrate is increased the distortion is reduced. Similarly, the percentage WER also decreases with increase in bitrate. The two curves have a similar tradeoff against bitrate. This shows that there is a strong correlation between distortion and percentage WER. This relationship can be used to predict the impact compression has on recognition performance.

Another interesting observation from Figures 2.13, 2.17, 2.18 and 2.19 and Table 2.7 is that the bitrate required to ensure that speech recognition performance is not degraded due to compression is 1100b/s for an about isolated digits task, 2000 b/s for a connected digits task, 4600 b/s for the spoken names task and 5700 b/s for the CSR task. This illustrates that the minimum bitrate for transparent speech recognition is strongly task dependent. In general, more complex speech recognition tasks require higher bitrate. It will be interesting to analytically quantify the minimum bitrate requirement for different speech recognition tasks. We leave this as potential future work.

Encoder	Clean Models (E_{CM})	Clean Models + MLLR (E^A_{CM})	Matched Models (E_{MM})	Matched Models + MLLR (E^A_{MM})	MLLR gain
Clean speech	1.88 (7.56)	1.57 (6.57)	-	-	16.5%
MELP	3.14 (12.07)	2.32 (8.70)	2.70 (10.47)	1.87 (8.53)	26.1%
GSM	2.50 (8.76)	1.73 (7.33)	2.29 (8.61)	1.55 (6.91)	30.8%
MFCC-LR	4.81 (14.78)	2.24 (8.49)	2.70 (10.25)	1.85 (8.08)	53.4%
MFCC-HR	2.10 (8.06)	1.60 (6.82)	2.05 (7.87)	1.58 (6.87)	23.8%
Aurora baseline	1.91 (8.68)	1.32 (6.39)	-	-	30.9%
Aurora	2.30 (9.26)	1.52 (7.09)	2.38 (9.48)	1.51 (7.03)	33.9%

Table 2.7: Word error rate (in percentage) for supervised MLLR adaptation. String error rate (in percentage) is shown in brackets. The improvements in MLLR are decrease (in percentage) in word error rate with respect to clean model results.

2.7.5 Model Transformation

2.7.5.1 Adaptation with a Single Model

Table 2.7 shows the results for the unsupervised MLLR adaptation experiment. The Aurora compression scheme is optimized for the front end defined by the Aurora standard, all other encoders were used with the HTK front end. The results with the Aurora when no compression is used is indicated as "Aurora baseline". Note that under clean conditions (i.e., no compression) the Aurora front end performs better than the HTK front end. We observe that, consistently for all the compression schemes, MLLR adaptation results in good improvements in the recognition performance. The results after adaptation are in fact better than when "matched" models are used. This is because we are using unsupervised adaptation and updating the models once every 20 utterances, and the utterances from each speaker are together, so we are benefiting from inter utterance similarities (as indicated by the improved performance with adaptation on clean speech).

To ensure that the comparisons are consistent we performed adaptation on the matched models (shown in column 5 of Table 2.7).

To show the advantage of adaptation with a single model, we can compute the degradation before and after adaptation. These can be evaluated by comparing the recognition performance with clean models to the recognition performance with matched models (baseline, no mismatch in training and testing). The single model degradation before adaptation is defined as $D = (E_{CM} - E_{MM})/E_{MM} * 100$, where E_{CM} is the error when compressed data is used for testing and clean speech is used for training, E_{MM} is the error when compressed data is used for training and testing. Similarly the single model degradation after adaptation is defined as $D^A = (E^A_{CM} - E^A_{MM})/E^A_{MM}) * 100$ (E^A_{CM} and E^A_{MM} are defined as above except that adaptation is used). These degradations are shown in Table 2.8. Observe that there is significant degradation before adaptation for MFCC-LR. However after adaptation the degradation is reduced substantially. For MFCC-HR, by adaptation from clean models we get almost same performance as adaptation from matched models (1.60 % vs. 1.58 % (see Table 2.7)). These results imply that with adaptation from a single model we are not only able to reduce the absolute error rates but we are also able to reduce the degradation from matched conditions (for MELP and GSM the relative degradation increased but the absolute error rate decreased; for GSM the relative increase was very small). This result is very significant because it demonstrates that we do not need encoder specific models to be trained at the server, instead we can achieve the same performance with adaptation of models trained from clean speech.

The results of the supervised MAP adaptation are shown in Table 2.9. The supervised MAP results are better than the unsupervised MLLR results as expected. For MELP

Compression	Degradation before Adaptation (D)	Degradation after Adaptation (D^A)
MELP	16.30	24.06
GSM	9.17	11.61
MFCC-LR	78.15	21.08
MFCC-HR	2.44	1.27
Aurora	-3.36	0.66

Table 2.8: Degradation (in percentage) in word error rate before and after adaptation for the different coding schemes. The degradation is with respect to using matched models for each compression scheme.

and GSM, MAP adaptation provides better results when compared to MLLR, however for the MFCC encoders the MAP performance does not provide as significant a decrease as for MELP and GSM (in fact for MFCC-LR, the MAP performance was worse than the MLLR performance). The reason for this could be that while MLLR does not model the initial parameters as a random vector MAP explicitly does. The MFCC encoder quantizes the MFCCs directly and this means that the actual distribution of the encoded MFCCs is not a continuous distribution anymore but a discrete distribution. However in the MAP formulation the MFCCs are modeled as continuous distributions and the conjugate distribution which lies in the same class as the original distribution is used as the prior distribution. Therefore the MAP formulation is no longer optimal and this could be making MAP less effective than MLLR for the MFCC encoders. Nevertheless, the improvement by using MAP adaptation is obvious from the results; we get more than 60% reduction using MAP for GSM and MELP. The reductions for the other methods are also significant.

Compression	Clean Models (E_{CM})	MAP (E_{CM}^A)	MAP gain
Clean speech	1.86 (7.54)	0.67 (3.85)	64.0%
MELP	3.12 (12.05)	1.19 (6.06)	61.9%
GSM	2.48 (8.72)	0.91 (4.09)	63.3%
MFCC-LR	4.78 (14.73)	3.34 (10.89)	30.1%
MFCC-HR	2.08 (8.01)	0.91 (4.36)	56.3%

Table 2.9: Word error rate (in percentage) for supervised MAP adaptation. String error rate (in percentage) is shown in brackets. The improvement for MAP is decrease (in percentage) in word error rate with respect to clean model results.

2.7.5.2 Encoder Optimized for Recognition

As mentioned before, compression introduces degradation in recognition performance. The compression degradation can be found by comparing the results with compression to those obtained with clean speech. The compression degradation before adaptation can be found as $D_C = (E_{CM} - E_{UC})/E_{UC} * 100$ and the compression degradation after adaptation can be found as $D_C^A = (E_{CM}^A - E_{UC}^A)/E_{UC}^A * 100$, where E_{UC} is the error when clean speech is used for training and testing (E_{UC}^A corresponds to the case when adaptation is used). These degradations are shown in Table 2.10 along with the rate required for the different compression schemes. The rate required by the MFCC encoders is significantly less than that required by GSM and is less than that required by MELP. However minimum degradation is introduced by the MFCC-HR encoder among all the compression schemes (WER only degraded from 1.57 % to 1.60 %). Also notice that after adaptation the MFCC-LR encoder operating at half the rate of MELP actually provides better results than MELP. This further justifies our initial claim that compression schemes optimized for recognition should be used to compress speech used for recognition for better performance.

Compression	Degradation before Adaptation (D_C)	Degradation after Adaptation (D_C^A)	Rate (kbps)
MELP	67.02	47.77	2.4
GSM	32.98	10.19	13
MFCC-LR	155.85	42.68	1.22
MFCC-HR	11.70	1.91	2.07
Aurora	20.42	15.15	3.60

Table 2.10: Degradation (in percentage) in word error rate before and after adaptation for the different coding schemes. The degradation is with respect to using clean speech for testing. Uncompressed speech requires 128 kbps and uncompressed MFCCs require 38.4 kbps.

Also note that the MFCC-HR scheme operating at a bitrate of 2.07 kbps has just 1.97 % degradation (11.70 % before adaptation) over clean speech performance, but the Aurora scheme operating at 3.60 kbps (the Aurora scheme actually requires 4.8 kbps, but that includes the 800 bps for energy and C0 both of which are not used in our experiments and 400 bps for error protection) has 15.15 % degradation (20.42 % before adaptation) over clean speech performance. Another important point to be noticed from this table is that consistently for all the encoding schemes the degradation after adaptation is lesser then the degradation before adaptation, which implies that adaptation is compensating for the compression mismatch in addition to compensating for other mismatches.

2.7.5.3 Effect of Adaptation Data

It is also important to find the dependency of the adaptation schemes on the amount of input data required. To find this we used MLLR adaptation in supervised mode and changed the amount of data used for adaptation. The experiments were carried out for clean data, MELP, GSM and MFCC-encoded data. The number of speakers in the test corpus was 113. The number of utterances chosen per speaker for adaptation was 1, 2, 4 and 8 resulting in 113, 226, 452 and 904 utterances used as adaptation data for the four different cases. The results of string error rate and word error rate are shown in Figure 2.21. Observe that with increased adaptation data the error rates decrease for the different encoders. Using more than 904 utterances provided no further improvement in performance. One of the drawbacks of the proposed scheme is that improvements are seen only after sufficient adaptation data has been observed, which may not be practical in some situations. This is basically a problem of the adaptation schemes (MLLR and MAP) which we have used here. To overcome this it may be necessary to combine MLLR/MAP adaptation with other rapid adaptation schemes which can operate with lesser adaptation data.

2.8 Conclusions

In this chapter we addressed the optimization of a DSR system. We showed that using speech encoders optimized for recognition rather than perceptual distortion significantly provides better recognition performance. In practical scenarios where the large number of clients can severely overload the DSR server, it is desirable to use a scalable system which provides trade-off between bitrate, complexity and recognition performance. To provide a multi-resolution compressed stream it is more convenient to work with MFCC encoders, as now the client has greater flexibility in controlling the input to the HMMs. We showed that our proposed scalable recognizer combined with a layered MFCC encoder can provide flexibility in adapting the DSR system to the changing bandwidth requirements and server

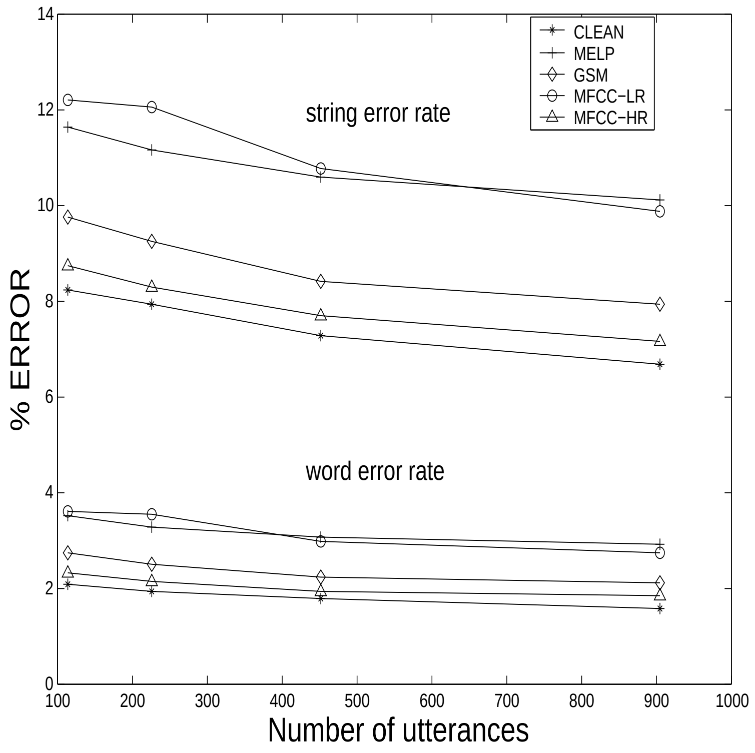

Figure 2.21: Effect of adaptation data on string error rate and word error rate for clean and encoded data.

load while achieving the best recognition performance possible. Large number of clients density also implies that the recognizer has to work in conjunction with a variety of different speech encoding schemes. We showed that conventional adaptation schemes can be effectively used to reduce the degradation caused by speech encoding. This is potentially very important in a practical system because now only one set of HMM models need to be trained at the server, and these models can be reused for other encoding schemes without significantly degrading the recognition performance. By addressing the DSR system optimization at the encoder, at the recognizer and at the system level we can ensure that the system operation is highly robust to user and network conditions. Each of these schemes gives us more freedom to fine tune the system enabling the DSR system to be adapted to provide good service to the user while constantly adjusting to the network and server conditions.

Chapter 3

Joint Compression-Classification with Quantizer/Classifier Dimension Mismatch

3.1 Introduction

In the previous chapter a speech coding technique optimized to improve recognition performance was proposed. As mentioned there, one of the constraints on the encoder was that it should be computationally inexpensive to enable implementation on low power mobile devices. In order to achieve moderate complexity, the encoder operated making the best coding decision independently in each MFCC frame. Clearly this does not guarantee overall optimality of encoding, given that the HMM operates on sequence of frames and the recognition decision is based on the entire sequence of frames. To ensure minimal degradation the entire sequence of frames used for recognition should have been jointly encoded by the encoder. However this operation is very computationally expensive and thus independent encoding of the frames remains a better practical approach. Given that independent encoding cannot incorporate the effect on recognition during the encoding operation, there is a need to study the problem of low dimension encoders each of which encode a sub-vector of a vector and the output of which is used by a classifier which operates on the encoded vector.

Consider an N dimensional vector $\mathbf{x}$, let the subvectors of $\mathbf{x}$ be $\mathbf{x_1}$ (of dimension N_1) and $\mathbf{x_2}$ (of dimension N_2), i.e., $\mathbf{x} = [\mathbf{x_1}\mathbf{x_2}]$ (Note that for purposes of explanation we are assuming that $\mathbf{x}$ has only two sub-vectors, but the arguments extend to cases when $\mathbf{x}$ has more sub-vectors). Let $\alpha_1(\cdot)$ and $\alpha_2(\cdot)$ be two encoders which operate on the N_1 and N_2 dimensional vectors; let $\beta_1(\cdot)$ and $\beta_2(\cdot)$ be the corresponding decoders; and let $\delta(\cdot)$ be an N dimensional classifier. We consider the distributed classification system shown in Figure 3.1. At the client the encoders $\alpha_1(\cdot)$ and $\alpha_2(\cdot)$ independently encode $\mathbf{x_1}$ and $\mathbf{x_2}$. At the server the decoded data is combined and used for classification by δ. The problem we want to solve is that of designing the encoders $\alpha_1(\cdot)$ and $\alpha_2(\cdot)$ and the decoders $\beta_1(\cdot)$ and $\beta_2(\cdot)$ so as to ensure that they have the least effect on classification.

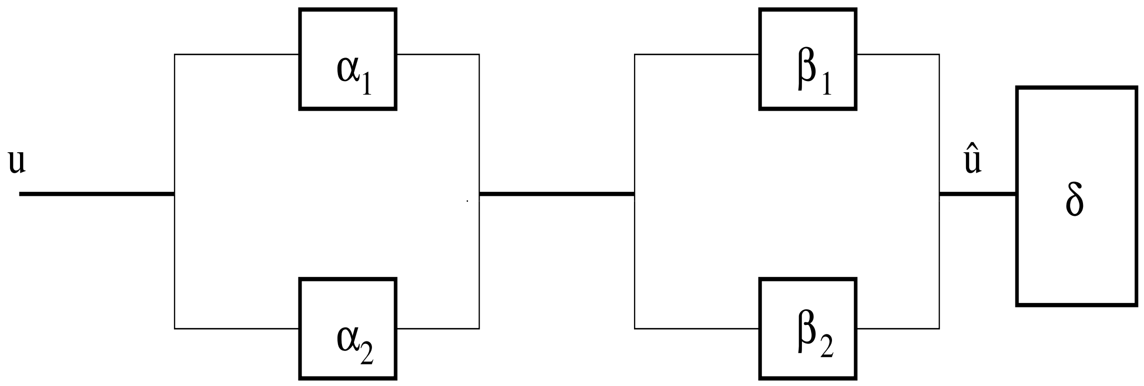

Figure 3.1: Distributed classification system. The dimension of the classifier is greater than the dimension of the encoders/decoders. The goal of the encoders is to quantize the data so that it has least effect on classification performance.

To answer this, first let us consider the simpler case where the dimension of the encoder and the classifier is the same, i.e., the encoder operates on the entire vector being used for classification. If an N-dimensional Vector Quantizer (VQ) was used to quantize each vector then the sum of distortion and Bayes risk can be used as a cost function to design the encoder [43, 42, 40]. The cost function can be written as

$$J = (\mathbf{x} \ - \ \beta(\alpha(\mathbf{x})))^2 \ + \ \lambda \sum_{k=0}^{M-1} \sum_{j=0}^{M-1} C_{jk} 1(\delta(\beta(\alpha(\mathbf{x}))) = k) P(Y = j | \mathbf{X} = \mathbf{x}) \qquad (3.1)$$

where $\alpha(\cdot)$ is the encoder, $\beta(\cdot)$ is the decoder, the classifier $\delta(\cdot)$ classifies the vector into one of M classes $\in Y$, λ is the Lagrange multiplier used to control the trade-off between classification and distortion performance, C_{jk} is a nonnegative number which represents the cost of classifying $\mathbf{x}$ as class k when the true class if j and $1(\textit{expression})$ is 1 if *expression* is true and 0 otherwise. The VQ so designed (Bayes VQ), to minimize the cost function J minimizes both MSE and Bayes risk. This design procedure ensures that the Voronoi regions of the VQ approximate the classification partitions of the classifier.

The same design procedure cannot be used without modification for the case when the sub-vectors of the vector $\mathbf{x}$ are encoded separately. The above design procedure implicitly relies on the fact that the classifier and encoder dimensions are the same. This can be observed by observing that the output of the decoder being designed becomes an input to the classifier. If we want to design an encoder for $\mathbf{x_1}$ i.e., a sub-vector of the vector $\mathbf{x}$ being used for classification, the Bayes risk component of the cost function J cannot be defined because Bayes risk incorporates the classifier $\delta(\cdot)$ in its definition and the classifier cannot produce an output on a sub-vector, i.e., $\delta(\mathbf{x_1})$ has no meaning.

Bayes VQ outputs, for each input, a quantization index that can be decoded to a reproduction level and a classification label, i.e., Bayes VQ is actually jointly designing a classifier and an encoder. In some systems (for example distributed speech/speaker recognition), we do not have control over the classifier while designing the encoder. Here we need to design an encoder that minimizes distortion and classification error given a fixed classifier. From Equation (3.1) it can seen that Bayes VQ requires knowledge of the probability density function (*pdf*) (either true *pdf* or posterior estimated *pdf*).

In this chapter we will show how the Bayes VQ design can be modified to take into account that there is a dimension mismatch between the encoders and the classifier. We introduce a new technique that jointly designs the encoders for the different sub-dimensions of the source data given a fixed classifier. By exploiting the fact that the classifier is already designed we will show that an empirical version of the design can be obtained. The main difficulty in this design problem is that while RD performance can be separately optimized in each of the component quantizers (operating on the sub-vectors) the actual misclassification cost depends on the complete vector. The main novelty of our method is that we show how information from different sub-dimensions can be combined to derive a misclassification cost which can be used in the design of the encoders. We will show that joint design of the encoders, by making use of information from other sub-dimensions while designing an encoder in a sub-dimension, can achieve improvements in performance over independent design of the encoders, where each of the encoders was designed to only minimize distortion.

In this chapter and the next chapter vectors are represented in bold and random variables are represented by capital letters. X = x implies that the random variable X takes on the value x. A vector $\mathbf{X}$ which consists of P sub-vectors is written as $\mathbf{X} = [\mathbf{X_1}, \mathbf{X_2}, ...\mathbf{X_P}]$, where each of the $\mathbf{X_i}s$ has dimension N_i and $\sum_i N_i = N$. We assume that the classifier classifies the data into one of M classes $Y \in \{0, 1, ..., M-1\}$. In addition we assume that C_{jk} the cost of classifying $\mathbf{x}$ as class k when the true class if j is

$$C_{jk} = \begin{cases} 0 & \text{if } j = k \\ 1 & \text{if } j \neq k \end{cases}$$

i.e., we assume that every class is equally important and incorrect decisions have equal penalty. This implies that Bayes risk is simply the probability of misclassification.

The rest of the chapter is organized as follows, Section 3.2 provides details of the Joint Product VQ design. In Section 3.3 the experiments and results are presented. Conclusions are presented in Section 3.4.

3.2 Joint Product VQ design using classification information

After briefly reviewing related previous work we start our discussion by considering a parametric algorithm where it is assumed that knowledge of the source *pdf* is available (Section 3.2.2). For situations where knowledge of the *pdf* is not available an empirical algorithm is derived from the parametric algorithm. The empirical algorithm is presented in Section 3.2.3.

3.2.1 Previous work

Consider an N dimensional classifier which uses vectors for classification. The classifier uses the entire vector to make its decision. A typical example would be in image classification where the input image is divided into $RbyR$ blocks and each block is classified (into one of two or more classes). Here the classifier dimension is $N = RbyR$. If an N dimensional Vector Quantizer (VQ) was used to quantize each block then the sum of distortion and Bayes risk [43, 42, 40] can be used as a cost function to design the encoder. The cost function can be written as (from equation (3.1) with Bayes risk replaced with probability of misclassification)

$$J = (\mathbf{x} \ - \ \beta(\alpha(\mathbf{x})))^2 \ + \ \lambda \sum_{j=0}^{M-1} 1(\delta(\beta(\alpha(\mathbf{x}))) \neq j) P(Y = j | \mathbf{X} = \mathbf{x}) \tag{3.2}$$

Good performance is achieved by this method, and in addition it provides a capability to trade-off between classification and distortion performance. Since a VQ is used for encoding the entire vector it suffers from the drawback of high complexity encoding, but Bayes VQ provides an upper bound on the joint distortion/classification performance for a given rate, which is useful for benchmarking the performance of other methods.

Design of a fast classification technique [2, 3] has also been reported. The main goal of this technique is to achieve fast classification of document images on the web for information retrieval. This technique involves performing all computations off-line and using table lookup for classification. In addition the use of a weighted distortion measure based on wavelet coefficients of the original image is shown to provide good improvements in performance. This technique is targeted for situations where only classification is required and compression is not an issue. The lookup tables at level N are generated by using the product of the VQs at level $N-1$. The lookup table design involves assigning every codeword combination from level $N-1$ VQs to the closest codeword at level N. As is clear this design involves product VQs and is aiming at minimizing classification error, which is similar to the problem we are trying to solve (in addition we also try to minimize distortion and compress the data). So by incorporating the joint design of the product VQs (presented in Section 3.2.2 and Section 3.2.3) into the design of the lookup tables we can potentially improve the classification performance achieved by the fast classification technique. This is discussed in detail in the next chapter.

