

# USING OBSERVATION UNCERTAINTY FOR ROBUST SPEECH RECOGNITION

A Thesis  
Presented to  
The Academic Faculty

by

**Jon A. Arrowood**

In Partial Fulfillment  
of the Requirements for the Degree  
Doctor of Philosophy

School of Electrical and Computer Engineering  
Georgia Institute of Technology  
November 2003

Copyright © 2003 by Jon A. Arrowood

# USING OBSERVATION UNCERTAINTY FOR ROBUST SPEECH RECOGNITION

Approved by:

\_\_\_\_\_  
Mark A. Clements, Advisor

\_\_\_\_\_  
Chin-Hui Lee

\_\_\_\_\_  
Nikil S. Jayant

\_\_\_\_\_  
Eric A. Carlen  
(School of Mathematics)

\_\_\_\_\_  
David V. Anderson  
(Computer Engineering Group)

Date Approved 2003 Nov 24

*To Caryn, Joey, Bobbie, and J.L.,*

*without whom this would never have been written.*

## ACKNOWLEDGEMENTS

Writing a dissertation requires far more than one author, so here I'd like to mention a few of those people who helped, but were unable to get their names on the front cover. First and foremost, I must thank my advisor Prof. Mark Clements. It takes a special person to allow a student to hang around as long as I have. I hope he thinks this research has been worth the considerable amount of effort he put forth to make sure I was supported while I explored more avenues than was necessary!

Many fellow Ph.D. students aided in my research and understanding: Pete Cardillo, Robert Morris, Xin Zhong, my brother Joe, my wife Caryn, Brian Delaney, Halûk Aydinoglu. A special Thank You goes to the many students who over the years ran the CSIP seminar series. I learned a lot, and appreciate the organization work they did.

I have also had the opportunity to work with an array of people that have made an impression on me. Mike Miller, Gary Elko, and all the guys at Fast-Talk, just to name a few.

Of course, it's not all been work here at Georgia Tech. And there are many people here who I can blame for how long I've been in grad school. Ram Rao, Matt Cobb, Chris Lancianci, Susy Meier, Caryn Riley, the Northside Tavern, and many more. Who knew the "song of the 80's" trivia list would bring the entire CSIP department to a standstill for two weeks?

Finally, I have to thank my family. Joey has always been there for me, and my parents have been the best teachers I've had. And what at first was company for coffee during eighteen-hour work days has become the most important person to me. Caryn Riley, thank you for, well, everything!

# TABLE OF CONTENTS

|   |           |
|---|-----------|
| DEDICATION . . . . .  | iii       |
| ACKNOWLEDGEMENTS . . . . .  | iv        |
| LIST OF TABLES . . . . .  | viii      |
| LIST OF FIGURES . . . . .   | ix        |
| SUMMARY . . . . .   | xi        |
| <b>I INTRODUCTION . . . . .</b>                                   | <b>1</b>  |
| <b>II BACKGROUND . . . . .</b>                                    | <b>5</b>  |
| 2.1 Hidden Markov Model LVCSR . . . . .                           | 5         |
| 2.2 Feature Extraction . . . . .                                  | 9         |
| 2.3 Robust Speech Recognition . . . . .                           | 11        |
| 2.3.1 Estimating A Distortion Model . . . . .                     | 13        |
| 2.3.2 Feature Transformation . . . . .                            | 13        |
| 2.3.3 Model Compensation . . . . .                                | 14        |
| 2.4 Baseline Recognition System . . . . .                         | 18        |
| <b>III THE UNCERTAIN OBSERVATION DECODING ALGORITHM . . . . .</b> | <b>21</b> |
| 3.1 Problem Description . . . . .                                 | 22        |
| 3.2 Motivating Example . . . . .                                  | 23        |
| 3.3 General Formulation . . . . .                                 | 26        |
| 3.4 Interpretations . . . . .                                     | 28        |
| 3.4.1 Expected Value Interpretation . . . . .                     | 29        |
| 3.4.2 Bayes Decision Theory Interpretation . . . . .              | 30        |
| 3.5 Special Cases . . . . .                                       | 32        |
| 3.5.1 Known Undistorted Feature Vectors . . . . .                 | 33        |
| 3.5.2 Completely Unknown Feature Vectors . . . . .                | 33        |
| 3.5.3 Gaussian Observation PDFs . . . . .                         | 35        |
| 3.5.4 Gaussian Mixture Observation PDFs . . . . .                 | 37        |
| 3.5.5 Uniform Observation PDFs . . . . .                          | 37        |
| 3.6 Complexity Analysis . . . . .                                 | 38        |