3.2.2 Parametric design

Our goal is to build less complex encoders each of which operate on sub-vectors of the original N dimensional vector. This reduction in complexity should be achieved while having the least effect on classification performance. Let the vector we want to classify $\mathbf{X} = \mathbf{x}$, be an N dimensional vector. Let $\mathbf{X} = [\mathbf{X_1}, \mathbf{X_2}]$ where both $\mathbf{X_1}$ and $\mathbf{X_2}$ are vectors of dimension N/2. When we use product VQs to encode each of the sub-vectors $\mathbf{X_1}$ and $\mathbf{X_2}$ independently the encoder operates only on a part of the total vector used for classification, and it is not straightforward to compute the misclassification cost (the second term of the cost function J in equation (3.2). This is because the classifier operates on an N-dimensional vector and thus unquantized vectors with the same value in one sub-vector could be classified into different classes based on the value of the vector in the other sub-dimensions. For example consider the vectors $\mathbf{x^1} = [\mathbf{x^a}, \mathbf{x^b}]$ which belongs to class $1 \in Y$ and $\mathbf{x^2} = [\mathbf{x^a}, \mathbf{x^c}]$ which belongs to class $2 \in Y$. This is shown in Figure 3.2. Although both vectors $\mathbf{x^1}$ and $\mathbf{x^2}$ have the same sub-vector $\mathbf{x^a}$ in the 1^{st} dimension they belong to different classes because they differ in the 2^{nd} sub-dimension.

In order to build encoders which operate on sub-vectors of the entire vector which is used for classification, it is necessary to define a cost function which plays a role similar to the Bayes risk component in equation (3.2). Consider quantizing $\mathbf{X_1}$ along the 1^{st}

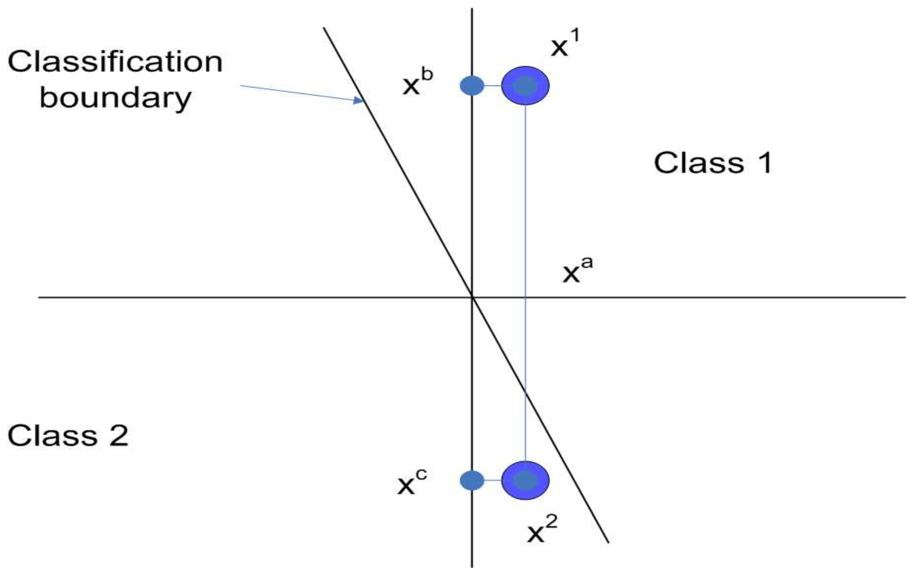

Figure 3.2: Even though the vectors x^1 and x^2 have the same value in one dimension, they belong to different classes as they significantly differ in the other dimension.

sub-dimension of $\mathbf{X}$. We require the misclassification cost for $\mathbf{X_1} = \mathbf{x_1}$. Since the class decisions are based on the entire vector $\mathbf{X}$ we cannot find the true misclassification cost using only the sub-vector $\mathbf{X_1}$, however we can consider using an "*average misclassification cost*". Since we need to estimate the misclassification cost for $\mathbf{X_1} = \mathbf{x_1}$ we can average $P(Y = j|\mathbf{X} = \mathbf{x})$, for the M classes in Y, over the sub-vector $\mathbf{X_2}$ when $\mathbf{X_1} = \mathbf{x_1}$ (see eq 3.9). This gives the probability of class j occurring when we only have knowledge of the sub-vector $\mathbf{X_1}$. This can be used to replace the Bayes risk component in equation (3.2). Let $\beta_1(\cdot)$ be the decoder in sub-dimension 1, $\beta_2(\cdot)$ be the decoder in sub-dimension 2, $\alpha_1(\cdot)$ be the encoder in sub-dimension 1, and $\alpha_2(\cdot)$ be the encoder in sub-dimension 2.

Then the distortion incurred when we quantize $\mathbf{X} = [\mathbf{x_1}, \mathbf{x_2}]$ along the 1^{st} sub-dimension is $(\mathbf{x_1} - \beta_1(\alpha_1(\mathbf{x_1})))^2$ and the total cost due to quantization is

$$\begin{aligned} J &= (\mathbf{x_1} \ - \ \beta_1(\alpha_1(\mathbf{x_1})))^2 \ + \\ & \lambda \sum_{j=0}^{M-1} 1(\delta(\beta_1(\alpha_1(\mathbf{x_1})), \beta_2(\alpha_2(\mathbf{x_2}))) \neq j) P(Y = j | \mathbf{X_1} = \mathbf{x_1}) \end{aligned} \tag{3.3}$$

where $\delta(\cdot,\cdot)$ is the same classifier as before but has been explicitly shown to operate on two variables to emphasize the fact that the vector used for classification contains 2 sub-dimensions. Specifically the cost of quantizing the $\mathbf{X} = [\mathbf{x_1}, \mathbf{x_2}]$ to the i^{th} codeword along the 1^{st} sub-dimension is

$$J = (\mathbf{x_1} \ - \ \beta_1(i))^2 \ + \ \lambda \sum_{j=0}^{M-1} 1(\delta(\beta_1(i), \beta_2(\alpha_2(\mathbf{x_2}))) \neq j) P(Y = j | \mathbf{X_1} = \mathbf{x_1}) \tag{3.4}$$

Thus the optimal encoder given this cost function will be

$$\begin{aligned} \alpha_1(\mathbf{x_1}) &= \arg\min_i [(\mathbf{x_1} - \beta_1(i))^2 \ + \\ & \lambda \sum_{j=0}^{M-1} 1(\delta(\beta_1(i), \beta_2(\alpha_2(\mathbf{x_2}))) \ \neq \ j) P(Y = j | \mathbf{X_1} = \mathbf{x_1}),] \end{aligned} \tag{3.5}$$

and correspondingly the Lloyd decoder will be

$$\beta_1 = \arg\min_z E[d(\mathbf{X_1}, \mathbf{z} | \alpha_1(\mathbf{X_1})),] \tag{3.6}$$

Which minimizes the distortion independently of the classification.

Similarly, the encoder and decoder for the $2nd$ sub-dimension are given by

$$\begin{aligned} \alpha_2(\mathbf{x_2}) &= \arg\min_i [(\mathbf{x_2} - \beta_2(i))^2 \ + \\ & \lambda \sum_{j=0}^{M-1} 1(\delta(\beta_1(\alpha_1(\mathbf{x_1})), \beta_2(i)) \ \neq \ j) P(Y = j | \mathbf{X_2} = \mathbf{x_2})] \end{aligned} \tag{3.7}$$

and

$$\beta_2 = \arg\min_z E[d(\mathbf{X_2}, \mathbf{z}|\alpha_2(\mathbf{X_2}))] \tag{3.8}$$

The main difference between the above design and Bayes VQ is that both $\alpha_1(\cdot)$ and $\alpha_2(\cdot)$ have an effect on classification performance but have to applied and designed separately.

As is observed form equation (3.5) and equation (3.7), we require $P(Y = j|\mathbf{X_1} = \mathbf{x_1})$ and $P(Y = j|\mathbf{X_2} = \mathbf{x_2})$ to design the encoders. We will show below how $P(Y = j|\mathbf{X_1} = \mathbf{x_1})$ can be evaluated as follows ($P(Y = j|\mathbf{X_2} = \mathbf{x_2})$ can be evaluated in a similar manner).

$$P(Y = j|\mathbf{X_1} = \mathbf{x_1}) = \int_{\mathbf{x_2}} P(Y = j|\mathbf{X} = \mathbf{x}) f(\mathbf{x_2}|\mathbf{x_1}) d\mathbf{x_2} \tag{3.9}$$

where $f(\mathbf{x_2}|\mathbf{x_1})$ is the conditional probability of $\mathbf{X_2} = \mathbf{x_2}$ given $\mathbf{X_1} = \mathbf{x_1}$, and the integration is a volume integral over the $N/2$-dimensional space of $\mathbf{x_2}$.

$$P(Y = j|\mathbf{X_1} = \mathbf{x_1}) = \int_{\mathbf{x_2}} \frac{P(Y = j \;\&\; \mathbf{X} = \mathbf{x})}{f(\mathbf{x_1})} d\mathbf{x_2} \tag{3.10}$$

$$P(Y = j \;\&\; \mathbf{X} = \mathbf{x}) = \begin{cases} P(\mathbf{X} = \mathbf{x}) & \text{if } \delta(\mathbf{x_1}, \mathbf{x_2}) = j \\ 0 & \text{otherwise} \end{cases} \tag{3.11}$$

So,

$$P(Y = j|\mathbf{X_1} = \mathbf{x_1}) = \int_{\mathbf{x_2}:\delta(\mathbf{x_1},\mathbf{x_2})=j} f(\mathbf{x_2}|\mathbf{x_1}) d\mathbf{x_2} \tag{3.12}$$

The above expression states that $P(Y = j|\mathbf{X_1} = \mathbf{x_1})$ is the probability of $\mathbf{X}$ satisfying $\delta(\mathbf{x_1}, \mathbf{x_2}) = j$ given $\mathbf{X_1} = \mathbf{x_1}$, which is intuitively satisfying.

Extension to the case when there are more then 2 sub-dimensions (say P sub-dimensions) in the N dimensional vector $\mathbf{X}$ is straightforward. If $\mathbf{X} = [\mathbf{X_1}, \mathbf{X_2}, ...\mathbf{X_P}]$ it can be shown that

$$P(Y = j|\mathbf{X_1} = \mathbf{x_1}) = \int\limits_{\mathbf{x_2},...,\mathbf{x_P}:\delta(\mathbf{x_1},...,\mathbf{x_P})=j} f(\mathbf{x_2}, ..., \mathbf{x_P}|\mathbf{x_1}) d\mathbf{x_2}...d\mathbf{x_P} \tag{3.13}$$

where $f(\mathbf{x_2}, ..., \mathbf{x_P}|\mathbf{x_1})$ is the conditional probability of $\mathbf{X_2} = \mathbf{x_2}\&...\&\mathbf{X_P} = \mathbf{x_P}$ given $\mathbf{X_1} = \mathbf{x_1}$ and $\delta(\cdot, \quad , \cdot)$ is an N dimensional classifier.

The generalized Lloyd algorithm (GLA) is used to design the encoder and decoder in each sub-dimension. Note that the classification cost in Equation (3.4) for $\mathbf{X_1}$ is dependent on the encoder and decoder for $\mathbf{X_2}$. This implies that the encoder design in each dimension is not independent of encoder/decoder design in the other dimension. Thus we require an iterative procedure where the encoder for $\mathbf{X_i}$ is optimized while encoders for all the $\mathbf{X_j}, j \neq i$, are fixed This results in an iterative design summarized as

Algorithm 5 *(Iterative quantizer design)*

Step 1 : *Initialize the encoders for sub-dimension 1 and 2.*

Step 2 : *Design the encoder for sub-dimension 1 using equation (3.5).*

Step 3 : *Design the decoder for sub-dimension 1 using equation (3.6).*

Step 4 : *If the codewords changed go to* **Step 2** *else go to* **Step 5***.*

Step 5 : *Design the encoder for sub-dimension 2 using equation (3.7).*

Step 6 : *Design the decoder for sub-dimension 2 using equation (3.8).*

Step 7 : *If the codewords changed go to* **Step 5** *else go to* **Step 8***.*

Step 8 : *If the codewords in either sub-dimension changed go to* **Step 2** *else* **stop***.*

Note that this iterative procedure is only required during the design of the encoders and the actual encoding can be done independently in each dimension without any iterations. Since one of the requirements is to keep the encoding complexity low, nearest neighbor encoding is used during actual encoding i.e.,

$$\alpha_j(\mathbf{x_j}) = \arg\min_i [(\mathbf{x_j} - \beta_j(i))^2] \tag{3.14}$$

for $j = 1, 2$. However if an approximation to the final classifier was simple (say a table lookup) then we can use the classification information during actual encoding to get better results, i.e., the actual encoder would be the same as the encoder used during design (equations (3.5) and (3.7)).

3.2.3 Empirical design

The algorithm developed in Section 3.2.2 required knowledge of the conditional *pdfs* of the sub-vectors. When the underlying *pdf* of the source is known then the conditional *pdfs* can easily be obtained and used in the design. However when we are dealing with vectors of high dimensionality obtained from real world sources the *pdfs* in general are unknown. One solution would be to model the source using Gaussian mixture (or some other) *pdf*[44]. However if the modeling is not very accurate the designed quantizers can be suboptimal and the performance of the encoders can be very poor. To overcome this shortcoming we propose an empirical version of the design algorithm. The goal here is to make use of the information in the training set to simulate the effect of the conditional *pdfs*. Obviously since the conditional *pdfs* are approximated using the training set, mismatch between the training set data and test set data can adversely affect the performance of the encoders. To obtain an empirical algorithm we need to estimate or approximate the probabilities using the training set. One method of approximating the probabilities is by the use of high rate VQs [71, 43]. To motivate the design for N dimensional data we first consider the simple case of 2 dimensional data, where the classifier operates on the 2 dimensional vector and each component of the vector is independently scalar quantized. Let the scalar in dimension 1 lie in the interval $[X_l, X_h]$. We want to quantize this scalar X_1 using N quantization values $Q^1_i,\ i = 0, 1, ...N-1$. The simplest technique to aid estimation of the probabilities would be to use uniform scalar quantization, where the real line between X_l and X_h is partitioned into small bins each of size Δ (Figure 3.3). To achieve good performance we need fine quantization, so $K^1 = \frac{X_h - X_l}{\Delta} \gg N$, however Δ should not be made very small because then the conditional *pdfs* will not be approximated properly by the bins, as it would result in very sparse data and thus reliable probability estimates may not be obtained. Let $b^1_j,\ \ j = 0, 1, ...K^1 - 1$ be the bins of the real line between X_l and X_h with each bin size being Δ. Now instead of quantizing every point on the real line between X_l and X_h to one of the quantization values Q^1_i, we can quantize the bins b^1_j to one of the quantization values Q^1_i. This will result in sub-optimality of the final quantizer (however if Δ is small the error introduced is negligible).

The distortion cost can be approximated as

$$\hat{d}(Q^1_i, b^1_j) = (Q^1_i - c^1_j)^2 \tag{3.15}$$

where c^1_j is the mid point of bin b^1_j.

The average misclassification cost $C(Q^1_i, b^1_j)$ can be written as

$$C(Q^1_i, b^1_j) = \sum_{j=0}^{M-1} 1(\delta(Q^1_i, \beta_2(\alpha_2(\mathbf{x_2}))) \neq j) \int_{\mathbf{x_2}} P(Y = j|\mathbf{X} = \mathbf{x}) f(\mathbf{x_2}|\mathbf{x_1}) d\mathbf{x_2} \tag{3.16}$$

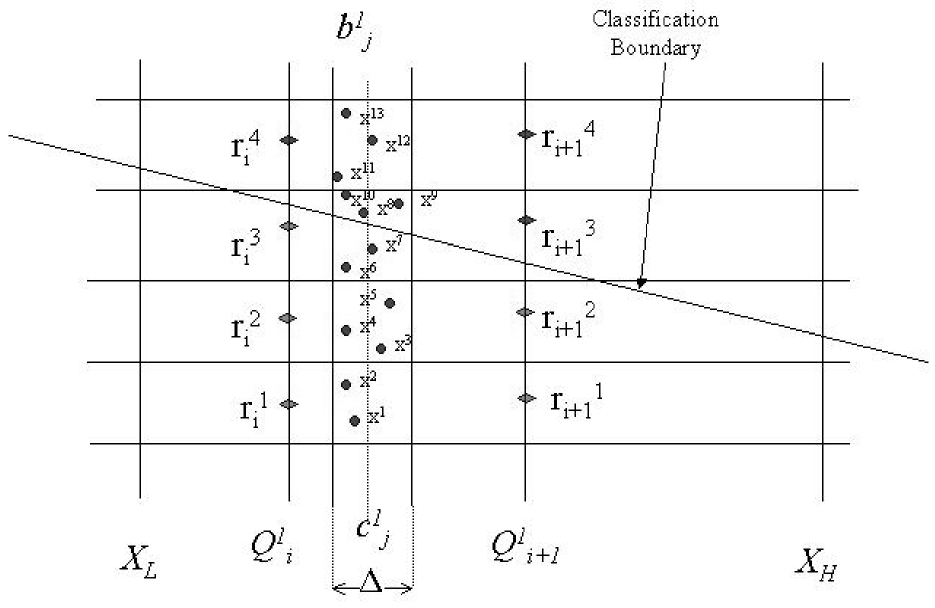

Figure 3.3: Illustration of the empirical algorithm (only one of the K^1 bins is shown). Bin b_j^1 is assigned to the quantization value Q_j^1 so that the joint cost of distortion and average misclassification is minimized. The width of the bin is Δ and its mid-point is c_j^1. In the example shown there are 13 source vectors in the bin b_j^1. The diamond points (r_i^k and r_{i+1}^k) are the 2 dimensional reproductions points got by associating the reproduction points Q_i^1 and Q_{i+1}^1 in the 1^{st} dimension with every reproduction point in the 2^{nd} dimension. The class of the reproductions points and the source vectors is determined by the region they fall into, i.e., the points below the classification line belong to class 0 and those above belong to class 1.

from equations (3.4) and (3.9). Using the bins got by scalar quantization this can be approximated as

$$\hat{C}(Q_i^1, b_j^1) \;=\; \frac{\sum_{l=1}^{n_j^1} 1(\delta(Q_i^1, \beta_2(\alpha_2(x_2^l))) \;\neq\; \delta(x_1^l, x_2^l))}{n_j^1} \tag{3.17}$$

Where n_j^1 is the number of vectors in bin b_j^1 and $\mathbf{x^l} = [x_1^l, x_2^l]$ $l = 1, 2, ...n_j^1$ is the subset of the training set such that x_1 is assigned to the bin b_j^1. The numerator of equation (3.17) gives the count of the number of vectors in b_j^1 which are misclassified if we assign b_j^1 to Q_i^1, dividing this by the total number of vectors in b_j^1 gives the classification error incurred in assigning b_j^1 to Q_i^1. Notice that $\hat{C}(Q_i^1, b_j^1)$ can be evaluated very efficiently. All that is required is to identify the vectors $\mathbf{x}$ in b_j^1 whose class label is different from $\delta(Q_i^1, \beta_2(\alpha_2(x_2^l)))$ i.e. the class of the codeword combination that $\mathbf{x}$ is assigned to after quantization. Now the cost function will be

$$\hat{J} = \hat{d}(Q_i^1, b_j^1) \;+\; \lambda\, \hat{C}(Q_i^1, b_j^1) \tag{3.18}$$

3.2.3.1 Bin assignment example during product VQ design

An example is shown in Figure 3.3 which shows one of the bins b_j^1 which is used to design the encoder. The cost of assigning b_j^1 to Q_i^1 is

$$\hat{J} = (Q_i^1 - c_j^1)^2 \;+\; \lambda\, \frac{3}{13} \tag{3.19}$$

since vectors $\mathbf{x_8}$, $\mathbf{x_9}$, and $\mathbf{x_{10}}$ do not belong to the same class as the reproduction point $\mathbf{r_i^3}$.

Similarly the cost of assigning b_j^1 to Q_{i+1}^1 is

$$\hat{J} = (Q_{i+1}^1 - c_j^1)^2 \;+\; \lambda\, \frac{2}{13} \tag{3.20}$$

since vectors $\mathbf{x_6}$, and $\mathbf{x_7}$ do not belong to the same class as the reproduction point $\mathbf{r^3_{i+1}}$. This modified cost function found by the empirical algorithm is used to design the scalar quantizers.

3.2.3.2 Empirical product VQ design

For all the input vectors $[X_1, X_2]$ for which $X_1 = x_1$ lies in bin b^1_j (whose center is c^1_j), the encoder is

$$\alpha_1(x_1) = \arg\min_i [(c^1_j - \beta_1(i))^2 \ + \ \lambda \frac{\sum_{l=1}^{n^1_j} 1(\delta(Q^1_i, \beta_2(\alpha_2(x^l_2))) \ \neq \ \delta(x^l_1, x^l_2))}{n^1_j}] \quad (3.21)$$

The Lloyd decoder does not use conditional probabilities so it is the same as before, i.e. $\beta_1 = \arg\min_y E[d(X_1, y|\alpha_1(X_1))]$. As before the encoders need to be designed iteratively until convergence.

For the simple case considered above we showed that using uniform scalar quantization would allow us to approximate the probabilities and leads to an empirical algorithm. However this technique does not easily generalize to higher dimensions where the vector to be classified is an N dimensional vector consisting of 2 or more sub-vectors. For the scalar case instead of using uniform quantization we can consider designing a high rate scalar quantizer (using GLA) for each dimension. As before the high rate quantizer in sub-dimension 1 partitions the real line between X_l and X_h into bins $b^1_j, \ j = 0, 1, ...K^1 - 1$. Unlike before, the size of the different bins may not be the same, but that is not required for our algorithm. The partitions are now decided by the position of the reproduction values $c^1_j, \ j = 0, 1, ...K^1 - 1$. The bins induced by this partitioning can be used as before to find the distortion and average misclassification costs.

This modified partitioning of the training set can be extended to higher dimensions without loss of generality. If the input to be classified $\mathbf{X}$ is an N dimensional vector made up of two sub-vectors $\mathbf{X_1}$ and $\mathbf{X_2}$, each vectors of dimension N/2, then we can design N/2 dimensional VQs for both sub-vectors $\mathbf{X_1}$ and $\mathbf{X_2}$. To design the encoder along sub-dimension 1, we use the N/2 dimensional VQ designed for the 1^{st} sub-dimension to partition the space spanned by $\mathbf{X_1}$ into K^1 Voronoi regions $b_j^1, \;\; j = 0, 1, ...K^1 - 1$. As before we want to encode the sub-vector $\mathbf{X_1}$ using N reproduction values $\mathbf{Q_i^1}, \; i = 0, 1, ...N - 1$. Each of the Voronoi regions b_j^1 is assigned to the reproduction value $\mathbf{Q_i^1}$ such that the total cost incurred is minimized. The distortion cost can be found as

$$\hat{d}(\mathbf{Q_i^1}, b_j^1) = (\mathbf{Q_i^1} - \mathbf{c_j^1})^2 \tag{3.22}$$

where $\mathbf{c_j^1}$ is the centroid of Voronoi region b_j^1.

If n_j^1 is the number of vectors in Voronoi region b_j^1, then $C(\mathbf{Q_i^1}, b_j^1)$ can be approximated as

$$\hat{C}(\mathbf{Q_i^1}, b_j^1) \;=\; \frac{\sum_{l=1}^{n_j^1} 1(\delta(\mathbf{Q_i^1}, \beta_2(\alpha_2(\mathbf{x_2^l}))) \;\neq\; \delta(\mathbf{x_1^l}, \mathbf{x_2^l}))}{n_j^1} \tag{3.23}$$

where $\mathbf{x^l} = [\mathbf{x_1^l}, \mathbf{x_2^l}] \;\; l = 1, 2, ...n_j^1$ is the subset of the training set such that $\mathbf{x_1}$ is assigned to the Voronoi region b_j^1. The total cost when Voronoi region b_j^1 is assigned to $\mathbf{Q_i^1}$ is

$$\hat{J} = (\mathbf{Q_i^1} - \mathbf{c_j^1})^2 \;+\; \lambda \; \frac{\sum_{l=1}^{n_j^1} 1(\delta(\mathbf{Q_i^1}, \beta_2(\alpha_2(\mathbf{x_2^l}))) \;\neq\; \delta(\mathbf{x_1^l}, \mathbf{x_2^l}))}{n_j^1} \tag{3.24}$$

After assigning all the Voronoi regions b_j^1 to the N reproduction values $\mathbf{Q_i^1}$, the Voronoi region associated to a reproduction level $\mathbf{Q_k^1}$ will be the merger of the Voronoi regions b_j^1 which are assigned to the reproduction level $\mathbf{Q_k^1}$.

Similarly the total cost of assigning the Voronoi regions $b_j^2,\ \ j = 0, 1, ...K^2 - 1$ in the 2nd dimension to the codeword $\mathbf{Q_i^2}$ is

$$\hat{J} = (\mathbf{Q_i^2} - \mathbf{c_j^2})^2 \ + \ \lambda \ \frac{\sum_{l=1}^{n_j^2} 1(\delta(\beta_1(\alpha_1(\mathbf{\tilde{x}_1^l})), \mathbf{Q_i^2}) \ \neq \ \delta(\mathbf{\tilde{x}_1^l}, \mathbf{\tilde{x}_2^l}))}{n_j^2} \tag{3.25}$$

where $\mathbf{Q_i^2}\ \ i = 0, 1, ..., N - 1$ are the reproduction values along the 2nd sub-dimension, $\mathbf{\tilde{x}^l} = [\mathbf{\tilde{x}_1^l}, \mathbf{\tilde{x}_2^l}]\ \ l = 1, 2, ..., n_j^2$ is the subset of the training set such that $\mathbf{\tilde{x}_2}$ is assigned to the Voronoi region b_j^2 and c_j^2 is the centroid of the Voronoi region b_j^2.

Algorithm 6 *(Empirical product VQ design)*

Step 1 : *Design a high rate VQ with codebook size = K^1 and K^2 for sub-dimension 1 and 2 respectively.*

Step 2 : *For $j = 0, 1, ..., K^1 - 1$ assign Voronoi region b_j^1 to the codeword Q_i^1 such that the cost (equation (3.24)) is minimized.*

Step 3 : *Update the codewords Q_i^1 (as the centroid of the training set assigned to it).*

Step 4 : *If codewords changed go to* **Step 2** *else go to* **Step 5**.

Step 5 : *For $k = 0, 1, ..., K^2 - 1$ assign Voronoi region b_j^2 to the codeword Q_i^2 such that the cost (equation (3.25)) is minimized.*

Step 6 : *Update the codewords Q_i^2 (as the centroid of the training set assigned to it).*

Step 7 : *If codewords changed go to* **Step 5** *else go to* **Step 8**.

Step 8 : *If the codewords in either sub-dimension changed go to* **Step 2** *else* **stop**.

3.3 Experiments and Results

Experiments were carried out on three different examples.

- Case (i) 2 dimensional Gaussian mixture data, 2 dimensional Bayes classifier, each dimension of the vector was quantized independently by scalar quantizers. The Gaussian data was a mixture of 2 sources, each with variance 1, and means were [1,0.2] and [-1,-0.2].

- Case (ii) 4 dimensional Gaussian mixture data, 4 dimensional Bayes classifier, the 1^{st} and 2^{nd} dimensions were encoded together by a 2 dimensional VQ and the 3^{rd} and 4^{th} dimension were encoded by another 2 dimensional VQ. The Gaussian data was a mixture of 2 sources, each with variance 1, and means were [1,0.2,0.2,0.2] and [-1,-0.2,-0.2,-0.2].

- Case (iii) 2 dimensional Gaussian mixture data, 2 dimensional Bayes classifier, each dimension of the vector was quantized independently by scalar quantizers. The Gaussian data was a mixture of 8 sources, each with variance 1, and means were [1,0.2],[-1,-0.2], [3,0], [-3,0], [3.4,-2], [-3.4,2], [5.4,-2.4] and [-5.4,2.4].

Figure 3.4 shows the experiment setup for the different examples considered.

For Case (i) two scalar quantizers were designed using the knowledge of the *pdfs*. For the same training data the empirical algorithm was also used to design 2 scalar quantizers. To compare the two partitioning techniques for the empirical algorithm, the partitioning was done with both a high rate uniform scalar quantizer and a high rate Lloyd-Max quantizer. Since one of the goals of the design was to keep the encoding complexity low, during encoding information about the *pdfs* was not used and instead MMSE encoding was used, where every input in each dimension is quantized to the closest reproduction point in the corresponding dimension. The optimal Bayes classifier for this data will assign an input

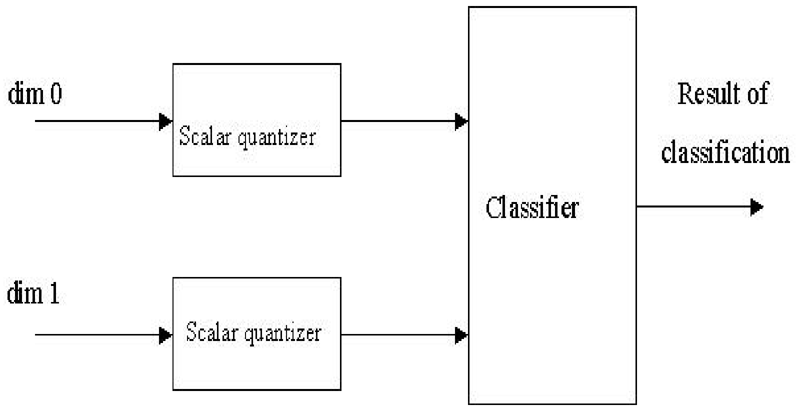

Figure 3.4: The setup of the Case (i) example. The input consists of a 2 dimensional vector, each dimension is quantized independently using scalar quantization, and the quantized outputs are used for classification by the 2 dimensional Bayes classifier. For Case (ii) the scalar quantizers are replaced by 2 dimensional VQs and the 2 dimensional classifier is replaced with a 4 dimensional Bayes classifier. For Case (iii) the 2 dimensional classifier is replaced with an 8 dimensional Bayes classifier.

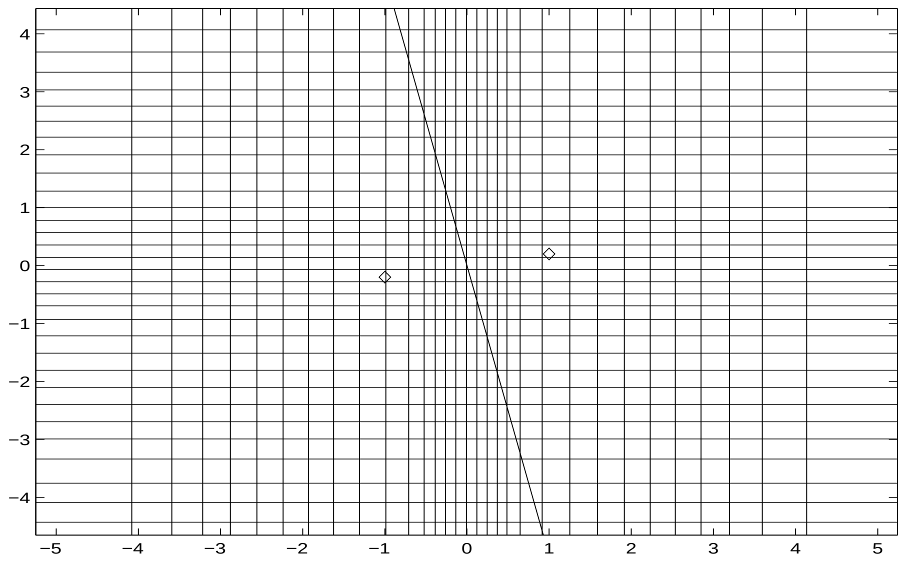

Figure 3.5: Quantization bins when the proposed algorithm is used to design the Scalar Quantizers. The means of the two classes are represented by diamond points. The diagonal line is the Bayes classifier boundary.

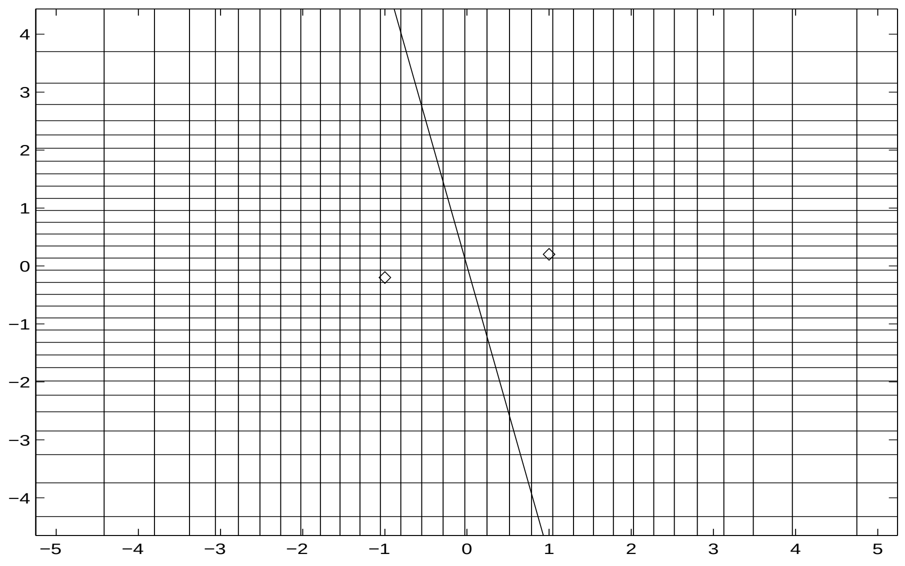

Figure 3.6: Quantization bins when the only distortion is minimized to design the scalar quantizers. The means of the two classes are represented by diamond points. The diagonal line is the Bayes classifier boundary.

vector to the class whose mean the input vector is closer to. The classification boundary for the Bayes classifier will be perpendicular to the line joining the mean of the two classes [56, 67] as shown in Figure 3.5. Figure 3.5 also shows the quantization bins obtained when we design scalar quantizers based on the proposed algorithm. Comparing this to Figure 3.6, which represents the quantization bins obtained when we design scalar quantizers to minimize only distortion we observe that for quantizers designed to minimize both classification error and distortion the quantization bins tend to get clustered together close to the classification boundary. This helps to reduce the classification error introduced due to quantization as now more bins are placed in regions which introduce classification error. The density of bins in near the classification boundary is controlled by the Lagrange multiplier. A small Lagrange multiplier (distortion is important) would mean the bins will be very similar to those obtained when we minimize only distortion.

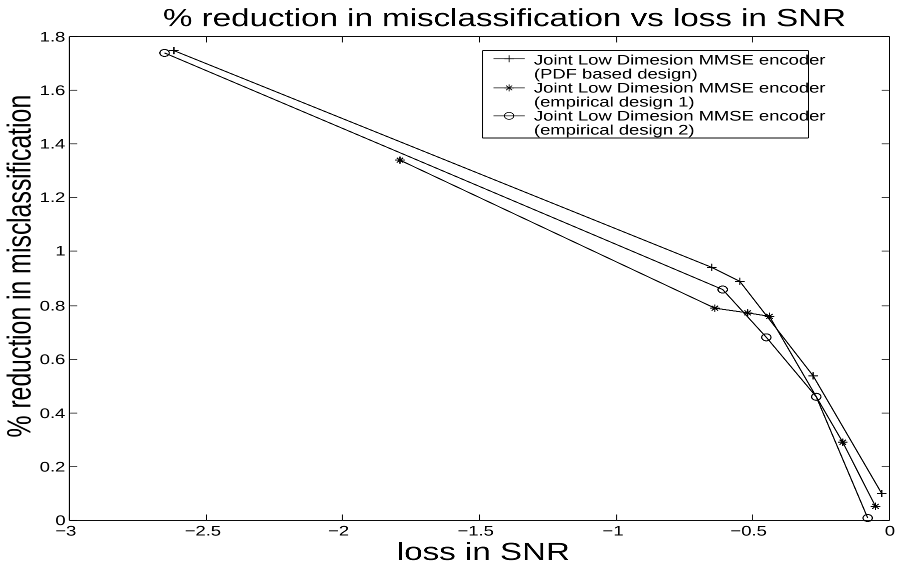

Figure 3.7: Trade-off between percentage reduction in misclassification and reduction in SNR for scalar quantizers designed for each dimension of a 2 dimensional Gaussian mixture vector (Case(i)). The top curve represents the *pdf* based design and the other two curves represent the empirical designs. Empirical design 1 corresponds to the case when high rate scalar quantizers are used to partition the training set, and empirical design 2 corresponds to the case when high rate Lloyd-Max scalar quantizers are used to partition the training set. The number of bins in empirical design 2 is lesser than the number of bins in empirical design 1, so the complexity of the design is lesser for empirical design 2. The percentage misclassification when the scalar quantizers are designed only to minimize distortion is 3.40%. So the reductions in misclassification are significant.

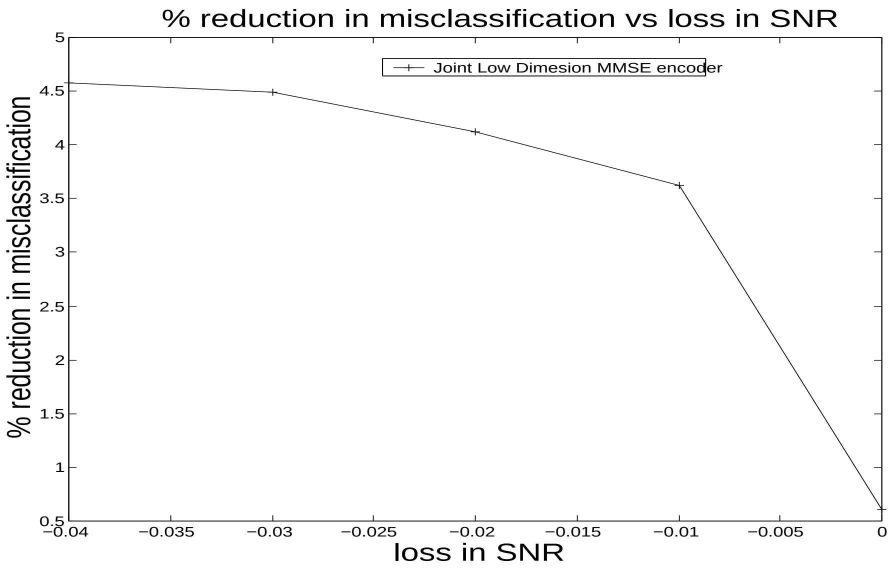

Figure 3.8: Trade-off between percentage reduction in misclassification and reduction in SNR for two 2 dimensional VQs designed for a 4 dimensional Gaussian mixture vector (Case(ii)). The percentage misclassification when the VQs are designed only to minimize distortion is 9.64%

A large Lagrange multiplier (classification error is important) would mean the classification boundary range will be densely packed with the quantization bins so as to minimize the misclassification (of course this will mean higher distortion). Figure 3.7 shows the reduction in misclassification percentage as a function of reduction in SNR when compared to the case that the quantizers were designed to minimize only distortion. It is observed that with a 0.7 dB decrease in SNR the misclassification is reduced by about 1%. The misclassification percentage when only distortion was minimized was 3.4% so substantial gain is obtained, if 2.5 dB loss in SNR is tolerable then the misclassification can be reduced in half. From the figure it can also be observed that the empirical algorithm performs almost as well as the *pdf* based algorithm proving the validity of the empirical algorithm. Comparing the results for the two different partitionings (uniform *vs* Lloyd-Max) it is clear that the performance of the Lloyd-Max partitioning is better than the uniform partitioning. This is to be expected because Lloyd-Max partitioning places most its bins in regions of high probability, so it capable of estimating the probability distribution more accurately. It should however be noted that in general the proposed method (parametric or empirical) cannot achieve 0 misclassification even if we ignore the distortion cost ($\lambda = \infty$), because when a 2 dimensional classification boundary is approximated with scalar quantizers perfect approximation is almost never possible (except if the classification boundary is horizontal or vertical).

For Case (ii) two 2 dimensional VQ were designed for the sub-vectors of the 4 dimensional source data using the knowledge of the *pdfs*. For comparison, a modified version of Bayes VQ (to account for the fixed classifier) was designed for the same training data. The results obtained after encoding the test set with the 2 dimensional VQs is shown in Figure 3.8. Again only MMSE encoding was used. The comparison between the proposed method and Bayes VQ can be seen in Figure 3.9. The top most point for each curve represents the SNR and misclassification when the quantizers were designed to only minimize distortion (i.e. $\lambda = 0$). Notice that the trade-off between SNR and misclassification is almost the same for both methods, however much lower misclassification can be achieved with Bayes VQ, since Bayes VQ can partition the space much more efficiently than when lower dimension product VQs are used. But the main gain achieved by the proposed algorithm over Bayes VQ is that the encoding operation is less complex, since each encoder operates on a sub-dimension of the vector independently unlike Bayes VQ which has to encode the entire vector.

The approximate time required for the two methods to encode 32000 4 dimensional vectors is shown in Table 3.1. Bayes VQ had a codebook of size 64 and the proposed

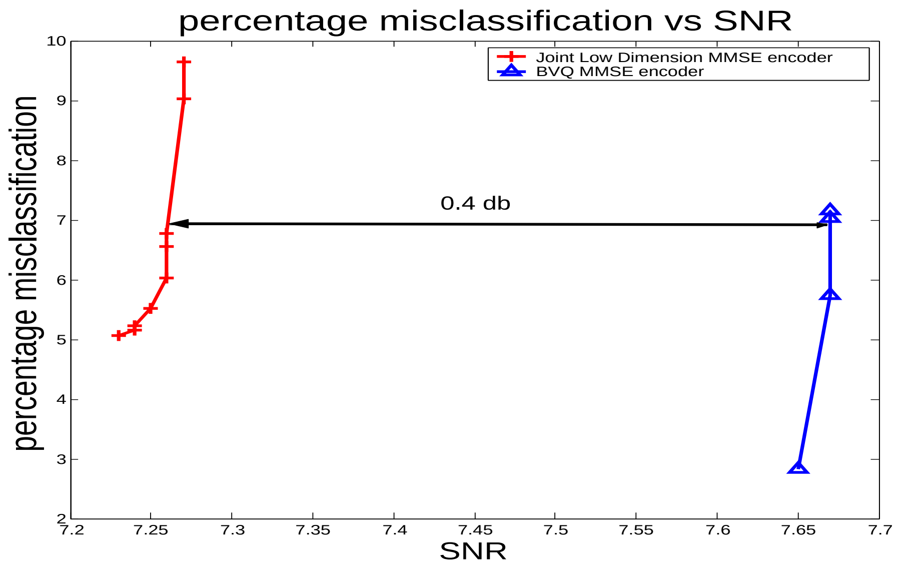

Figure 3.9: Percentage misclassification vs SNR for two 2 dimensional VQs and Bayes VQ designed for a 4 dimensional Gaussian mixture vector (Case(ii)). For both curves the points with maximum SNR correspond to the case when only distortion is minimized.

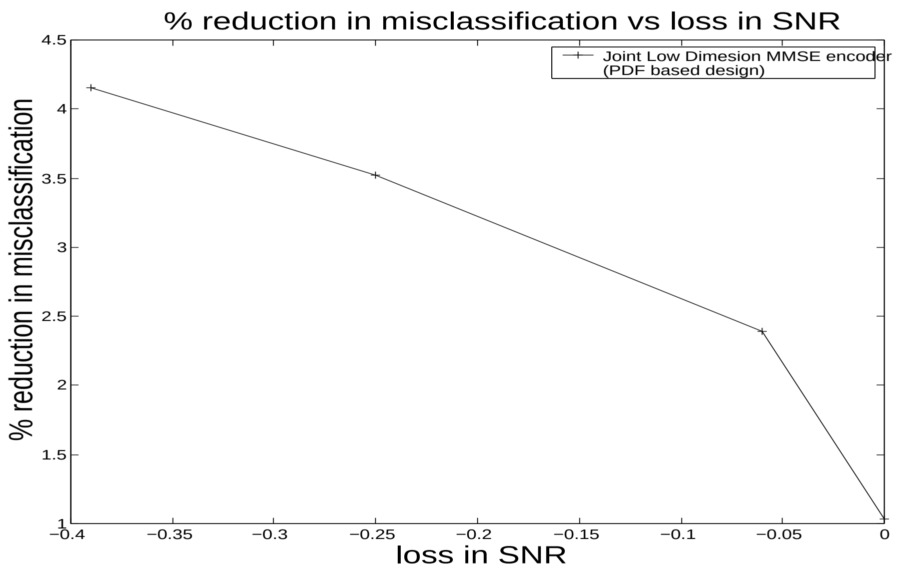

Figure 3.10: Trade-off between percentage reduction in misclassification and reduction in SNR for 2 Scalar Quantizers designed for a 2 dimensional 8 source Gaussian mixture vector (Case(iii)). The percentage misclassification when the scalar quantizers are designed only to minimize distortion is 13.99%

method had codebooks of size 8 in each sub-dimension. It is observed that by using encoders operating on lower dimensions we are able to reduce the complexity by more than a factor of 2. This difference in complexity will become more pronounced when the dimension of the input vector is much larger, in fact for very high dimensionality it may not be possible to use Bayes VQ.

Low dimension encoder	Bayes VQ
0.66 s	1.41 s

Table 3.1: CPU time (in seconds, on a Sun workstation) required to encode 32000 4 dimensional vectors. Bayes VQ has a codebook of size 64. Proposed low dimension encoder has codebooks of size 8 in both sub-dimensions. The difference in time can be expected to be much higher for larger dimension vectors.

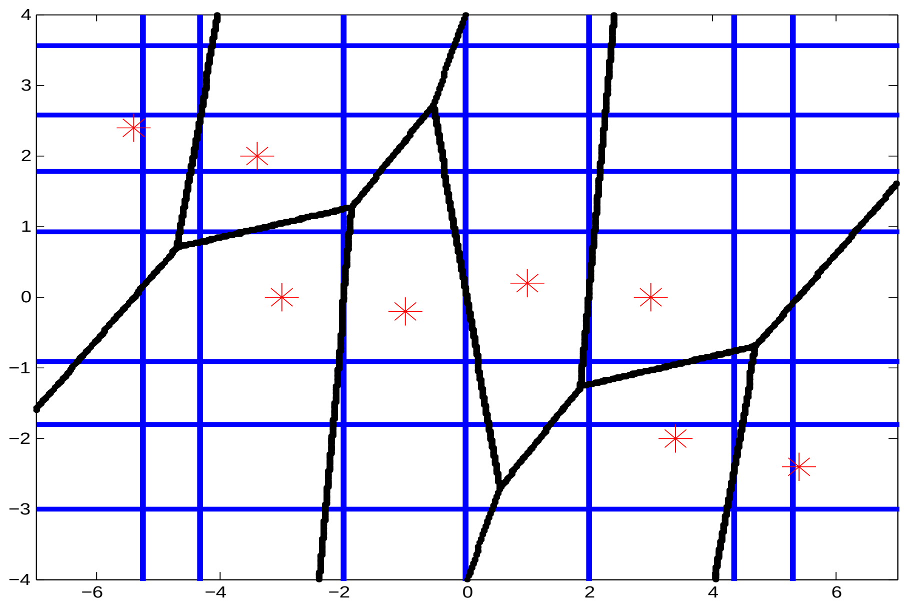

Figure 3.11: The optimal Bayes classifier (solid black line) and the quantization bins (thin horizontal and vertical lines) for Case(iii), i.e., a mixture of 8 sources. The stars represent the means of the 8 sources. Observe that the quantizations bins approximate the Euclidean partition of the classifier.

For Case (iii) two scalar quantizers were designed using the knowledge of the *pdfs*. Again MMSE encoding was used to encode the test data. Figure 3.11 shows the optimal Bayes classifier and the bins obtained for the scalar quantizer. As can observed from the figure, since there are eight classes the classification boundary is no longer simple as in the previous examples. However, since the variance of the different classes are still equal the Bayes classifier is the same as before i.e., an input vector is assigned to the class whose mean is closest to the input vector. Figure 3.10 shows the trade-off between SNR and misclassification for this example. The misclassification when only distortion was minimized was 13.99%. With less than 0.4 dB reduction in SNR the misclassification was reduced to 9.84%. This shows that the proposed algorithm works well even when the classification boundary is not trivial.

3.4 Conclusions

In this chapter, an algorithm was presented for Joint compression and classification, to handle the situation when the encoder dimension is lower than the classifier dimension. It was shown that using information from other dimensions enables better design of encoders to reduce the combined distortion and classification error cost. An empirical version of the algorithm was presented which did not rely on the knowledge of the source *pdf*. The proposed algorithm provided ability to trade-off distortion and classification performance. For situations where encoder complexity should be low, sub-dimension encoding provides an alternative to encoding the entire vector using a high dimension encoder, trading performance for reduction in complexity. In subsequent chapters the algorithm has been used to design efficient product VQs (Chapter 4) and to design speech encoders for distributed speech recognition applications (Chapter 5).

Chapter 4

Performance Improvements in Fast Table-Lookup Encoding for Image Compression Applications

4.1 Introduction and Motivation

Shannon's rate-distortion theorem gives the minimum achievable rate for a given distortion (pp. 346, [16]). To achieve the rate distortion limit encoders operating on large source blocks lengths have to be considered. VQs with high dimensions are one set of encoders which have been shown to have good performance in the rate distortion sense. Nearest neighbor VQ encoding identifies, in the VQ codebook, the codeword closest to the input vector to be encoded. For rate R bits/sample, the number of codewords (assuming no entropy coding) in the VQ codebook will be $M = 2^{NR}$, where N is the dimension of the VQ codebook. As can be observed the codebook size increases exponentially with both rate and dimension. Hence for high rates and dimensions the codebook size will become very large. This directly impacts the encoding time; for every input vector we require 2^{NR} distance comparisons in the N dimensional space. To overcome this problem structured VQs, which impose structural constraints on the codebook so as to reduce the encoding time, have been proposed. However most of these structured VQs suffer from the drawback of not always being able to identify the closest codeword for the given input vector. This results in higher mean square error (MSE) as compared to unconstrained VQ, which is undesirable. One such structured VQ is product VQ (PVQ), where the input vector is broken into smaller sub-vectors and each of these are independently encoded. Optimal PVQ aims at maximizing the rate-distortion (RD) performance, ideally approaching that of the higher dimensional unstructured VQ.

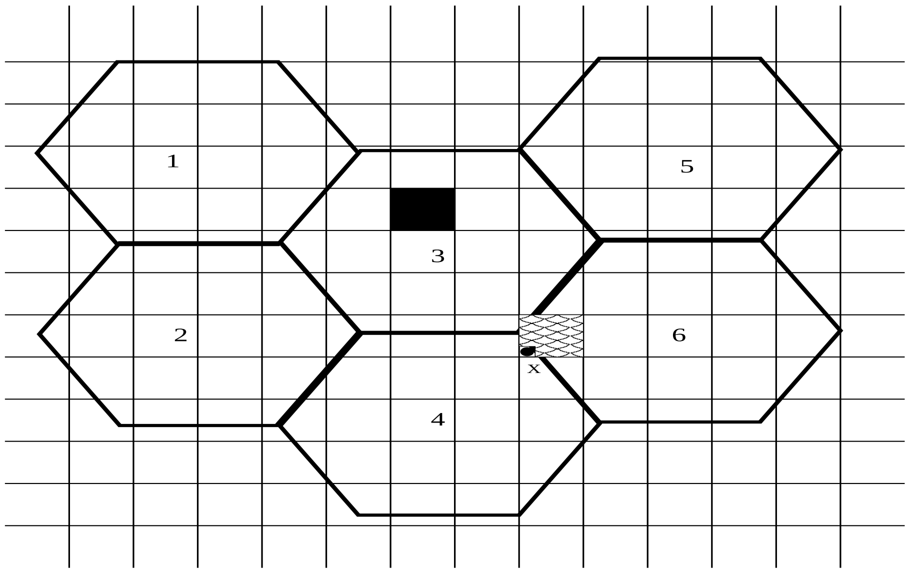

Figure 4.1: Overlap between product regions and Voronoi regions of the higher dimension VQ. Observe that the shaded region intersects multiple Voronoi regions, while the solid region is fully contained in a single Voronoi region.

Observe that the PVQ provides a partition of the N-dim Euclidean space, as does the higher dimension VQ. This is illustrated in Figure 4.1. Consider the following decoding strategy

- Encode the input using PVQ and identify the PVQ Voronoi region in which the input vector lies.

- Find the k Voronoi regions of the VQ which are intersected by this PVQ Voronoi region.

- Find the closest codeword to the input vector from among the codewords corresponding to the k intersecting VQ Voronoi regions.

By the above procedure we are always guaranteed to find the true closest codeword to the input vector. However unlike full search VQ which needed 2^{NR} distance comparisons we only require k distance comparisons in addition to performing the initial PVQ (note that k will usually be different for different input vectors). In general $k << 2^{NR}$, so the encoding complexity will be significantly reduced. For example, in Figure 4.1 the solid PVQ Voronoi region only intersects Voronoi region 3 and hence no further comparisons are required; for the shaded Voronoi region, which intersects Voronoi regions 3, 4 and 6, three distance comparisons are required. Conventional PVQ design minimizes the MSE along each sub-dimension. However since we are refining the result of the PVQ encoder with the above strategy, to minimize the number of codewords comparisons we should minimize the mismatch in Euclidean space partition between that provided by the high dimension VQ and that provided by the PVQ. This can be cast as a joint compression and classification problem. Consider Figure 3.11, this shows the optimal Bayes classifier and the bins obtained for the scalar quantizer for Case(iii) of the previous chapter (Section 3.3). We can see that product scalar quantizer is approximating the Euclidean space partition of the optimal Bayes classifier. For the new PVQ design we have a similar requirement, i.e., the PVQ design should approximate the Voronoi regions of the high dimension VQ. In this chapter we use the ideas developed in last chapter to design PVQ to approximate a high dimension VQ.

Consider Figure 4.2, the input vector u is encoded as $\hat{u}$ by the full search VQ. If we were to encode the vector u using a PVQ and then encode the PVQ result with a full search VQ, we get the output as $\grave{u}$. To minimize the number of comparisons we should minimize

$$\int 1(\hat{u} \neq \grave{u}) f(u) du$$

where $f(u)$ is the *pdf* of u. This is equivalent to the joint compression and classification problem with the classifier replaced by the high dimension VQ.

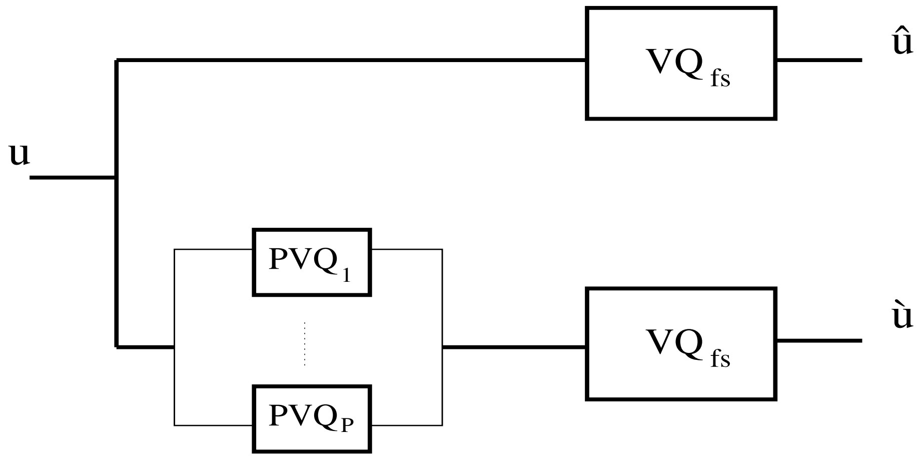

Figure 4.2: The top path represents the conventional full search VQ encoding scheme. The bottom path shows the conceptual encoding scheme where the PVQ quantized data is encoded by the full search VQ. Since in this case the full search VQ is operating on quantized data it can be replaced with a table lookup.

Hence in this chapter, we consider the design of a PVQ such that the objective is to minimize the error introduced in the labeling; i.e., we design the PVQ to minimize the probability that the quantized label is different when quantizing the PVQ quantized vector with the high dimension VQ instead of quantizing the original vector.

For the joint compression and classification problem a parametric algorithm which required knowledge of the joint *pdf* of the source was proposed in the last chapter. An empirical algorithm was also proposed. However the empirical algorithm required partitioning of the space into small bins. While this technique can be applied to scalars it is not straightforward to extend it to higher dimensions. The applications in the last chapter considered classifiers of dimension 2 and 4; however in this chapter we shall consider image compression by VQ, where the VQ operates on large block sizes (e.g., 4 by 4 blocks). For this vector size (16) reliable estimation of the joint *pdf* will require a very large training set which is usually not available. In addition the empirical algorithm of the last chapter will be computationally very expensive and may not provide reliable results when the training set is small. Hence we develop an empirical algorithm which can overcome these shortcomings and can be easily extended to higher dimensions. We will show that the design of the PVQs using this empirical algorithm can achieve improved performance over independent design of the VQs, where the individual VQs are designed to only minimize distortion.

Related work is discussed in Section 4.2. Our proposed algorithm is explained in Section 4.3. The new empirical design algorithm which can be used for large vector dimensions is proposed for situations where the *pdf* is not known and cannot be reliably estimated. An application using Hierarchical VQ (HVQ) as the PVQ is considered in Section 4.4. Finally conclusions are provided in Section 4.5.

4.2 Previous work

We first briefly review related work in the areas of fast encoding with codeword refinement and iterative PVQ design.

4.2.1 Pre-processing for fast encoding

Pre-processing the input data can enable fast encoding with an unstructured codebook. Fine-Coarse VQ [37] uses a structured VQ as the pre-processor. This is followed by a table lookup to find the final codeword from the unstructured higher dimension codebook. In the FCVQ design, the initial structured VQ is kept fixed, and the optimal lookup tables and the unconstrained codebook are designed based on the coarse VQ. However this differs from our approach where we fix the higher dimension codebook and design the initial structured VQs and lookup tables. Obviously FCVQ cannot be used in applications where we are given a VQ and the goal is to speed up encoding performance. However our proposed method will work in these applications. Also the PSNR obtained by our method will be higher than the PSNR obtained by FCVQ because in our case the higher dimension codebook is designed using only the training data, while in FCVQ the higher dimension codebook is optimized for the training data and the initial structured VQ.

K-d [7] trees were originally proposed for information retrieval by associative searches. They have been widely used for nearest neighbor encoding in vector quantization (see [50] and references in it). The K-d tree is used to identify the leaf node bucket containing the input vector. To find the closest codeword all codewords of the codebook whose Voronoi region intersects the bucket associated to the leaf node are examined. The Euclidean partitioning of the K dimensional space by the Voronoi regions of unstructured codebook is approximated by the partitioning of the Euclidean space by the buckets of the K-d tree. However good approximation is restricted by (i) a greedy non-iterative algorithm and (ii) the requirement of rectangular buckets. Our design is more general since it is an iterative algorithm (where the encoders along each dimension are iteratively redesigned until convergence) and in addition has the flexibility of dividing the vector into sub-vectors and using VQs for each of the sub-dimensions. Hence it is not restricted to only rectangular partition of the K dimension space. Therefore the final product regions can better approximate the Voronoi regions of the unstructured codebook, as compared to the approximation obtained by the rectangular buckets of the K-d tree, which will reduce the number of codeword comparisons required after the final product region has been identified.

4.2.2 Structured VQs

To reduce the encoding complexity of high dimension VQ_{fs}, structured VQs with faster encoding time have been proposed. These include tree structured VQ, multistage VQ, product VQ (PVQ) and Hierarchical VQ (HVQ) [11]. Unlike VQ_{fs} which for an input vector is able to find the closest codeword, the structured VQs may not always find the closest codeword. So while structured VQs reduce the encoding time, most of them have the undesirable effect of lower PSNR when compared to VQ_{fs}.

4.2.2.1 Hierarchical VQ

HVQ is a method of encoding vectors by using only table-lookups. The vector is partitioned into smaller vectors, and each of these sub-vectors is used to index a table. If l_i is the lookup table at level i, then the HVQ encoding can be represented as (for a 3-stage HVQ encoder)

$$HVQ(\mathbf{X}) = l_3(\{l_2(\{l_1(\mathbf{X_1}, \mathbf{X_2}), l_1(\mathbf{X_3}, \mathbf{X_4})\}), l_2(\{l_1(\mathbf{X_5}, \mathbf{X_6}), l_1(\mathbf{X_7}, \mathbf{X_8})\})\}) \quad (4.1)$$

where $\mathbf{X} = [\mathbf{X_1}, \mathbf{X_2}, ...\mathbf{X_8}]$ is a vector of dimension N. If $\beta^{i-1}(\cdot)$ and $\beta^{i}(\cdot)$ are the decoders at the $i-1^{th}$ and i^{th} level, then

$$l_i(m, n) = argmin_k(\beta^i(k), \{\beta^{i-1}(m), \beta^{i-1}(n)\}), \quad (4.2)$$

i.e., every product codeword combination is represented by the closest codeword from the next higher dimension. The codebook at every level is designed using the generalized Lloyd algorithm (GLA) with sub-vectors of suitable dimension extracted from the training set vectors. Note that we require a maximum of one table lookup per sample for encoding.

4.2.2.2 Weighted Transform Hierarchical VQ

An advantage of HVQ is that the complexity of the distortion measure does not influence the complexity of the encoder. The distortion measure is pre-computed and stored in the lookup table. Hence using (complex) perceptually meaningful distortion measures [10] can improve the performance with no increase in encoding complexity. Instead of using the conventional mean square error distortion measure, i.e,

$$d(\mathbf{x}, \hat{\mathbf{x}}) = \sum (x_j - \hat{x}_j)^2 \quad (4.3)$$

a transform $\mathbf{T}$ is used and a perceptually weighted distortion measure is used in the transformed domain, i.e.,

$$d(\mathbf{x}, \hat{\mathbf{x}}) = \sum w_j (y_j - \hat{y}_j)^2 \tag{4.4}$$

where y_j and $\hat{y}_j$ are the components of $\mathbf{y} = \mathbf{T}\mathbf{x}$ and $\hat{\mathbf{y}} = \mathbf{T}\hat{\mathbf{x}}$ respectively, and $w_1, \ldots, w_N$ is the set of perceptual weights. Note that if $\mathbf{T}$ is an orthogonal transform and if the mean square error distortion measure is used instead of the perceptually weighted distortion measure then there is no gain in using WTHVQ, because the codewords designed for the transformed data will just be rotations and/or reflections of the codewords designed for the original data. The design algorithm for WTHVQ is the same as that for HVQ, except that the VQ codebooks for every level are designed using the perceptually weighted distortion measure.

4.2.2.3 Entropy constrained product VQ

In the conventional PVQ design, each of the VQs is designed independently to minimize the weighted sum of distortion and rate along each sub-dimensions. Here separate entropy coders are used for each of the sub-dimension. However using a single entropy coder for the entire vector can result in a significant reduction in bitrate. In this case rate depends on the entire vector and an optimal entropy constrained PVQ (EC-PVQ) design should consider it as a global cost. For the EC-PVQ problem with a vector entropy coder, an iterative algorithm, which is similar to our proposed iterative algorithm, is proposed in [33] to optimally allocate bits to the different sub-dimensions of a Mean-Gain-Shape-VQ (MGSVQ). The EC-PVQ problem is similar to the joint compression and classification problem since both involve minimization of a local and a global cost. However for the joint compression and classification problem we assume the presence of a high dimension classifier (quantizer) which we are trying to approximate by the PVQ design. Also the cost functions to be minimized in the two problems are different (distortion and rate instead of distortion and misclassification).

4.3 Joint Product Encoder Design

We assume that we need to design P encoders and decoders; α_i and β_i, $i = 1, 2, \dots, P$ where

$$\begin{aligned} \alpha_i &: X_i \rightarrow \mathcal{I}_i, X_i \in R^{n_i}, \mathcal{I}_i \in \mathcal{Z} \\ \beta_i &: \mathcal{I}_i \rightarrow Y_i, \mathcal{I}_i \in \mathcal{Z}, Y_i \in R^{n_i} \end{aligned} \tag{4.5}$$

where the reproduction vector for i^{th} sub-dimension, Y_i, takes one of the values in $\{y_{i1}, y_{i2}, \dots, y_{ik^i}\}$ and $\sum_i n_i = N$. Note that the case when the same encoder and decoder is used for all the sub-dimensions is a special case with $Y_i = Y_j \forall\ i, j\ \in [1, 2, \dots, P]$ and $n_i = N/P\ \forall\ i$. We assume that the local cost functions Θ_i operate on sub-dimensions of the vector while the global cost function Φ operates on the entire vector. Our goal is to build encoders for each sub-dimension such that they minimize the cost function Θ_i and at the same time the decoded vector minimizes the global cost Φ. Since the encoders in each sub-dimension operate on only part of the entire vector it is not straightforward to compute the effect of encoding in a sub-dimension on the cost function Φ. An empirical algorithm which can be used with high dimension vectors is presented which designs the encoders by combining the global cost function Φ and the local cost functions Θ_i.

Without loss of generality we will show the encoder and decoder design for the i^{th} sub-dimension; the design is similar for other sub-dimensions. We can assume that we originally initialize all encoders to a certain state, so when designing the i^{th} encoder all other sub-vectors $\mathbf{X_j} : j \neq i$ have been assigned to a codeword along their respective sub-dimension. The training vectors $\mathbf{X} = [\mathbf{X_1}, \mathbf{X_2}, ...\mathbf{X_P}]$ are labeled, so we know $\delta(\mathbf{X})$, where $\delta(\cdot)$ is the N dimensional VQ. Given a training vector $[\mathbf{X_1} = \mathbf{x_1}, \mathbf{X_2} = \mathbf{x_2}, \ldots, \mathbf{X_P} = \mathbf{x_p}]$, we have to decide which codeword in the i^{th} codebook best represents $\mathbf{x_i}$, so that both distortion and misclassification are minimized. The codeword which best represents $\mathbf{x_i}$ in minimizing misclassification is given by

$$\arg\min_k [1(\delta(\beta_1(\alpha_1(\mathbf{x_1})), \ldots, \beta_i(k), \ldots, \beta_P(\alpha_P(\mathbf{x_P}))) \neq \delta(\mathbf{X}))] \tag{4.6}$$

Note that there can be multiple codewords minimizing equation (4.6), in which case to minimize misclassification it does not matter which codeword we pick. However for convergence we should also consider distortion and pick the closest codeword from among the competing codewords which minimize misclassification. So combining the above cost with distortion we get the encoder for sub-dimension i, i.e.,

$$\begin{aligned}\alpha_i(\mathbf{x_i}) \quad = \quad & \arg\min_k [(\mathbf{x_i} - \beta_i(k))^2 \; + \\ & \lambda \cdot 1(\delta(\beta_1(\alpha_1(\mathbf{x_1})), \ldots, \beta_i(k), \ldots, \beta_P(\alpha_P(\mathbf{x_P}))) \neq \delta(\mathbf{X}))\end{aligned} \tag{4.7}$$

and the decoder that minimizes the distortion is found as

$$\beta_i = \arg\min_z E[d(\mathbf{X_i}, \mathbf{z} | \alpha_i(\mathbf{X_i}))] \tag{4.8}$$

The intuition behind the above encoder design is that it penalizes the codewords which cause misclassification in $\mathbf{X}$, given the current configuration, and biases the decision towards codewords which result in correct classification, while not increasing the MSE significantly. By incorporating the classification cost during the encoder design, we are indirectly influencing the decoder outputs (i.e., β_i). This ensures that our system minimizes the joint cost of distortion and misclassification, better than a system designed using an independent encoder design targeting only distortion.

The total cost over all the dimensions is

$$J = \sum_{i=1}^{P} \left[(\mathbf{X_i} - \beta_i(\alpha_i(\mathbf{x_i})))^2 \right] + \lambda * 1(\delta(\{\beta_1(\alpha_1(\mathbf{x_1})), \ldots, \beta_P(\alpha_P(\mathbf{x_P}))\}) \neq \delta(\mathbf{X})) \quad (4.9)$$

The encoder as defined by equation (4.7) will not increase the total cost J, since it is the best assignment of the sub-vector $\mathbf{x_i}$, given the current configuration along the other dimensions. The cost J is bounded below by 0, so the design of the encoder and decoder by equations (4.7) and (4.8) can be expected to converge (the algorithm always converged during our simulations), although not necessarily to a global minimum. The problem with this encoder design is that the resulting encoders are not guaranteed to be regular for a given λ. Consider two different vectors both of which have $\mathbf{x_i}$ in the i^{th} sub-dimension. Depending on the class of the two vectors the above encoders assign each of these to different codewords in the i^{th} codebook. While this is not desirable, the extent of non-regularity can be controlled by the Lagrange multiplier λ. The actual encoding is done using a minimum distance criteria, so as to eliminate the problems of non-regularity. After the design of the encoders, one iteration of the GLA was used with nearest neighbor encoding to produce the final codebook. The iterative design is summarized as

```
Algorithm 7 (Iterative empirical quantizer design)
Step 1 : Initialize the encoders for the $P$ sub-dimensions, initialize $\lambda$ to a fixed constant
Step 2 : for $i = [1, \ldots, P]$
            Use GLA to design the encoder and decoder using equations (4.7) and (4.8) for
            $i^{th}$ sub-dimension.
Step 3 : If the codewords in any dimension changed go to Step 2.
Step 4 : Use GLA with only distortion criteria to generate the final codebook.
```

4.4 Applications

PVQs while having lower encoding time achieve a lower PSNR when compared to a full dimension VQ. As mentioned in Section 4.2.2 the last stage of the conventional HVQ chooses the VQ_{fs} codeword which is closest to the product region reproduction level. This procedure is not guaranteed to select the closest VQ_{fs} codeword for all input vectors. Consider the case of a 2-D VQ being approximated by a product of 2 scalar quantizers as shown in Figure 4.1. Observe that the shaded product region intersects Voronoi regions 3,4 and 6 of the higher dimension VQ. For the input $\mathbf{x}$ which lies in the shaded product region the closest VQ_{fs} codeword is 4, however the HVQ encoder will encode $\mathbf{x}$ with VQ_{fs} codeword 6, since this is the closest codeword to the shaded product region reproduction level. This sub-optimality can be eliminated by searching among the codewords whose Voronoi regions intersect the shaded product region (i.e., 3,4 and 6 in the example). This codeword refinement [22] after product encoding will ensure that the distortion introduced for the input vector will be the same as introduced by VQ_{fs}. But unlike the full search algorithm we do not need to search for the best candidate from the entire full dimension codebook, we only need to compare those codewords whose Voronoi region intersects the product region. We cannot use the technique proposed in [22] to identify the candidate codewords because their method relies on rectangular partitioning of the Euclidean space by the buckets of the K-d tree. Instead we can use the training data to determine which Voronoi regions are intersected by a product region. For an input vector, HVQ is used to find the product region p, and full search is used to find the closest VQ_{fs} codeword c. In the last stage HVQ lookup table, the index of c is added to the candidate list for the product region p. This procedure is repeated for all training vectors in the training set. During encoding once a vector is found to lie in a product region, we can search for the closest codeword from among the candidate codewords for that product region. This will result in exact match for vectors in the training set. However for vectors not in the training set, the "true" closest codeword may not be identified. To overcome this problem we find the distance of the product region reproduction to the closest VQ_{fs} codeword and store all codewords which are within a threshold $T(> 1)$ of this minimum distance.

Due to codeword refinement the encoding time is increased when compared to HVQ. The increase can be reduced by adopting the proposed joint encoder algorithm. The increase in complexity arises because unlike HVQ, we now have multiple choices at the last stage. By reducing the number of potential candidate codewords we can reduce the encoding complexity. For an HVQ with L stages, the goal is to reduce the mismatch between the partition of the Euclidean space by the first $L-1$ stages of the HVQ and the

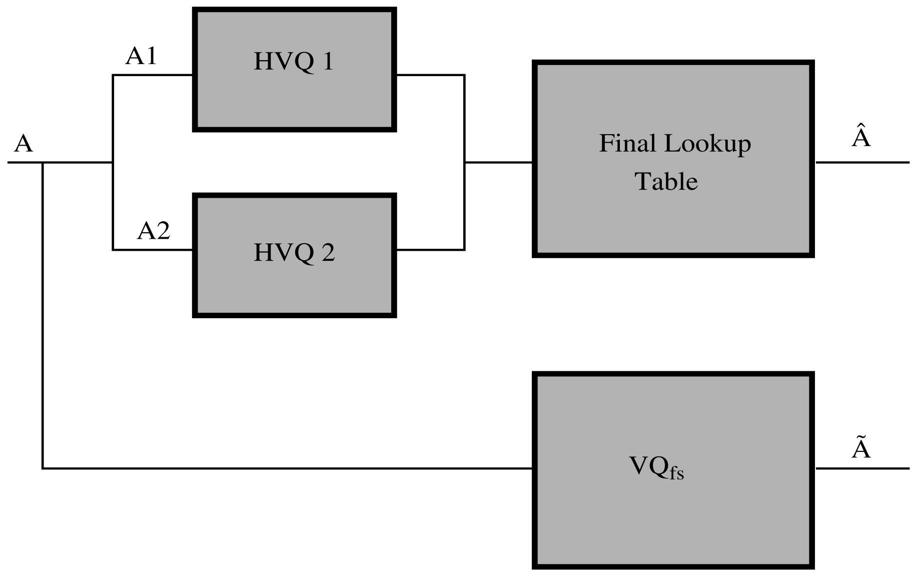

Figure 4.3: The HVQ encoding operation can be looked as a classification problem. The VQ_{fs} is the classifier and $L-1$ stage HVQ is the encoder. The goal is to ensure that for vectors in a training set the same index is given by the above two methods.

partition of the Euclidean space by the Voronoi regions of the VQ_{fs} codebook. Figure 4.3 shows two encoding schemes: (i) encode the 2 sub-vectors of the vectors with two $L-1$ stage HVQs and use the combination for the final table lookup; (ii) encode the vector directly with a VQ_{fs}.

We can treat this as a classification problem where our objective is to minimize the likelihood of a vector being assigned to the wrong VQ_{fs} codeword by L stage HVQ lookup encoding, i.e.,

$$min_X\,[VQ_{fs}(\mathbf{X}) \neq l_L(\{HVQ_{L-1}(\mathbf{X_1}), HVQ_{L-1}(\mathbf{X_2})\})] \tag{4.10}$$

where l_L is the L^{th} level lookup table of the L stage HVQ. We treat the VQ_{fs} Voronoi regions as classes and we want to encode the N sub-dimensions of the vector independently with $L-1$ stage HVQs and ensure that the misclassification introduced is minimized.

Since the final encoding decision is made by codeword refinement, only the misclassification is important, actual distances do not matter. To ensure that the misclassification is minimized our proposed joint encoder algorithm is used to design two codebooks (one for each sub-dimension). The partition of the Euclidean space provided by the $L-1$ stage HVQs using our codebook matches better, when compared to the partition provided by a conventional HVQ design, with the partition of the Euclidean space by the Voronoi regions of the VQ_{fs} codebook, so the average number of codewords for refinement is reduced.

The above algorithm was used to design a modified HVQ, a modified WTHVQ and a modified entropy constrained HVQ (ECHVQ) [9]. In the ECHVQ the last stage codebook is an entropy constrained VQ, designed to minimize the cost function $J = D + \lambda R$, where D is the distortion, R is the rate and λ is a Lagrange multiplier used to control the trade-off between rate and distortion. The performance of the modified HVQs was tested using images. Images of size 512 by 512 were used for both training and testing. The full dimension vector was a 4 by 4 pixel block, the HVQ tables had 4 levels. The 3^{rd} level tables were designed using our proposed joint encoder algorithm and the 4^{th} level table contained all the candidate codewords. The test images were *lenna* and *baboon* and they were not included in the training set (neither for the VQ_{fs} design nor for the HVQ design). Table 4.1 shows the PSNR and number of vector distance computations for VQ_{fs}, partial distance elimination algorithm (VQ_{pde}) [6], HVQ, codeword refined HVQ (CR-HVQ) and codeword refined HVQ with joint encoder design (CR-JE-HVQ). Tables 4.2 and 4.3 respectively show the results for WTHVQ and ECHVQ. From Table 4.1 we can observe that with codeword refinement the PSNR of CR-HVQ is much better than that

Codebook size	Algorithm	Threshold	PSNR		Number of distance computations per image ($*10^3$)	
			Lenna	Baboon	Lenna	Baboon
64	VQ_{fs}	-	28.84	22.38	1049	
	VQ_{pde}	-	28.84	22.38	209	285
	HVQ	-	28.43	22.05	-	
	CR-HVQ	1.5	28.81	22.32	31	39
		2.0	28.83	22.36	35	59
	CR-JE-HVQ	1.5	28.80	22.33	31	41
		2.0	28.83	22.36	35	60
256	VQ_{fs}	-	30.61	23.30	4194	
	VQ_{pde}	-	30.61	23.30	595	925
	HVQ	-	29.71	22.74	-	
	CR-HVQ	1.5	30.47	23.18	67	72
		2.0	30.53	23.25	75	122
	CR-JE-HVQ	1.5	30.48	23.17	62	72
		2.0	30.54	23.24	70	119
1024	VQ_{fs}	-	31.89	24.09	16777	
	VQ_{pde}	-	31.89	24.09	1915	3181
	HVQ	-	30.45	23.21	-	
	CR-HVQ	1.5	31.68	23.83	165	132
		2.0	31.76	23.95	181	246
	CR-JE-HVQ	1.5	31.60	23.78	141	126
		2.0	31.70	23.91	155	226

Table 4.1: PSNR and encoding time for VQ_{fs}, VQ_{pde} [6], HVQ, codewords refinement HVQ (CR-HVQ) and codewords refinement joint encoder HVQ (CR-JE-HVQ). The PSNR of CR-HVQ and CR-JE-HVQ is better than that of HVQ, however the encoding time is significantly lesser than VQ_{fs}

Codebook size	Algorithm	Threshold	PSNR		Number of Distance computations per Image ($*10^3$)	
			Lenna	Baboon	Lenna	Baboon
64	$WTVQ_{fs}$	-	28.76	22.26	1049	
	$WTVQ_{pde}$	-	28.76	22.26	141	143
	WTHVQ	-	28.30	22.06	-	
	CR-WTHVQ	1.5	28.73	22.22	26	27
		2.0	28.75	22.24	29	35
	CR-JE-WTHVQ	1.5	28.73	22.22	26	26
		2.0	28.75	22.24	28	35
256	$WTVQ_{fs}$	-	30.39	22.91	4194	
	$WTVQ_{pde}$	-	30.39	22.91	368	387
	WTHVQ	-	29.62	22.63	-	
	CR-WTHVQ	1.5	30.30	22.85	52	47
		2.0	30.33	22.88	56	63
	CR-JE-WTHVQ	1.5	30.32	22.84	51	46
		2.0	30.34	22.87	54	61
1024	$WTVQ_{fs}$	-	31.60	23.44	16777	
	$WTVQ_{pde}$	-	31.60	23.44	1210	1296
	WTHVQ	-	30.47	22.99	-	
	CR-WTHVQ	1.5	31.43	23.30	116	85
		2.0	31.48	23.35	122	120
	CR-JE-WTHVQ	1.5	31.34	23.24	101	80
		2.0	31.40	23.29	106	106

Table 4.2: PSNR and encoding time for VQ_{fs}, VQ_{pde} [6], WTHVQ, codewords refinement WTHVQ (CR-WTHVQ) and codewords refinement joint encoder WTHVQ (CR-JE-WTHVQ). The PSNR of CR-WTHVQ and CR-JE-WTHVQ is better than that of WTHVQ, however the encoding time is significantly lesser than $WTVQ_{fs}$

Codebook size	Algorithm	Threshold	PSNR		Number of Distance computation per Image ($*10^3$)	
			Lenna	Baboon	Lenna	Baboon
64	$ECVQ_{fs}$	-	27.77	22.04	1032	
	$ECVQ_{pde}$	-	27.77	22.04	284	333
	ECHVQ	-	27.36	21.73	-	
	CR-ECHVQ	1.5	27.73	21.98	17	22
		2.0	27.76	22.02	19	34
	CR-JE-ECHVQ	1.5	27.72	21.98	16	21
		2.0	27.75	22.02	18	32
256	$ECVQ_{fs}$	-	30.05	23.31	4194	
	$ECVQ_{pde}$	-	30.05	23.31	699	945
	ECHVQ	-	29.20	22.70	-	
	CR-ECHVQ	1.5	29.90	23.14	42	51
		2.0	29.96	23.23	47	83
	CR-JE-ECHVQ	1.5	29.91	23.13	39	47
		2.0	29.97	23.22	43	77
1024	$ECVQ_{fs}$	-	31.71	24.13	16777	
	$ECVQ_{pde}$	-	31.71	24.13	1960	3067
	ECHVQ	-	30.39	23.22	-	
	CR-ECHVQ	1.5	31.44	23.83	101	93
		2.0	31.54	23.97	111	164
	CR-JE-ECHVQ	1.5	31.40	23.79	86	81
		2.0	31.52	23.92	95	142

Table 4.3: PSNR and encoding time for $ECVQ_{fs}$, $ECVQ_{pde}$ [6], ECHVQ, codewords refinement ECHVQ (CR-ECHVQ) and codewords refinement joint encoder ECHVQ (CR-JE-ECHVQ). The PSNR of CR-ECHVQ and CR-JE-ECHVQ is better than that of ECHVQ, however the encoding time is significantly lesser than $ECVQ_{fs}$

Algorithm	Codebook size	Total lookup table size (k byte)	
		T=1.5	T=2.0
CR-HVQ	64	723	1139
	256	887	1690
	1024	1100	2745
CR-JE-HVQ	64	803	1176
	256	966	1721
	1024	1052	2190
HVQ	64	246	
	256	262	
	1024	279	

Table 4.4: Memory requirement for the lookup tables. For small VQ codebook size the memory required by CR-HVQ is lesser than CR-JE-HVQ, however for large VQ codebook size savings in memory can be achieved.

for HVQ, and with a threshold of 2 we almost achieve the performance of VQ_{fs}. Observe that for all cases the number of distance comparisons for CR-HVQ and CR-JE-HVQ is significantly lesser than that for VQ_{fs} and VQ_{pde}. Also note that the encoding time for the same threshold is almost always lesser for CR-JE-HVQ when compared to CR-HVQ (for images in the train set the number of comparisons was always lesser with CR-JE-HVQ when compared to CR-HVQ), although the PSNR is almost the same, which shows that with our proposed algorithm, the number of potential candidate codewords for refinement is reduced when compared to a conventional HVQ encoder with codeword refinement. Observe that as the VQ codebook size increases the reduction in number of comparisons for CR-JE-HVQ over CR-HVQ becomes more significant, because at lower codebook size the PVQ designed to minimize just distortion is able to closely approximate the VQ Voronoi regions however as the number of VQ Voronoi regions increase the mismatch also increases and thus our design gets improved performance. Similar observations can be made for the WTHVQ and ECHVQ. This shows the generality of the proposed technique, and it can be applied to some of the other HVQ techniques proposed in [9]. Finally Table 4.4 shows the total lookup memory size required for HVQ, CR-HVQ and CR-JE-HVQ. We observe that both CR-HVQ and CR-JE-HVQ require significantly greater memory requirement when compared to HVQ. Also since CR-JE-HVQ requires an extra level 3 lookup table, the memory requirement for VQ codebook sizes 64 and 256 is lower for CR-HVQ, however for VQ codebook size the total savings in memory outweighs the memory requirement for the extra lookup table and the total memory requirement for CR-JE-HVQ is lesser than that for CR-HVQ.

4.5 Conclusions

The joint compression and classification algorithm presented in the previous chapter was modified to enable its application to high dimension cases when the source *pdf* was not available. We proposed a codeword refinement strategy to achieve improved PSNR performance over HVQ. To minimize the number of codewords used for refinement we used the empirical joint compression and classification algorithm to propose a new design for table lookup encoders which could reduce the encoding time over plain codeword refinement schemes. An interesting extension of the method of refining with codewords of the intersecting Voronoi regions to a given PVQ Voronoi regions is the potential applications in trading off between performance and complexity. If every intersecting Voronoi region is labeled with the probability of intersection between the PVQ Voronoi region and it, then the higher the intersection the greater will be the probability and non-intersecting codewords will have probability zero. We can form a probability indexed descending list of VQ codewords. To reduce the complexity we have to remove codewords from the candidate list. To ensure minimum effect on performance the codewords with least probability should be removed (note that non-intersecting codewords will not be in the candidate list, and hence every codeword we remove will potentially lower the PSNR). This will enable us to quantify the performance vs. complexity trade-off.

Chapter 5

Minimum Mutual Information Loss Encoder

5.1 Introduction

In the previous chapter, we showed that when classification is one of the objective of the application, then a pure MSE optimized quantizer is not optimal. In this chapter we extend this analysis to investigate a sub-dimension based distortion metric which enables design of quantizers which are better optimized for classification applications. Assume that the encoded data is only used for classification (and not for reproduction). MSE optimized quantizers attempt to ensure that the decoded data is as "similar" as possible to the original data in the Euclidean distance sense. While minimizing MSE will ensure that classification performance is not severely degraded, it is not apparent that the MSE design is optimal for classification applications. Then what is the right metric to be used during quantizer design? Obviously, optimizing the quantizer with the metric used to evaluate classification performance is the right strategy. The metric used to evaluate classifier performance is *probability of classification error*, P_e (also called probability of misclassification). If C is the true class label of the data and $\tilde{C}$ is the class label estimated by the classifier, then

$$P_e = P(C \neq \tilde{C}) \tag{5.1}$$

Unfortunately P_e does not have a mathematically tractable expression which will allow us to use it for (sub-vector) quantizer design. Consider a simple one dimension, two class problem. The *pdfs* of the classes are shown in Figure 5.1. It is obvious that to minimize P_e the class boundary should be placed at 0. An MSE optimized quantizer, however, does not use this knowledge of the position of the class boundary. In the previous chapter we showed how incorporating the knowledge of the position of the class boundary by using indicator function can be used to design better quantizers for classification applications. In this chapter instead of using the "hard" information provided by indicator functions we use the "soft" information provided by class conditional *pdfs*. This is explained in more detail before.

To enable use of the "soft" information we make use of the *information-theoretic measure mutual information (MI)* to enable design of quantizers for classification applications. In particular, we measure the quantizer performance by the loss in MI introduced by quantization, i.e.,

$$MIL = I(X;C) - I(\hat{X};C) \tag{5.2}$$

where, X is the data to be classified, C is the true class label associated to X and $\hat{X}$ is quantized value of X. $I(X;C)$ is the mutual information between the data and the class labels and $I(\hat{X};C)$ is the mutual information between the quantized data and the

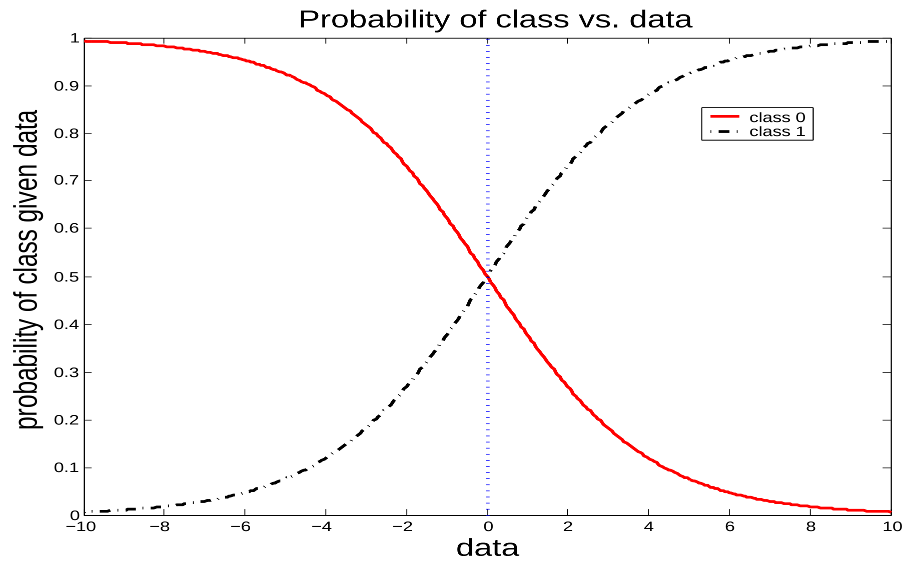

Figure 5.1: Probability of class given data for a two class problem. The means are -1 and 1 for the two classes and both classes have standard deviation 2.

class labels. Therefore, instead of preserving Euclidean distance between X and $\hat{X}$, we attempt to preserve the class information contained in data X.

The advantage of using MI loss instead of P_e, is that MI loss can be defined even for sub-vectors of the entire vector used for classification while P_e could not be defined. This is important because to alleviate the complexity during encoding it is necessary for the encoders to operate on low dimensions sub-vectors of the entire vector used for classification. Further justification for choosing to minimize MI loss can be obtained from Fano's inequality. It is known that mutual information between the classes and the data used for classification provides a (tight) lower bound to probability of error during classification. It can additionally be shown that the loss in mutual information due to quantization approximates the increase in probability of classification error due to quantization.

Specifically, it will be shown that when the quantizer design minimizes the MI loss, then the "distance" between the *a-priori* class conditional *pdfs* $p(c|x)$ and the *a-posteriori* class conditional *pdfs* $p(c|\hat{x})$ after quantization is minimized. The optimal classifier rule for the example in Figure 5.1 is

$$\tilde{C} = \begin{cases} 0 & \text{if } p(c_0|x) \geq p(c_1|x) \\ 1 & \text{otherwise} \end{cases} \tag{5.3}$$

By ensuring the quantizer design makes $p(c|\hat{x})$ as similar to $p(c|x)$ as possible it can be expected that the impact quantization has on classification will be minimized. The measure of dissimilarity between *pdfs* can be measured by the Kullback-Lieber (KL) divergence between them, which is given by

$$D_{KL}(p(c|x)||p(c|\hat{x}) = \int_x f(x) \sum_{i \in \mathcal{C}} p(c_i|x) log \left(\frac{p(c_i|x)}{p(c_i|\hat{x})} \right) dx \tag{5.4}$$

It will be shown that the KL divergence emerges as the right "effective distortion metric" when quantizers are designed to minimize the MI loss incurred by quantization. Note that this approach encompasses the previous approach (Chapter 3) as a special case if the "soft" information $p(c_i|x) \forall i \in \mathcal{C}$ is replaced by the "hard" information

$$L(c_i) = \begin{cases} 1 & \text{if } p(c_i|x) \geq p(c_j|x) \forall j \neq i \\ 0 & \text{otherwise} \end{cases} \tag{5.5}$$

with ties broken arbitrarily.

5.2 Relation between mutual information and P_e

Let the pair $\{X, C\}$ represent a data vector X associated with a class label $C \in \mathcal{C}$. Let $\delta(\cdot)$ be a deterministic statistical classifier which operates on the data vector X to estimate the class associated to it. Let $\hat{X} = Q(X)$ be the quantized value of X, where $Q(\cdot)$ is the quantizer. Let $\tilde{C} = \delta(X)$ and $\hat{C} = \delta(\hat{X})$ be the class labels estimated by the classifier from X and $\hat{X}$, respectively. Let $P_e^u = P(\tilde{C} \neq C)$ and $P_e^q = P(\hat{C} \neq C)$ be the probability of error when unquantized and quantized data are used by the classifier respectively.

5.2.1 Fano's inequality

Suppose we wish to estimate (classify) a random variable Θ and observe a random variable Z which is related to Θ by the conditional distribution $p(z|\theta)$. From Z we calculate a function $\hat{\Theta} = g(Z)$, which is an estimate of Θ. If the probability of error $P_e = P(\Theta \neq \hat{\Theta})$, then Fano's inequality states that (for proof see pg 39 of [16])

$$H(P_e) + P_e log(|\Theta| - 1) \geq H(\Theta|Z)$$

5.2.2 Relating P_e to loss in mutual information

During classification the value of the data X is known, the value of the label C needs to be determined, therefore by Fano's inequality

$$H(P_e^u) + P_e^u log(|\mathcal{C}| - 1) \geq H(C|X)$$

After rearranging we get

$$\begin{aligned} P_e^u &\geq \frac{H(C|X) - H(P_e^u)}{log|\mathcal{C}|} \\ &= \frac{H(C) - I(C;X) - H(P_e^u)}{log|\mathcal{C}|} \end{aligned} \tag{5.6}$$

Similarly P_e^q can be lower bounded as,

$$P_e^q \geq \frac{H(C) - I(C;\hat{X}) - H(P_e^q)}{log|\mathcal{C}|} \tag{5.7}$$

It is known that Fano's inequality is tight (see pg 40 of [16] for an example of a pmf which achieves the bound with equality). Therefore equations (5.6) and (5.7) can be written as

$$P_e^u = \frac{H(C) - I(C;X) - H(P_e^u)}{log|\mathcal{C}|} + \epsilon_1 \tag{5.8}$$

$$P_e^q = \frac{H(C) - I(C;\hat{X}) - H(P_e^q)}{log|\mathcal{C}|} + \epsilon_2 \tag{5.9}$$

where ϵ_1 and ϵ_2 are the difference between the probability of errors P_e^u and P_e^q and their respective lower bounds in equations (5.6) and (5.7).

Then the increase in probability of error due to quantization is

$$\begin{aligned} P_e^i &= P_e^q - P_e^u \\ &= \frac{I(C;X) - I(C;\hat{X})}{log|\mathcal{C}|} - \frac{H(P_e^q) - H(P_e^u)}{log|\mathcal{C}|} + \epsilon_1 - \epsilon_2 \end{aligned} \tag{5.10}$$

If we assume quantization does not severely degrade classification performance then,

$$H(P_e^q) \simeq H(P_e^u)$$

Additionally, we can expect the difference $\epsilon_1 - \epsilon_2$ will be small. Therefore,

$$P_e^i \simeq \frac{I(C;X) - I(C;\hat{X})}{log|\mathcal{C}|} \tag{5.11}$$

From the above equation we observe that the increase in probability of classification error due to quantization is closely approximated by the loss in mutual information. Hence, it is reasonable to expect that by minimizing loss in mutual information during quantization indirectly ensures that the increase in probability of classification error is also minimized.

5.3 Information Bottleneck Method

The Information Bottleneck (IB) Method [66] provides a variation principle for efficient representation of information for a given rate. The authors assume the presence of a relevance variable C (for classification application the relevance variable would be the class label), which is not independent from the signal to be compressed X, i.e., they share a positive mutual information $I(X;C)$.

Their proposed method attempts to quantize X while retaining as much information about C as possible. The MI between X and C is given by

$$I(X;C) = \int_x f(x) \sum_c p(c|x) log \left(\frac{p(c|x)}{p(c)} \right) dx \tag{5.12}$$

The authors trade-off between the MI and the rate required to represent the compressed data. Their assignment retains a fixed amount of meaningful information about C while minimizing the number of bits needed from X. The cost function minimized is

$$J_1 = I(X;\hat{X}) - \lambda I(\hat{X};C) \tag{5.13}$$

or, equivalently the cost function to be minimized can be re-written as

$$J_2 = I(X;\hat{X}) + \lambda[I(X;C)] - I(\hat{X};C)] \tag{5.14}$$

Notice that since $I(X;C)$ is a constant, the two minimizations are equivalent. However Equation (5.14) provides a better intuitive understanding of the problem for classification applications. Since now the problem is stated as minimizing the MI loss due to quantization for a given rate. In Section 5.2 it was shown that the MI loss directly correlates to the increase in probability of classification error due to quantization.

The main result of the IB method is that minimization of Equation (5.13) when using a soft partition of the input data space, results in the Kullback-Liebler divergence between the *a-priori* conditional class *pdfs* $P(c|x)$ and the *a-posteriori* conditional class *pdfs* after quantization $P(c|\hat{x})$, $D_{KL}[p(c|x)||p(c|\hat{x})]$, emerges as the relevant "effective distortion metric". Additionally, an iterative algorithm similar to the Blahut-Arimoto (BA) algorithm is provided to design quantizers.

The short-comings of the IB method are

- It requires the knowledge of the joint *pdf* p(x,c). In most practical applications this joint *pdf* is not available.

- The BA-algorithm induces a soft partition of the input space. In actual encoding we require a hard partition of the input space. It is not clear if the IB method is optimal when a hard partitioning of the input space is used.

- The rate allocation problem is not addressed. It is well known that rate allocation plays an important role in achieving good performance for rate-distortion (RD) based systems. It can be expected that rate allocation will play an important role even in rate-mutual information loss (RMI) based systems.

- Dimension mismatch between the vector used for classification and that used for encoding has not been addressed. As mentioned before for practical encoding schemes the data vector X has to be encoded using a product encoder, due to computational requirements.

In the following section we address these short-comings. We provide an empirical quantizer design in Section 5.4 which uses a hard partitioning of the input space. We address the rate-allocation problem in Section 5.4.2.

5.4 Minimum Mutual Information Loss Encoder

Before describing our minimum mutual information loss encoder, we introduce our notation. We represent vectors in bold and the components of the vector are enclosed in $\{\}$. Random variables (RVs) are represented by uppercase letters and the value taken by the RVs by lowercase letters, i.e., $\mathbf{X} = \mathbf{x} = \{x_1, \ldots, x_N\}$ implies the vector RV $\mathbf{X}$ takes the value $\mathbf{x}$, which has N components x_i, $i = 1, \ldots, N$. $\delta(\cdot)$ denotes a statistical classifier, $\alpha(\cdot)$ and $\beta(\cdot)$ denote the encoder and decoder respectively. We use the quantizer, $Q(\cdot)$, as a shorthand for the encoder-decoder pair, i.e., $Q(\cdot) \triangleq \beta(\alpha(\cdot))$.

Let $[\mathbf{Y}, C]$ denote a continuous (NT)-dimensional RV $\mathbf{Y} = \{\mathbf{X_1}, \ldots, \mathbf{X_T}\}$ which is associated with a class label C that takes a value in $1, \ldots, L$, and $\mathbf{X_i}$ is an N-dimensional RV. In speech recognition the RV $\mathbf{Y}$ represents feature frames of a phoneme, $\mathbf{X_i}$ represents a feature frame, and C represents the phoneme $\mathbf{Y}$ belongs to. Assume to represent $\mathbf{Y}$[1], we are given a rate R, the rate constraint could be due to transmission or storage requirements. The problem we consider is that of finding the best representation $\hat{\mathbf{Y}} = \{\hat{\mathbf{X}}_\mathbf{1}, \ldots, \hat{\mathbf{X}}_\mathbf{T}\} = \{Q(\mathbf{X}_1), \ldots, Q(\mathbf{X}_T)\}$ s.t. $H(\hat{\mathbf{Y}}) \leq R$, which minimizes the probability of error in classification, i.e., $P_e(\delta(\hat{\mathbf{Y}}) \neq C)$, where $H(\hat{\mathbf{Y}})$ is the entropy of the RV $\hat{\mathbf{Y}}$. The fundamental problem we are addressing is given the constraint that the classifier has

[1]The rate required to represent a continuous RV $\mathbf{X}$ with exact precision is ∞

to operate on compressed data with a rate limit, what is the best product quantizer that minimizes the probability that the class label is different when obtained from unquantized and quantized data? Note that we design the same quantizer $Q(\cdot)$ for all the feature frames, $\mathbf{X_i}$, $i = 1, \ldots, T$.

Unlike $P_e(\cdot)$, which is not defined for sub-vectors (the classifier can only make its decision using the entire vector), MI can be calculated even between sub-vectors $\mathbf{X_i}$ and the class labels. Since we design the same quantizer for all feature frames in what follows we drop the subscript in $\mathbf{X_i}$.

Different speech utterances could contain the same feature frames (possibly in different temporal locations) and still belong to different classes. So it is obvious that given a feature frame the class labels need not be the same, i.e., every feature frame $\mathbf{X} = \mathbf{x}$ is associated with a conditional *pdf* $p(c|x)$. Hence MI between the RV $\mathbf{X}$ and the RV C (class label) is given by

$$I(\mathbf{X}; C) = \int_{\mathbf{x}} f(\mathbf{x}) \sum_{c} p(c|\mathbf{x}) log \left(\frac{p(c|\mathbf{x})}{p(c)} \right) d\mathbf{x} \tag{5.15}$$

It is well known that

$$H(C|\mathbf{X}) = H(C) - I(\mathbf{X}; C) \tag{5.16}$$

where $H(C)$ is the original entropy of the class labels and $H(C|\mathbf{X})$ is the entropy of the class labels after observing $\mathbf{X}$. Therefore the MI between C and $\mathbf{X}$ is the amount by which uncertainty in class labels is reduced by observing $\mathbf{X}$. Obviously the larger $I(\mathbf{X}; C)$ is, the more relevant (useful) $\mathbf{X}$ is for the classification task. This intuition has been used previously in speech recognition to propose a maximum mutual information speech recognizer design technique [39] as an alternative to maximum likelihood techniques.

Our work also makes use of this intuition to design a *minimum mutual information loss (MMIL) quantizer*, $Q_{MI}(\cdot)$, which minimizes the loss in MI, $I(\mathbf{X};C) - I(\hat{\mathbf{X}};C)$, due to compression, where $\hat{\mathbf{X}} = Q_{MI}(\mathbf{X})$. The loss in MI due to compression corresponds to an increase in class uncertainty (see Equation (5.21) below), hence the MMIL quantizer is well suited for compressing data in classification applications. The MMIL quantizer treats loss in MI as the distortion incurred during quantization, similar to a minimum mean square error (MMSE) quantizer treating Euclidean distance as the distortion incurred.

The MI between the quantized data $\hat{\mathbf{X}}$ and C is

$$I(\hat{\mathbf{X}};C) = \sum_{c}\sum_{\hat{\mathbf{x}}} p(c,\hat{\mathbf{x}}) log\left(\frac{p(c,\hat{\mathbf{x}})}{p(c)p(\hat{\mathbf{x}})}\right) \leq I(\mathbf{X};C) \tag{5.17}$$

where $I(\hat{\mathbf{X}};C) \leq I(\mathbf{X};C)$ by the data processing inequality.

The *optimal MMIL quantizer*, $Q^*_{MI}(\cdot)$, subject to a rate constraint, is obtained by a constrained minimization

$$Q^*_{MI}(\cdot) = argmin_{Q_{MI}:(I(\mathbf{X};C)-I(Q_{MI}(\mathbf{X});C))\leq D_{MI}} I(\mathbf{X};Q_{MI}(\mathbf{X}))$$

Based on standard Lagrange techniques this constrained minimization can be converted into an unconstrained minimization, i.e.,

$$\begin{aligned} Q^*_{MI}(\cdot) &= argmin_{Q_{MI}} I(\mathbf{X};Q_{MI}(\mathbf{X})) \\ &\quad +\lambda(I(\mathbf{X};C) - I(Q_{MI}(\mathbf{X});C)) \end{aligned} \tag{5.18}$$

where λ is the Lagrangian multiplier which controls the trade-off between rate and distortion (i.e., loss in MI).

5.4.1 MMIL Encoder: Quantizer Design

One of our main motivations is to provide an *empirical* algorithm which can be used in practical applications to design quantizers directly from sample training data. To enable this we show how the distortion in Equation (5.18) can be estimated directly from data samples used to design the quantizer. First from the Markov chain $C \leftrightarrow \mathbf{X} \leftrightarrow \hat{\mathbf{X}}$, we have

$$p(\hat{\mathbf{x}}|c) = \int_{\mathbf{x}} p(\hat{\mathbf{x}}|\mathbf{x})p(\mathbf{x}|c)d\mathbf{x} \tag{5.19}$$

and

$$p(c|\hat{\mathbf{x}}) = \int_{\mathbf{x}} p(c|\mathbf{x})p(\mathbf{x}|\hat{\mathbf{x}})d\mathbf{x} \tag{5.20}$$

From Equation (5.16), the MI loss (distortion) is

$$I(\mathbf{X}; C) - I(\hat{\mathbf{X}}; C) = H(C|\hat{\mathbf{X}}) - H(C|\mathbf{X}) \tag{5.21}$$

where

$$H(C|\mathbf{X}) = -\int_{\mathbf{x}} f(\mathbf{x}) \sum_{c} p(c|\mathbf{x}) log\left(p(c|\mathbf{x})\right) d\mathbf{x} \tag{5.22}$$

and

$$H(C|\hat{\mathbf{X}}) = -\sum_{c} p(c) \sum_{\hat{\mathbf{x}}} p(\hat{\mathbf{x}}|c) log\left(p(c|\hat{\mathbf{x}})\right) \tag{5.23}$$

Substituting Equation (5.19) in Equation (5.23) we get

$$H(C|\hat{\mathbf{X}}) = -\sum_{c} p(c) \int_{\mathbf{x}} p(\mathbf{x}|c) \sum_{\hat{\mathbf{x}}} p(\hat{\mathbf{x}}|\mathbf{x}) log\left(p(c|\hat{\mathbf{x}})\right) d\mathbf{x} \tag{5.24}$$

Note that $p(\hat{\mathbf{x}}|\mathbf{x}) = 1$ if $Q(\mathbf{x}) = \hat{\mathbf{x}}$ and 0 otherwise. Therefore define

$$q(c|\hat{\mathbf{x}}) = \begin{cases} p(c|\hat{\mathbf{x}}) & \text{if } Q(\mathbf{x}) = \hat{\mathbf{x}} \\ 0 & \text{otherwise} \end{cases} \tag{5.25}$$

Substitute Equations (5.22), (5.24) and (5.25) in Equation (5.21) (by continuity arguments we assume $0log(0) = 0$). After rearranging, we get

$$I(\mathbf{X}; C) - I(\hat{\mathbf{X}}; C) = \int_{\mathbf{x}} f(\mathbf{x}) \sum_{c} p(c|\mathbf{x}) log \left(\frac{p(c|\mathbf{x})}{q(c|\hat{\mathbf{x}})} \right) d\mathbf{x} \tag{5.26}$$

If $d(\mathbf{x}, \hat{\mathbf{x}})$ represents the distortion between $\mathbf{x}$ and $\hat{\mathbf{x}}$, then $E[d(\mathbf{x}, \hat{\mathbf{x}})] = \int_{\mathbf{x}} f(\mathbf{x}) d(\mathbf{x}, \hat{\mathbf{x}}) d\mathbf{x}$. This implies that the distortion metric for the MMIL quantizer is $\sum_c p(c|\mathbf{x}) log \left(\frac{p(c|\mathbf{x})}{q(c|\hat{\mathbf{x}})} \right)$, i.e., the distortion is the KL distance between the *a-priori* conditional class *pdfs* before quantization, $p(c|\mathbf{x})$, and the *a-posteriori* conditional class *pdfs* after quantization, $q(c|\hat{\mathbf{x}})$. Therefore the MMIL quantizer attempts to choose those codewords $\hat{\mathbf{X}}$ which best preserve the *a-priori* conditional class *pdfs*. It is obvious that for statistical classifiers (e.g. HMMs) which choose the class label based on a maximum likelihood decision using $p(c|x)$, the MMIL quantizer will have less detrimental effect on the probability of misclassification when compared to traditional mean square error (MSE) based quantizers which do not explicitly consider the conditional class *pdfs* during quantization design. If the source *pdfs* are not known fine quantization is used to find an empirical estimate $\tilde{p}(c|x)$ of $p(c|x)$ from labeled training data [7]. Equation (5.20) is then used to find $p(c|\hat{x})$, where $\tilde{p}(c|x)$ is used instead of $p(c|x)$.

Algorithm 8 *Minimum mutual information loss vector quantizer design*

Step 0: *Calculate* $\tilde{p}(c|\mathbf{x})$ *for* $c = 1, \ldots, L$

Step 1: *Initialize all codewords* $\hat{\mathbf{x}}_i$*,* $i = 1, \ldots, I$*,* $d(0) = \infty, k = 1$*. Let* ϵ *be a small positive constant.*

Step 2: *Find* $p(c|\hat{\mathbf{x}}_i)$*,* $c = 1, \ldots, L$*,* $i = 1, \ldots, I$*; using Equation (5.20)*

Step 3: $\alpha(\mathbf{x}) = argmin_i \sum_c p(c|\mathbf{x}) log \left(\frac{p(c|\mathbf{x})}{q(c|\hat{\mathbf{x}}_i)} \right)$

Step 4: $\hat{\mathbf{x}}_i = \beta(i) = E[\mathbf{x}|\alpha(\mathbf{x}) = i]$

Step 5: $d(k) = \int_{\mathbf{x}} f(\mathbf{x}) \sum_c p(c|\mathbf{x}) log \left(\frac{p(c|\mathbf{x})}{q(c|\beta(\alpha(\mathbf{x})))} \right) d\mathbf{x}$

Step 6: *if* $(d(k-1) - d(k))/d(k) < \epsilon$ *STOP, else* $k = k+1$, *go to* **Step 2**

Here in **Step 4** we make use of the result that for encoders designed using KL distance as the distortion measure, the optimal decoder is the Lloyd decoder [58].

The flexibility of our encoder design is that since we are using loss in MI as the distortion measure rather than $P_e(\cdot)$, it can easily be applied to design independent quantizers for each of the components of the vector $\mathbf{X}$. If the vector $\mathbf{X}$ has N components, i.e., $\mathbf{X} = \mathbf{x} = \{x_1, \ldots, x_N\}$ then the optimal quantizer $Q_{MI}^{j*}(\cdot)$ for the j^{th} component is designed by minimizing $I(X_j; C) - I(Q_{MI}^{j*}(X_j); C)$.

Given that the designed quantizers need to satisfy a rate constraint, the standard entropy constrained quantization design technique [14] can be adopted, with MI loss as the distortion. The entropy constrained encoder $\alpha_n(\cdot)$ for the n^{th} component is

$$\alpha_n(x_n) = argmin_i \sum_c p(c|x_n) log \left(\frac{p(c|x_n)}{q(c|\hat{x}_{n_i})} \right) + \lambda l_{n_i} \tag{5.27}$$

where l_{n_i} is the number of bits used to represent the i^{th} codeword $\hat{x}_{n_i}$ in the n^{th} sub-dimension. As a simplification we use $l_{n_i} = -log_2(p(\hat{x}_{n_i}))$. The decoder is the Lloyd decoder $\beta_n(i) = E[x_n|\alpha_n(x_n) = i]$.

5.4.2 MMIL Encoder: Rate-Allocation

Rate-allocation (or bit-allocation) plays an important role during designing independent encoders for components of a vector. The GBFOS algorithm [53] has been used for rate-allocation in MMSE encoders. It relies on the calculation of rate vs. distortion points for each of the components and then selects the combination of points which satisfy the rate constraint and yield the minimum distortion. For MMIL quantizers the distortion is MI loss, hence several *rate vs. mutual information loss* points are calculated for each of the components. Then if the available rate is R, these calculated points are used by the GBFOS algorithm to allocate rates, $R_n, n = 1, \dots, N; \sum_n R_n \leq R$, to each of the components. The GBFOS algorithm is presented below.

Algorithm 9 *(GBFOS bit allocation)*

Step 1 : *For $n = 1, \dots, N$, set $B_n = q$. This is the initial bit allocation.*

Step 2 : *Calculate, for $n = 1, \dots, N$, for $i = 1, \dots, B_n$,*

$$S_n(B_n, B_n - i) \quad = \quad \frac{\Delta MIL_{overall}}{\Delta R_{overall}} = -\frac{MIL_n(B_n) - MIL_n(B_n - i)}{i} \tag{5.28}$$

Step 3 : *For each $n = 1, \dots, N$, determine i for which $S_n(B_n, B_n - i)$is minimized.*

Step 4 : *Determine the component for $S_n(B_n, B_n - i)$ is the lowest. Assume it is component l. (If minimum $S_n(B_n, B_n - i)$ is not unique, then select all components with this value). Set $B_l = B_l - i$.*

Step 5 : *Calculate $B_{alloc} = \sum_n B_n$. Check if $B_{alloc} \leq B$; if so for $n = 1, \dots, N, set R_n = B_n/T$, where T is the sampling frequency and Stop*

Step 6 : *Repeat Steps (2), (3), (4) and (5).*

The final rate-allocation for the component n is given by R_n.

To satisfy the rate allocation, during quantizer design, λ in Equation (5.27) is modified until $H(\hat{X}_n) \leq R_n$, standard bisection techniques can be used to find the "best" λ.

5.5 Experiments and Results

To evaluate our techniques we generated a mixture of eight 2D Gaussian sources. The means and the optimal Bayes classification boundaries for this mixture source are shown in Figure (5.2). 10,000 samples from each class were generated, each dimension was independently quantized and then used for classification. A classification error occurs if a sample from class i is classified as belonging to class $j \neq i$. The experiments were repeated 100 times and results reported are average results. The baseline classification error using unquantized data was 27.4%.

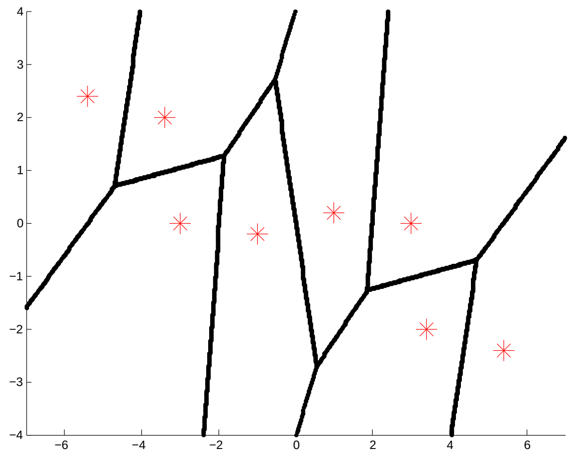

Figure 5.2: The mixture of eight 2D Gaussian sources used to evaluate our proposed techniques. Misclassification was 27.4% even when unquantized features were used, indicating significant overlap between the different sources.

The different quantization techniques (Q1-Q4) evaluated are listed in Table [5.1]. To illustrate the effect of MI on the different aspects of the quantizer, we progressively increase the significance the role MI plays in the quantizer operation. Q4 represents the quantization technique incorporating MI in all three phases, design, rate-allocation and encoding. This represents our best system. Figure (5.3) shows the results obtained when the different quantizers were used for encoding the data before classification. We plot the

ID	Design	Rate-allocation	Encoding
Q1	MSE	MSE	MSE
Q2	MSE	MI	MSE
Q3	MI	MI	MSE
Q4	MI	MI	MI

Table 5.1: In the different quantization techniques, we varied the design algorithm, the rate-allocation technique and the encoding scheme. Here MSE refers to mean square error distortion and MI refers to our proposed mutual information loss distortion. Note that we progressively increase the amount of role MI plays in the quantizer operation, by first using it only for rate-allocation, then for both rate-allocation and quantizer design and finally for all three operations.

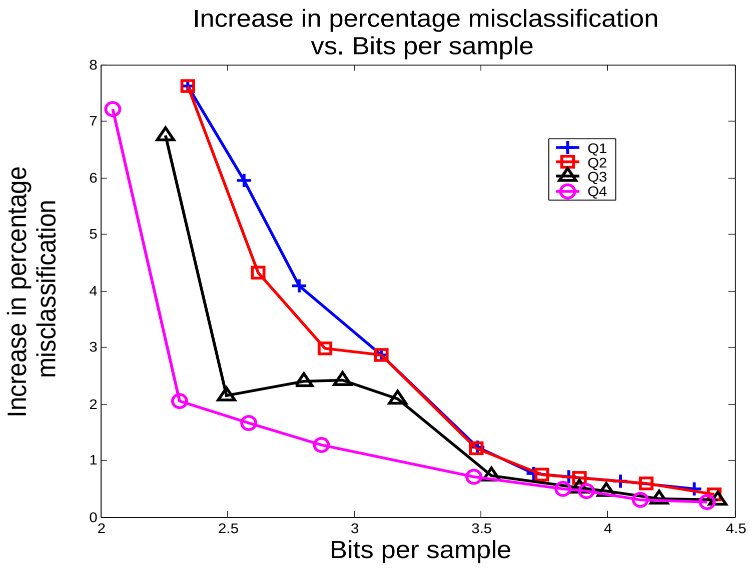

Figure 5.3: The results obtained by the different quantization schemes. Observe that as MI is increasingly used in the quantizer, the rate-classification performance always becomes better.

increase in misclassification vs. bits-per-sample. The increase in misclassification is with respect the baseline performance using unquantized data. Observe that at high bitrate (> 3.5 bits-per-sample), the performance of all systems is almost the same, although Q4 achieves the best results of the four techniques. However at low bit rates the sub-optimality of MSE for classification becomes clear. When the GBFOS algorithm uses rate vs. mutual information loss points instead of rate vs. MSE points to allocate rates to the different components of the vector we observe that the performance improves or stays the same (Q2 is better than Q1). Once MI is also included for quantizer design (Q3), we observe substantial improvement in performance. At 2.3 bits-per-sample the increase in misclassification due to Q1 is 7.6%, while due to Q3 it is only 2.2%. Notice that this improvement comes with no extra cost during encoding, i.e., Q3 still uses the sub-optimal MSE encoding, thus having no extra run time computational increase. However we observe that between 2.5 and 3.5 bits-per-sample, the performance of Q3 is not monotonic. This is due to the fact that a quantizer designed to minimize loss in MI, is used by an encoding scheme which minimizes MSE. To eliminate this mismatch MI can be used as the criteria during actual encoding (Q4). Observe that Q4 outperforms Q1, Q2 and Q3 at all bitrates, and additionally eliminates the non-monotonicity of Q3 (however the run time encoding complexity increases).

The significant performance improvements achieved for this eight class task motivated us to apply the MMIL technique to the problem of encoding speech in a distributed speech recognition (DSR) task (see Chapter 2). Our scalable DSR encoder [64] uses uniform scalar quantizers, hence we concentrate on rate-allocation. The rate-allocation to each of the mel frequency cepstral coefficients (MFCCs) was based on the importance of each

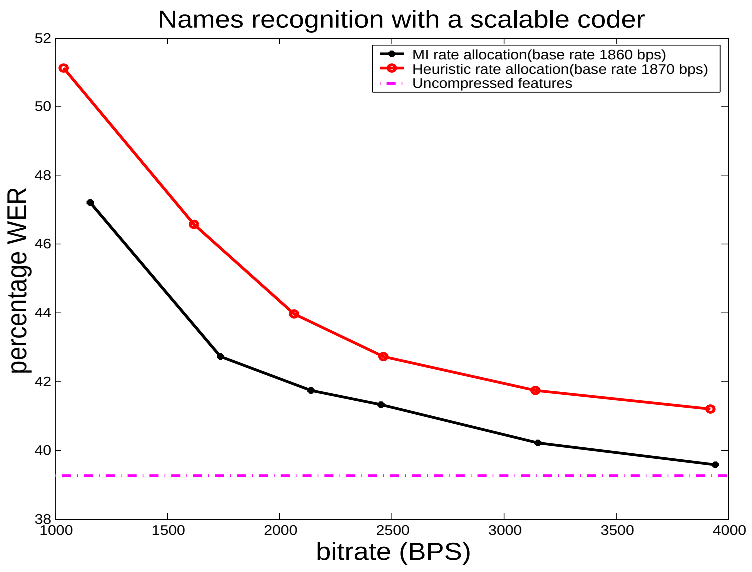

Figure 5.4: WER in spoken names recognition task. Significant performance improvements is achieved at all bitrates by the MI rate-allocation compared to the heuristic rate-allocation. At 3920 bps, the MI rate-allocation scheme results in only a 0.31% increase in WER when compared to using unquantized MFCCs.

MFCCs for recognition. It is unclear how to map importance to rate-allocation. However loss in MI is a better metric for allocating the rates, since the loss in MI indicates the increase in class uncertainty. The results obtained when rate-allocation to the MFCCs was done with MI is shown in Figure (5.4). The recognition task was a two stage spoken names recognition (see 2.4.3 for details). Observe that for all bitrates the MI rate-allocated encoder significantly outperforms the heuristically rate-allocated encoder. At 3920 bps the MI rate-allocation reduces the increase in WER due to compression from 1.92% for the heuristic rate-allocation to 0.31%, more than a six fold decrease. In terms of bitrate, the MI rate-allocation achieves at 2500 bps the same recognition performance as the heuristic rate-allocation does at 3920 bps, a 36% reduction in bitrate with no penalty in recognition performance.

5.6 Conclusions

We proposed mutual information, an alternative to MSE, as a more appropriate distortion criteria for encoder design in classification (recognition) applications. We showed that KL distance is the right optimization measure to ensure that the quantizer design minimizes loss in MI between the data samples and class labels. We provided a practical empirical quantizer design technique. We also showed that rate-allocation based on MI can result in significant performance improvements in distributed speech recognition.

Chapter 6

Future Work

6.1 Successively Refinable Predictive Encoding

In Chapter 2 a scalable predictive encoding scheme was presented for encoding MFCCs. The advantages of using scalable encoding for speech recognition were also discussed. Scalable encoding is also very useful even in other applications. For example consider digital video broadcast from a server to several clients. Assume that some of the clients have access to high bandwidth channels while others are bandwidth constrained. The most straightforward method of video transmission would involve the server transmitting two independent video streams, one encoded at high rate and the other at low rate. Obviously this is a wasteful strategy. The better alternative would be for the server to transmit a single scalable stream and the clients would receive the part of the video stream based on their bandwidth capacity, i.e., clients with high bandwidth would receive the entire stream while clients with low bandwidth would only receive the initial portion of the scalable stream.

Obviously, while this enables savings in bitrate, it can affect the performance. More formally assume that when the video is encoded at rates R_1 and R_2 ($R_2 > R_1$) and the distortion is D_1 and D_2 respectively. Then is it possible to encode the video with rate R_1 and distortion D_1 and then add a refinement at rate $R_2 - R_1$ and achieve distortion D_2? Or more generally is it possible to reach every point on the rate-distortion curve in a scalable manner? The answer to this question was presented by Equitz and Cover [20] and Rimoldi [52], who proved that for a source X to be successively refinable from $\hat{X}_1$ (rate R_1 and distortion D_1) to $\hat{X}_2$ (rate R_2 and distortion D_2), X, $\hat{X}_1$ and $\hat{X}_2$ have to form a Markov chain, i.e.,

$$X \leftrightarrow \hat{X}_2 \leftrightarrow \hat{X}_1 \tag{6.1}$$

6.1.1 Methods of scalable encoding

Perfect successive refinability is not achievable by practical scalable encoding techniques even for sources for which it is known that Equation (6.1) is satisfied. The encoding technique proposed in [20] relies on random coding techniques, and is only optimal asymptotically in the limit as the block length of the vector being encoded goes to infinity. In practical encoding techniques, it is computationally inferable to use large dimension vectors during encoding. Instead the encoders operate on low dimension vectors.

Our primary interest is in the design of scalable predictive encoders. Both [20] and [52] consider independent and identically distributed (*iid*) sources. It can be expected that design of practical scalable predictive encoders will be more challenging than scalable encoders for *iid* sources.

Most scalable predictive encoding techniques can be broadly classified in two different techniques, *Refinable scalable predictive encoders* and *Morphable scalable predictive encoders.*

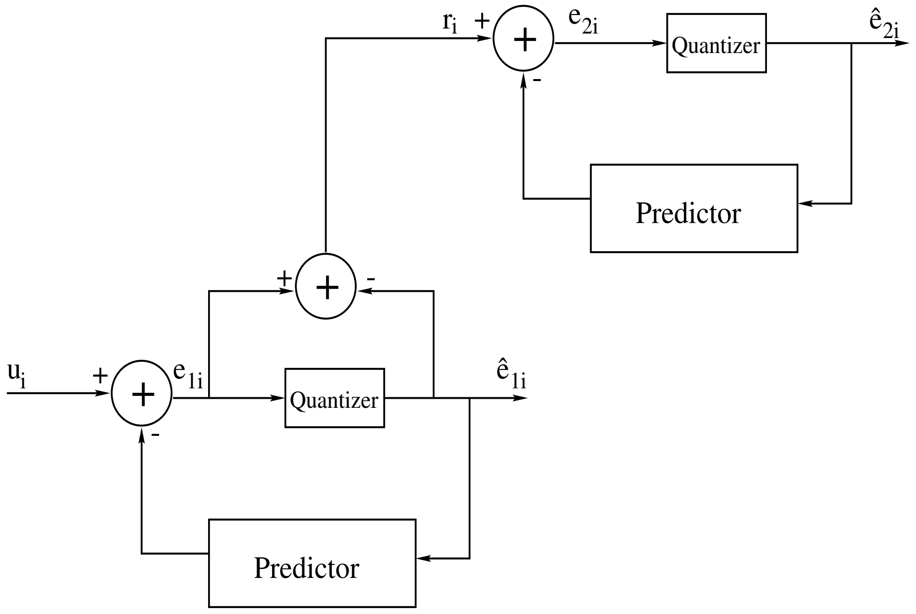

Figure 6.1: A refinable scalable predictive encoding scheme. Subsequent DPCM loops encode the quantization error of the previous DPCM loop.

A refinable scalable predictive encoder is shown in Figure 6.1. For ease of explanation let us assume that scalable stream consists of two layers. The lower rate description requiring rate R_1 is initially generated by a DPCM loop with a coarse quantizer. The quantization error of the first DPCM loop r_i is predicted with another DPCM loop to generate the refinement layer. The basic operation consists of initially generating a description requiring rate R_1, Then $R_2 - R_1$ refinement bits are sent to generate the higher fidelity description. The advantage of this approach is that both rate requirements R_1 and R_2 are fixed. However while distortion D_1 is fixed, we do not have control over D_2 (in general the distortion achieved will be greater than D_2).

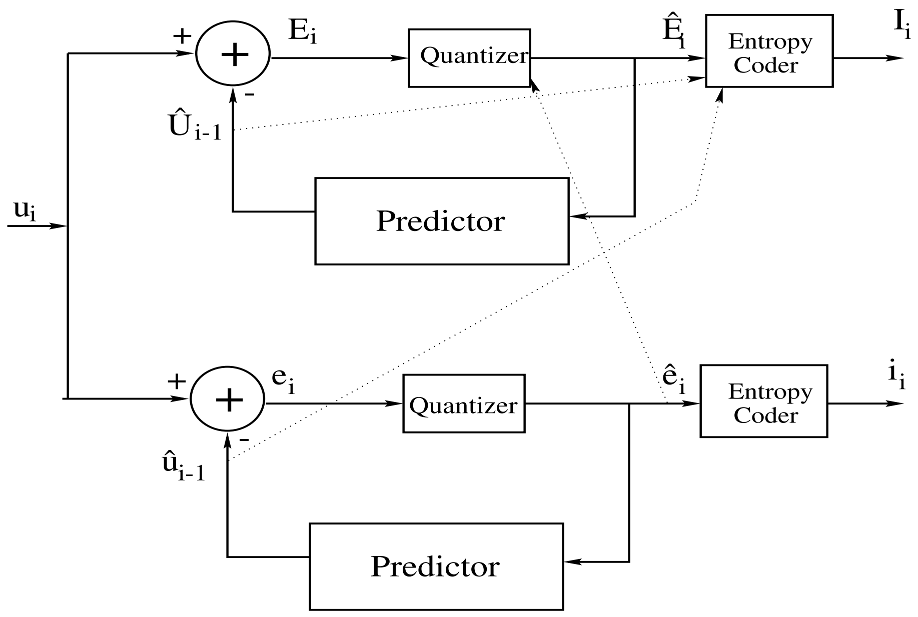

Figure 6.2: A morphable scalable predictive encoding scheme. Each layer is encoded using independent DPCM loops. The quantized prediction error of previous DPCM loops is used by the subsequent DPCM quantizer and entropy coder.

A morphable scalable predictive encoder [34] is shown in Figure 6.2. Here two separate predictive encoders are used for the two layers. Then we get descriptions l_0 and l_1. The lowest rate description l_0 is transmitted as is, however instead of transmitting l_1 a bitstream m_1 is calculated from l_0 and l_1 which will enable the decoder to *morph* l_0 into l_1. The basic operation involves generating both descriptions requiring rates R_1 and R_2. Then morph bits, m_1, are generated and transmitted as the higher layer bitstream. The decoder uses the morph bits and the lower description to generate the same "exact" higher rate description as was available at the output of the higher layer DPCM loop. The important point to note is that in this technique the high fidelity representation exactly achieves the distortion D_2. The difference with the previous technique being that now the rate required by the refinements bits will be greater than $R_2 - R_1$. The obvious advantage of this techniques is that we can achieve the exact distortions D_1 and D_2 at both points. This allows us precise control over the fidelity quality for the user.

Primary motivation for investigating morphable scalable predictive encoding schemes stems from the fact that fine grain scalability (FGS) (which falls into the category of refinable scalable predictive encoding), introduced in MPEG-4 for generating scalable video streams is extremely inefficient. With the possible potential practical impact scalable predictive encoding can have on the development of layered video encoders, it is worth exploring both the theoretical and practical implications of morphable scalable predictive encoders.

6.2 Predicting Rate Requirement for Classification Tasks

Consider two different classification tasks. Assume the first classification task has to decide the class label from $\mathcal{C}_1$ and the second decides the class label from $\mathcal{C}_2$. Assume that $|\mathcal{C}_1| >> |\mathcal{C}_2|$. An example would be two speech recognition tasks, the first one could be a continuous speech recognition (CSR) task where the candidate classes are all possible words in the task and the second could be a digit recognition task where the possible classes are in (0,1,...,9 and oh). For both these tasks MFCCs are used for recognition. However intuitively it is apparent that the amount of information required for the digit recognition task will be much lesser than that required for CSR.

When we consider these recognition tasks in the context of Distributed Speech Recognition (DSR), we can expect that the rate required for minimal recognition performance degradation in the digit recognition task will be lesser than the rate required for CSR.

The interesting question is, can we predict the recognition performance for different rates for different recognition tasks? Of course the most straightforward way to do this would be to actually encode the MFCCs at different rates and perform recognition experiments for different tasks at these different rates and determine the recognition performance. But in practice this can become impractical. Hence there is a necessity to develop a methodology which can with low computational complexity predict the rate-recognition performance for classification tasks.

Consider Equation (6.2), which was proved in Chapter 5

$$P_e^q - P_e^u \simeq \frac{I(C;X) - I(C;\hat{X})}{log|\mathcal{C}|} \tag{6.2}$$

This states that the increase in probability of error due to quantization is approximately proportional to the loss in mutual information due to quantization. Hence we can expect that mutual information (MI) loss can be used to predict the increase in probability of error, when the input data is quantized. Calculation of MI loss involves less effort than recognition experiments. Additionally, this procedure is not dependent on the particular implementation of the recognizer and hence can generalize to a wide variety of variations of speech recognizer implementations (for eg. HMM based, segmental based, ANN based).

Note that in Equation (6.2), the MI between the class labels and the MFCC vector needs to be estimated. Typically the MFCC vector consists of 13 components. Hence to estimate the MI loss, knowledge (or estimation) of the joint *pdf* $p(c, x_1, \ldots, x_13)$ is required. Usually the MFCCs are modeled as mixture Gaussians. Unfortunately no closed form expressions of MI exist for mixture Gaussians. Hence calculation of MI loss can be challenging in practice. One method to overcome this would be (as in Chapter 5) would be to use fine quantization to discretize the MFCCs, i.e., $\tilde{X} = Q(\mathbf{X})$. Then $I(C; \tilde{\mathbf{X}})$ is calculated as

$$I(\tilde{\mathbf{X}}; C) - I(\hat{\mathbf{X}}; C) = \sum_{\tilde{\mathbf{x}}} f(\tilde{\mathbf{x}}) \sum_{c} p(c|\tilde{\mathbf{x}}) log \left(\frac{p(c|\tilde{\mathbf{x}})}{q(c|\hat{\mathbf{x}})} \right) \tag{6.3}$$

The number of codewords in the vector quantizer can be chosen to ensure $P_e^{\tilde{X}} - P_e^{u} \simeq 0$, hence there would be no loss in estimation by using $\tilde{\mathbf{X}}$ instead of $\mathbf{X}$.

Another possibility would be use the Renyi entropy instead of Shannon entropy. The Renyi entropy is defined as

$$h_\alpha(\mathbf{X}) = \frac{1}{1-\alpha} log_2 \left(\int f_X^\alpha(\mathbf{x}) d\mathbf{x} \right) \tag{6.4}$$

and it converges to the Shannon entropy as $\alpha \to 1$, i.e.,

$$h(\mathbf{X}) = -\int f_X(\mathbf{x}) log_2(f_X(\mathbf{x}) d\mathbf{x} = lim_{\alpha \to 1} h_\alpha(\mathbf{X}) \tag{6.5}$$

A minimum spanning tree (MST) approach can be used to estimate the Renyi entropy [27, 26]. The spanning tree can be constructed using the MFCCs ($\mathbf{X}$). Each edge in the tree is associated with a cost. The MST is defined as the set of edges which connect all vertices in $\mathbf{X}$ with minimum total cost. i.e., the cost of the MST is

$$L_{X_N} = min_{e \in T} \sum_{e} \parallel e \parallel^\gamma \tag{6.6}$$

where e denotes an edge in the tree T, $\parallel \cdot \parallel$ and γ denote the Euclidean norm and power exponent, respectively.

Then the estimate of Renyi entropy is given by [27, 26],

$$h_\alpha(\mathbf{X}) = \left(log_2 \left(\frac{L_{X_N}}{N^\alpha} \right) - log_2(\beta_{L,\gamma} \right) / (1-\alpha) \tag{6.7}$$

where $\alpha = (d-\gamma)/d$ (d is the dimension of the vector), $\beta_{L,\gamma}$ is a density dependent constant only depending on the type of spanning tree used and γ. By setting γ to a small constant, the estimate of the Renyi entropy closely approximates the Shannon entropy.

Then the estimate of MI is given by

$$I_\alpha(C;\mathbf{X}) = \frac{1}{1-\alpha}\left(log_2\left(\frac{L_{X_N}}{N^\alpha}\right) - \sum_{c\in\mathcal{C}} log_2\left(\frac{L_{X_N}|C=c}{N^\alpha}\right)p(c)\right) \tag{6.8}$$

6.3 Optimal Transformation for Minimizing Mutual Information Loss

In Chapter 5 MI based quantization and rate allocation techniques were presented. These were shown to provide good improvements over MSE based encoders for classification applications. However the quantizer operated on the original source data. For conventional RD based systems, transform coding has proven to be an extremely powerful component of the encoder. It has enabled the use of simple scalar quantizers (SQ) on the transformed data to achieve good performance comparable to (or better than) that achieved using vector quantizers (VQ) on the original data.

It can expected that use of transform coding will be beneficial even in rate-mutual information based systems. Most rate-distortion (RD) transform coders employ orthogonal transforms, i.e, if the transform is T, then $TT^{-1} = I$. The advantage of using orthogonal transforms is that they preserve energy, i.e., the MSE introduced in the transform space will be equal to the MSE in the original space.

For mutual information, any invertible transform preserves mutual information. I.e., if the original N-dim data is X, which is associated with a class label C and $Y = TX$ is the transformed data, then

$$I(X;C) = I(Y;C) \tag{6.9}$$

if T^{-1} exists.

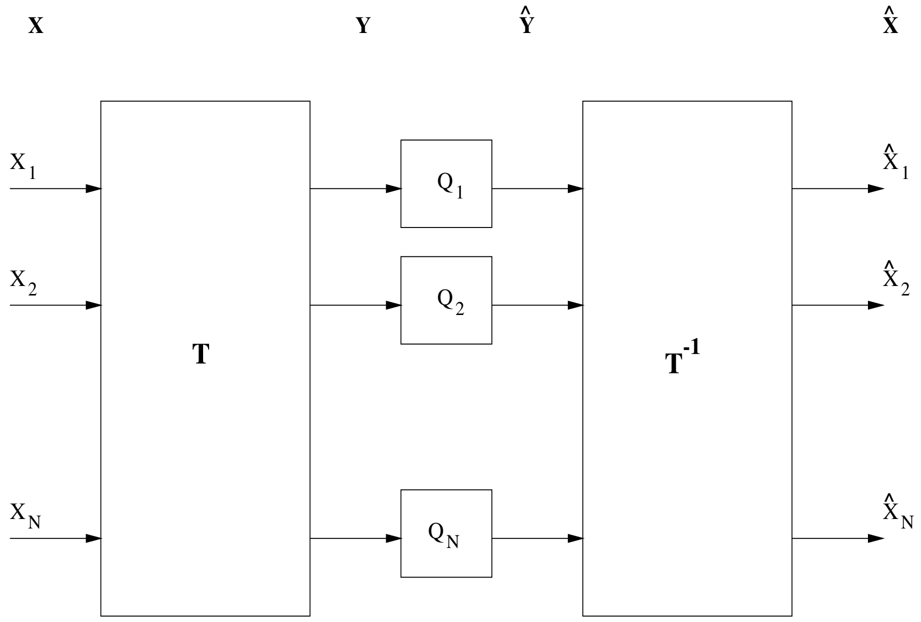

Figure 6.3: Encoder employing an invertible transform. The transform coefficients are independently quantized. The final decoded vector is obtained by inverse transforming the quantized coefficients.

Consider the source encoding system shown in Figure 6.3. The N-dim input X is transformed into Y by the transform T. The components Y_i of Y are independently quantized by scalar quantizers $Q_i(\cdot)$, i.e., $\hat{Y}_i = Q_i(Y_i)$. The quantizer vector $\hat{Y}$ is inverse transformed to get the final decoded vector $\hat{X}$. For mean square error (MSE) based systems, the quantizers $Q_i(\cdot)$ are designed to minimize $E[Y_i - \hat{Y}_i]$. For MSE based systems it has been shown that the Karhunen-Loeve (KL) transform is the optimal transform [28].

It is shown that the KL transform makes the transformed coefficients uncorrelated and a MSE system employing the KL transform minimizes the MSE distortion for a given rate.

If the system in Figure 6.3 is designed to minimize mutual information loss, then what is the optimal transform T? The optimal transform for a MSE system was a transform which made the transform coefficients uncorrelated, so is the optimal transform for minimum mutual information loss systems a transform which ensures that the transformed coefficients are independent?

Let the input vector X have two components X_1 and X_2. Consider two invertible transforms $T^{'}$ and $T^{''}$, where $T^{'}$ is the transform with makes transformed components in each class independent and $T^{''}$ is an arbitrary transform. Let $Y^{'} = T^{'}X$ and $Y^{''} = T^{''}X$. Consider optimal MI quantizers $Q_i^{'}$ and $Q_i^{''}$, $i = 1, 2$ which are used to quantize the components of $Y^{'}$ and $Y^{''}$ respectively, i.e., $\hat{Y'} = \{Q_1^{'}(Y_1^{'}), Q_2^{'}(Y_2^{'})\}$ and $\hat{Y''} = \{Q_1^{''}(Y_1^{''}, Q_2^{''}(Y_2^{''}\})$. Let the reconstructed values be $\hat{X}^{'} = T^{'-1}\hat{Y'}$ and $\hat{X}^{''} = T^{''-1}\hat{Y''}$.

The MI losses for two cases are

$$\begin{aligned} MIL_{indep} = I(X;C) - I(\hat{X}^{'};C) &= I(Y^{'};C) - I(\hat{Y}^{'};C) \\ &= I(X;C) - I(\hat{Y}^{'};C) \end{aligned} \tag{6.10}$$

and

$$\begin{aligned} MIL_{arbit} = I(X;C) - I(\hat{X}^{''};C) &= I(Y^{''};C) - I(\hat{Y}^{''};C) \\ &= I(X;C) - I(\hat{Y}^{''};C) \end{aligned} \tag{6.11}$$

$$
\begin{aligned}
I(\hat{Y}^{'};C) &= I(\hat{Y_1}^{'},\hat{Y_2}^{'};C) \\
&= I(\hat{Y_1}^{'};C) + I(\hat{Y_2}^{'};C) - I(\hat{Y_1}^{'};\hat{Y_2}^{'}|C) \\
&= I(\hat{Y_1}^{'};C) + I(\hat{Y_2}^{'};C) \qquad (6.12)
\end{aligned}
$$

This follows from the fact that $T^{'}$ makes the transformed components in each class independent hence $I(\hat{Y_1}^{'};\hat{Y_2}^{'}|C) = 0$.

$$
\begin{aligned}
I(\hat{Y}^{''};C) &= I(\hat{Y_1}^{''},\hat{Y_2}^{''};C) \\
&= I(\hat{Y_1}^{''};C) + I(\hat{Y_2}^{''};C) - I(\hat{Y_1}^{''};\hat{Y_2}^{''}|C) \qquad (6.13)
\end{aligned}
$$

Since $T^{''}$ is an arbitrary transform $I(\hat{Y_1}^{''};\hat{Y_2}^{''}|C) \geq 0$.

$$
\begin{aligned}
MIL_{indep} - MIL_{arbit} &= I(\hat{Y}^{''};C) - I(\hat{Y}^{'};C) \\
&= I(\hat{Y_1}^{''};C) + I(\hat{Y_2}^{''};C) - I(\hat{Y_1}^{''};\hat{Y_2}^{''}|C) \\
&\quad -I(\hat{Y_1}^{'};C) - I(\hat{Y_2}^{'};C) \\
&\leq I(\hat{Y_1}^{''};C) + I(\hat{Y_2}^{''};C) - I(\hat{Y_1}^{'};C) - I(\hat{Y_2}^{'};C) \qquad (6.14)
\end{aligned}
$$

if $I(\hat{Y_1}^{''};C) + I(\hat{Y_2}^{''};C) = I(\hat{Y_1}^{'};C) - I(\hat{Y_2}^{'};C)$ then $MIL_{indep} \leq MIL_{arbit}$.

The intuition is that when the transform $T^{'}$ is used, each component of the transformed data provides completely extra information about the class labels. However when an arbitrary transform $T^{''}$ is used the information provided by the different components is not completely independent. Hence it can be expected that the total amount of information provided by the transform coefficients about the class when transform $T^{''}$ is used will be less than when the transform $T^{'}$ is used.

6.4 Application Specific Distributed Source Coding

Consider a sensor network where several sensors acquire observations about a target. The signals acquired will be correlated due to the spatial proximity of the sensors. To reduce the bandwidth required for transmission we can take advantage of this correlation. The observations can be jointly encoded. However this requires inter-sensor communication, which is not desirable from an energy point of view, since transmission tends to dominate the power requirements in sensor networks. The rule of thumb is processing is preferable to transmission. Consider that we have encode two correlated observations X and Y. Joint encoding requires a rate $H(X, Y)$ for lossless transmission. This requires the encoder know both X and Y. However the surprising result by Slepian-Wolf is that with the same rate, $H(X, Y)$, the two observations can be independently encoded, provided that they are decoded at a central location. This is shown in Figure 6.4. Similarly the Wyner-Ziv theorem provides the rate-distortion function for distributed encoding when lossy encoding instead of lossless encoding is employed.

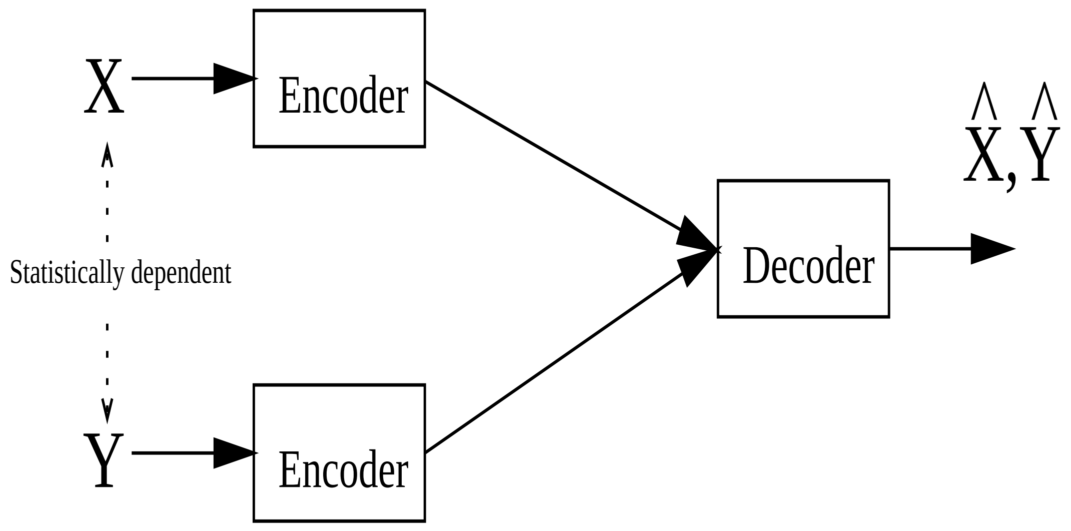

Figure 6.4: Distributed encoding of two statistically dependent sources. X and Y are independently encoded but jointly decoded at a central decoder.

These two techniques, Slepian-Wolf and Wyner-Ziv either attempt to lossless transmit the data or attempt to minimize the distortion in the decoded data. Consider a sensor network where the system attempts to identifies the type of the vehicle traveling on the road. Here the objective is to first decide if a vehicle is traveling on the road and if so, then to identify the type of the vehicle. This is a classification task which makes use of the observations acquired by several different sensors. It is not required that the sensors transmit the entire acquired data to the central location making the classification decision. It is sufficient that the sensors transmit sufficient information required to reliable identify the presence of the vehicle(s) and identify it(them). Similar to the distributed classification problem discussed before, now this is a system which involves distributed coding and distributed classification. The conventional distributed coding algorithms [45] only make use of the correlation between $X_i, i = 1, \ldots, N$, where X_i are observations acquired at N different sensors. However for classification tasks the information provided by each of the X_i about the class label is relevant not the individual X_i's themselves. Hence it is obvious that the Slepian-Wolf/Wyner-Ziv framework is not necessarily optimal for the *distributed-source-distributed-classification* problem. The optimal rate-constrained solution would be to transmit sufficient information from each sensor subject to a bandwidth constraint which would ensure that the classification at the central location is minimally affected.

One method to do this would be consider the mutual information between the observation at a sensor and the class labels that need to be discriminated among. Consider two sensors that acquire observations X and Y about a phenomena. The applications goal is to estimate a class label C from these observations. Based on the minimal mutual information loss framework the required optimal encoders would be quantizers that minimize $I(X;C) - I(\hat{X};C)$ at the first sensor, and minimize $I(Y;C|\hat{X}) - I(\hat{Y};C|\hat{X})$ at the second sensor. Obviously this requires complete knowledge of the joint pdf $p(x,y,c)$. In general even if this *pdf* is known or can be estimated it would require significant inter-sensor communication, which we want to avoid. Hence it is required to develop a framework similar to Slepian-Wolf/Wyner-Ziv, which address the problem of distributed-source-distributed-classification problem. The question to be answered is, is it possible to attain the same performance in a distributed mutual information source coding system without communication between sensors as that could be achieved with communication between sensors? The advantage of this would be

(1) Better system performance can be achieved, since we are concerned only about the applications objective (2) The rate requirement can be significantly reduced (3) Only relevant information will be transmitted

The more general question thats needs to addressed is, can distributed mutual information based systems also be incorporated into the same framework as MSE based distributed systems. This would enable development of optimal distributed-source-distributed-classification systems which would ensure that loss due to quantization in a distributed encoding system have minimal effect on the applications final desired result, i.e., classification.

Reference List

[1] Y. Abe, H. Itsui, Y. Maruta, and K. Nkajima. A two-stage speech recognition method with an error correction model. In *Eurospeech '99*, Budapest, September 1999.

[2] Anuradha Aiyer and Robert M. Gray. A fast, table-lookup algorithm for classifying document images. In *ICIP 1999*, pages 590–594, Kobe, Japan, October 1999.

[3] Anuradha Aiyer and Robert M. Gray. Fast classification using weighted distortion. In *ICIP 2000*, Vancouver, Canada, September 2000.

[4] S. Bayer. Embedding speech in web interfaces. In *Proc ICSLP*, pages 1684–1688, Philadelphia, PA, October 1996.

[5] F. Behet, R. de Mori, and G. Subsol. Very large vocabulary proper name recognition for directory assistance. In *IEEE International Conference on Automatic Speech Recognition and Understanding 2001*, pages 222 –225, 2001.

[6] C. D. Bei and R. M. Gray. An improvement of the minimum distortion encoding algorithm for vector quantization. *IEEE Trasactions on Communications*, 33(10):1132–1133, October 1985.

[7] Jon Louis Bentley. Multidimensional binary search trees used for associative searching. In *Communications of the ACM*, volume 23, September 1975.

[8] Alexis Bernard and Abeer Alwan. Source and channel coding for remote speech recognition over error-prone channel. In *ICASSP 2001*, volume 4, 2001.

[9] Navin Chaddha, Philip Chou, and Robert Gray. Constrained and recursive hierarchical table-lookup vector quantization. In *Proc. IEEE Data Compression Conference (DCC)*, 1996.

[10] Navin Chaddha, Mohan Vishwanath, and Philip Chou. Hierarchical vector quantization of perceptual weighted block transform. In *Proc. IEEE Data Compression Conference (DCC)*, pages 3–12, 1995.

[11] P. C. Chang, J. May, and R. M. Gray. Hierarchical vector quantizers with table-lookup encoders. In *Proceedings 1985 IEEE International Conference on Communications*, volume 3, pages 1452–1455, 1985.

[12] Dan Chazan, Ron Hoory Gilad Cohen, and Meir Zibulski. Speech reconstruction from Mel-frequency cepstral coefficients and pitch frequency. In *IEEE ICASSP 2000*, 2000.

[13] Joe C. Chen, Kung Yao, and Ralph E. Hudson. Source localization and beam forming. In *IEEE Signal Processing Magazine*, pages 30–39. IEEE, March 2002.

[14] P. A. Chou, T. Lookabaugh, and R. M. Gray. Entropy constrained vector quantization. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, ASSP-37:31–42, January 1989.

[15] Paolo Coletti and Marcello Federico. A two-stage speech recognition method for information retrival applications. In *Eurospeech '99*, Budapest, September 1999.

[16] Thomas M. Cover and Joy A. Thomas. *Elements of Information Theory.* Wiley Series in Telecommunications. John Wiley & Sons, New York, NY, USA, 1991.

[17] CSLU. Names 1.1, the CSLU OGI names corpus. http://cslu.cse.ogi.edu /corpora/names.

[18] S. B. Davis and P. Mermelstein. Comparison of parametric representations for monosyllabic word recognition in continuously spoken sentences. *IEEE Acoust., Speech, Signal Processing*, ASSP-28(4):357–366, 1980.

[19] V. V. Digalakis, L. G. Neumeyer, and Manolis Perakakis. Quantization of cepstral parameters for speech recognition over the world wide web. *IEEE Journal on Selected Areas in Communication*, 17(1):82–90, January 1999.

[20] W. Equitz and T. Cover. Successive refinement of information. *IEEE Transactions on Information Theory*, 37(2):269–275, March 1991.

[21] Speech processing, transmission and quality aspects (STQ); distributed speech recognition; front-end feature extraction algorithm; compression algorithms. Technical Report Standard ES 201 108, European Telecommunications Standards Institute (ETSI), April 11 2000.

[22] J. H. Friendman, J. L. Bentley, and R. A. Finkel. An algorithm for finding best matches in logarithmic expected time. *ACM Transactions on Mathematical Software*, 3(3):209–266, September 1977.

[23] Yuqing Gao, B. Ramabhadran, J. Chen, H. Erdogan, and M. Picheny. Innovative approaches for large vocabulary name recognition. In *ICASSP 2001*, volume 1, pages 53–56, 2001.

[24] Sun Han and Shun ichi Amari. Statistical inference under multiterminal data compression. *IEEE Transcations on Information Theory*, 44:2300–2324, October 1998.

[25] Qiang Huo, Chorkin Chan, and Chin Hui Lee. Bayesian adaptive learning of the parameters of hidden Markov model for speech recognition. *IEEE Transactions on Speech and Audio Processing*, 3(5):334–345, September 1995.

[26] Alfred O. Hero III, Bing Ma, Olivier J.J. Michel, and John Gorman. Applications of entropic spanning graphs. *IEEE Signal Processing Magazine*, 19(5):85–95, September 2002.

[27] Alfred O. Hero III and Olivier Michel. Asymptotic theory of greedy approximations to minimal k-point random graphs. *IEEE Transactions on Information Theory*, 45:1921–1939, September 1999.

[28] Anil K. Jain. *Fundamentals of Digital Image Processing.* Prentice Hall, 1997.

[29] Jean-Claude Junqua. SmarTspelL^{TM}: A multipass recognition system for name retrieval over the telephone. *IEEE Transactions on speech and audio processing*, 5(2):173–182, March 1997.

[30] Leonard Kleinrock. Nomadic computing and communication. In *The Unpredictable Certainty Information Infrastructure Through 2000.* National Research Council, 2000. *http://bob.nap.edu/html/whitepapers/ch-40.html.*

[31] Chin-Hui Lee and Jean-Luc Guavain. Bayesian adaptive learning and map estmiation of hmm. In C.-H. Lee and F. Soong, editors, *Automatic Speech and Speaker Recognition, Advanced Topics*, pages 83–107. Kluwer Academic Publishers, Boston, Mass., 1996.

[32] Dan Li, Kerry D. Wong, Yu Hen Hu, and Akbar Sayeed. Detection, classification and tracking of targets. In *IEEE Signal Processing Magazine*, pages 17–29. IEEE, March 2002.

[33] Michael Lightstone and Sanjit K. Mitra. Optimal variable-rate mean-gain-shape vector quantization for image coding. *IEEE Transactions on Circuits and Systems for Video Technology*, 6:660–668, December 1996.

[34] J. Macnicol, J. Arnold, and M. Frater. Scalable video coding by stream morphing. *IEEE Transactions on Circuits and Systems for Video Technology*, 15:306–319, Feb 2005.

[35] P. Mayorga, R. Lamy, and L. Besacier. Recovering of packet loss for distributed speech recognition. In *Eusipco 2002*, Toulouse, France, September 2002.

[36] B. Milner and S. Semnani. Robust speech recognition over IP networks. In *ICASSP 2000*, Istanbul, Turkey, June 2000.

[37] Nader Moayeri, David L. Neuhoff, and Wayne E. Stark. Fine-coarse vector quantization. *IEEE Transactions on Signal Processing*, 39(7):1503–1515, July 1991.

[38] Hy Murveit, John Butzberger, Vassilios Digalakis, and Mitch Weintraub. Large vocabulary dictation using SRIs $DECIPHER^{TM}$ speech recognition system: Progressive search techniques. In *ICASSP-93*, volume II, pages 319–322, April 1993.

[39] Y. Normandin, R. Cardin, and R. De Mori. High-performance connected digit recognition using maximum mutual information estimation. *IEEE Trans. on Speech and Audio Processing*, 2(2), 1994.

[40] K.L. Oehler and R.M. Gray. Combining image compression and classification using vector quantization. *IEEE Trans. Pattern Analysis and Machine Learning*, 17(5):461–473, May 1995.

[41] Antonio Ortega, Baltasar Beferull-Lozano, Naveen Srinivasamurthy, and Hua Xie. Compression for recognition and content-based retrieval. In *Proc. of Eusipco*, Tampere,Finland, September 2000.

[42] K. O. Perlmutter, R. M. Gray, K. L. Oehler, and R.A. Olsen. Bayes risk weighted tree-structured vector quantization with posterior estimation. In *DCC 1994*, pages 274–283, March 1994.

[43] K.O. Perlmutter, S.M. Perlmutter, R.M. Gray, R.A. Olsen, and K.L. Oehler. Bayes risk weighted vector quantization with posterior estimation for image compression and classification. *IEEE Trans. Image Processing*, 5:347–360, February 1996.

[44] Kris Popat and Rosalind W. Picard. Cluster-based probability model applied to image restoration and compression. In *ICASSP-94*, volume V, pages 381–384, Adelaide, Australia, April 1994.

[45] Sandeep Pradhan and Kannan Ramchandaran. Distributed source coding using syndromes (DISCUS). In *DCC, Data Compression Conference*, Snowbird, Utah, March 1999.

[46] Economist print edition. Just talk to me. In *Economist, Technology Quarterly*. The Economist Newspaper Limited, December 6^{th} 2001.

[47] L. R. Rabiner. A tutorial on hidden Markov models and selected application in speech recognition. In *Proc IEEE*, volume 77, pages 257–286, February 1989.

[48] Bhiksha Raj, Michael L. Seltzer, , and Richard M. Stern. Robust speech recognition: The case for restoring missing features. In *CRAC 2001*, Aalborg, Denmark, September 2001.

[49] Tenkasi Ramabadran, Jeff Meunier, Mark Jasiuk, and Bill Kushner. Enhancing distributed speech recognition with back-end speech reconstruction. In *Eurospeech 2001*, Aalborg, Denmark, September 2001.

[50] V. Ramasubramanian and Kuldip K. Paliwal. Fast k-dimensional tree algorithms for nearest neighbor search with application to vector quantization encoding. *IEEE Transactions on Signal Processing*, 40(3):518–530, March 1992.

[51] G. N. Ramaswamy and P. S. Gopalakrishnan. Compression of acoustic features for speech recognition in network environments. In *IEEE ICASSP 1998*, pages 977–980, 1998.

[52] Bixio Rimoldi. Successive refinement of information: Characterization of the achievable rates. *IEEE Transactions on Information Theory*, 40(1):253–259, January 1994.

[53] E. A. Riskin. Optimum bit allocation via the generalized BFOS algorithm. *IEEE Transactions on Information Theory*, 37(2):400–402, March 1991.

[54] Eve Riskin, Constantinos Boulis, Scott Otterson, and Mari Ostendorf. Graceful degradation of speech recognition performance over lossy packet networks. In *Eurospeech 2001*, Aalborg, Denmark, September 2001.

[55] Kenneth Rose and Shankar Regunathan. Towards optimal scalability in predictive video coding. In *ICIP 98*, pages 929–933, 1998.

[56] Jurgen Schurmann. *Pattern Classification - A Unified View of Statistical and Nueral Approaches*. John Wiley & Sons, Inc., 1996.

[57] Abhinav Sethy, Shrikanth Narayanan, and S. Parthasarathy. Syllable-based recognition of spoken names. In *ISCA Pronunciation Modeling and Lexicon Adaptation Workshop*, 2002.

[58] John Shore and Robert M. Gray. Minimum cross-entropy pattern classification and cluster analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-4:11–17, January 1982.

[59] Raghavendra Singh and Antonio Ortega. Erasure recovery in predictive coding enviornments using multiple description coding. In *IEEE Workshop on Multimedia Signal Processing*, 1999.

[60] Naveen Srinivasamurthy, Shrikanth Narayanan, and Antonio Ortega. Use of model transformations for distributed speech recognition. In *ISCA ITR-Workshop 2001 (Adaptation Methods for Speech Recognition)*, Sophia-Antipolis, France, August 2001.

[61] Naveen Srinivasamurthy and Antonio Ortega. Joint compression-classification with quantizer/classifier dimension mismatch. In *VCIP 2001*, San Jose, CA, January 2001.

[62] Naveen Srinivasamurthy and Antonio Ortega. Reduced complexity quantization under classification constraints. In James A. Storer and Martin Cohn, editors, *Proc. IEEE Data Compression Conference (DCC)*, pages 402–411, 2002.

[63] Naveen Srinivasamurthy, Antonio Ortega, and Shrikanth Narayanan. Efficient scalable speech compression for scalable speech recognition. In *Eurospeech 2001*, Aalborg, Denmark, September 2001.

[64] Naveen Srinivasamurthy, Antonio Ortega, and Shrikanth Narayanan. Efficient scalable encoding for distributed speech recognition. *Speech Communication*, 48:888–902, August 2006.

[65] Naveen Srinivasamurthy, Antonio Ortega, Qifeng Zhu, and Abeer Alwan. Towards efficient and scalable speech compression schemes for robust speech recognition applications. In *ICME 2000*, New York, NY, July 2000.

[66] Naftali Tishby, Fernando C. Pereira, and William Bialek. The information bottleneck method. In *37-th Annual Allerton Conference on Communication, Control and Computing*, pages 368–377, 1999.

[67] Julius T. Tou and Rafael C. Gonzalez. *Pattern Recognition Principles*. Addison-Wesley Publishing Company, 1974.

[68] Vijitha Weerackody, Wolfgang Reichl, and Alexandros Potamianos. An error-protected speech recognition system for wireless communications. *IEEE Transactions on Wireless Commumications*, 1(2):282–291, April 2002.

[69] Mark Weiser. The computer for the 21st century. In *Scientific American*, pages 94–104. September 1991.

[70] Hua Xie and Antonio Ortega. Feature representation and compression for content-based image retrieval. In *VCIP 2001*, San Jose, CA, January 2001.

[71] Q. Xie, C.A. Laszlo, and R.K. Ward. Vector quantization technique for nonparametric classifier design. *IEEE Transactions, Pattern Analysis and Machine Intelligence*, 15(12):1326–1330, Dec 1993.

[72] Kaisheng Yao, Jingdong Chen, Kuldip K. Paliwal, and Satoshi Nakamura. Feature extraction and model based compensation for noisy speech recognition evaluated on aurora 2 task. In *Eurospeech 2001*, Aalborg, Denmark, September 2001.

[73] Steve Young, Dan Kershaw, Julian Odell, Dave Ollason, Valtcho Valtchev, and Phil Woodland. The HTK book (for htk version 3.0). (htk.eng.cam.ac.uk/prot-docs/HTKBook/htkbook.html), July 2000.

[74] Qifeng Zhu and Abeer Alwan. An efficient and scalable 2D DCT-based feature coding scheme for remote speech recognition. In *ICASSP 2001*, volume 1, 2001.