|           |   |           |
|-----------|---|-----------|
| 3.7       | Experiments with UO Decoding . . . . .  | 43        |
| 3.7.1     | The Recognition Engine . . . . .  | 43        |
| 3.7.2     | Experimental Setup . . . . .  | 44        |
| 3.7.3     | Oracle Experiment 1: Stationary distortion . . . . .  | 44        |
| 3.7.4     | Oracle Experiment 2: Time-varying distortion . . . . .  | 47        |
| 3.7.5     | Oracle Experiment 3: Erroneous covariance estimates . . . . .                                     | 49        |
| 3.7.6     | Oracle Experiment 4: Comparison of full and diagonal observation<br>covariance matrices . . . . . | 50        |
| 3.8       | Comparison to Related Work . . . . .  | 54        |
| 3.8.1     | The Missing Data Problem . . . . .  | 54        |
| 3.8.2     | Stochastic Weighted Viterbi Decoding . . . . .  | 56        |
| 3.8.3     | Uncertainty Decoding . . . . .  | 56        |
| 3.9       | Summary and Discussion . . . . .  | 58        |
| <b>IV</b> | <b>UO DECODING OF NOISY SPEECH . . . . .</b>  | <b>61</b> |
| 4.1       | Analysis of Feature Extraction in Additive Noise . . . . .  | 61        |
| 4.1.1     | Nonstationary background noise estimation . . . . .   | 64        |
| 4.2       | Extending feature transformation to PDF feature extraction . . . . .                              | 65        |
| 4.2.1     | Application to Stationary Noise . . . . .   | 67        |
| 4.3       | Direct estimation of observation PDFs . . . . .   | 69        |
| 4.3.1     | Markov Chain Monte-Carlo (MCMC) Enhancement . . . . .   | 69        |
| 4.3.2     | MCMC Non-Gaussian methods . . . . .   | 72        |
| 4.3.3     | Other techniques . . . . .  | 73        |
| 4.4       | Bursty Noise . . . . .  | 74        |
| 4.5       | Summary and Discussion . . . . .  | 74        |
| <b>V</b>  | <b>APPLICATION OF OBSERVATION UNCERTAINTY TO DISTRIBUTED<br/>SPEECH RECOGNITION . . . . .</b>     | <b>76</b> |
| 5.1       | MFCC VQ Adaptation . . . . .  | 78        |
| 5.1.1     | Problem Description . . . . .   | 78        |
| 5.1.2     | Two Proposed Solutions . . . . .  | 79        |
| 5.1.3     | Experiment and Results . . . . .  | 83        |
| 5.2       | Speech Coder Adaptation . . . . .   | 87        |

|                   |  |            |
|-------------------|--|------------|
| 5.2.1             | Problem Description . . . . .                                  | 87         |
| 5.2.2             | Two Proposed Solutions . . . . .                               | 87         |
| 5.2.3             | Experiment and Results . . . . .                               | 90         |
| 5.3               | Handling Packet-Lossy Channels . . . . .                       | 95         |
| 5.3.1             | Problem Description . . . . .                                  | 95         |
| 5.3.2             | Application of UO to Lossy Channels . . . . .                  | 97         |
| 5.3.3             | PDF Feature Extraction . . . . .                               | 98         |
| 5.3.4             | Experiments and Results . . . . .                              | 99         |
| 5.4               | Summary and Discussion . . . . .                               | 100        |
| <b>VI</b>         | <b>UO TECHNIQUES APPLIED TO HMM TRAINING . . . . .</b>         | <b>104</b> |
| 6.1               | Motivation . . . . .   | 104        |
| 6.2               | Review of Baum-Welch Training . . . . .                        | 105        |
| 6.3               | Baum-Welch Reestimation with Observation Uncertainty . . . . . | 108        |
| 6.4               | Analysis . . . . .   | 111        |
| 6.5               | Evaluation on the Aurora2 Database . . . . .                   | 114        |
| 6.6               | Summary and Discussion . . . . .                               | 117        |
| <b>VII</b>        | <b>CONCLUSIONS . . . . .</b>                                   | <b>119</b> |
| 7.1               | Contributions . . . . .  | 119        |
| 7.2               | Observations . . . . .   | 120        |
| 7.3               | Future Work . . . . .  | 121        |
| <b>APPENDIX A</b> | <b>— PHONEME SET . . . . .</b>                                 | <b>123</b> |
| <b>REFERENCES</b> | <b>. . . . .</b>   | <b>124</b> |
| <b>VITA</b>       | <b>. . . . .</b>   | <b>129</b> |

## LIST OF TABLES

|           |  |     |
|-----------|--|-----|
| Table 1   | Gender-dependent phoneme recognition experiments. . . . .  | 12  |
| Table 2   | Baseline error rates for phoneme recognition experiments. . . . .  | 20  |
| Table 3   | Computational overhead for UO Decoding . . . . .   | 41  |
| Table 4   | Computational burden of UO decoding in an additive noise environment with no language modeling. . . . .              | 42  |
| Table 5   | Covariance matrix representations of speech and noisy features . . . . .   | 51  |
| Table 6   | Oracle 4 MFCC implementation speeds. . . . .   | 53  |
| Table 7   | Digit recognition results for UO decoding in stationary additive noise. . .  | 68  |
| Table 8   | Phoneme error rates in stationary additive noise . . . . .   | 71  |
| Table 9   | Phoneme error rates in stationary additive noise . . . . .   | 72  |
| Table 10  | Comparison of MFCC VQ recognition systems. . . . .   | 87  |
| Table 11  | Comparison of recognition methods for coded speech. . . . .  | 94  |
| Table 12  | The set of Gilbert Model state transition and frame loss probabilities used for experiments in this section. . . . . | 99  |
| Table 13  | ASR Performance on packet-lossy data . . . . .   | 100 |
| Table A.1 | Phoneme set used for the phoneme loop grammar recognition experiments performed throughout this thesis. . . . .      | 123 |

## LIST OF FIGURES

|           |   |    |
|-----------|---|----|
| Figure 1  | Topological model of HMM-based speech recognition . . . . .   | 6  |
| Figure 2  | HMM lattice decoding used for trace projection speech recognition. . . . .  | 9  |
| Figure 3  | Flowchart for MFCC generation . . . . .   | 10 |
| Figure 4  | Block diagram of speech production mismatches . . . . .   | 11 |
| Figure 5  | Example of feature transformation. . . . .  | 15 |
| Figure 6  | Block diagram of a feature transformation operation. . . . .  | 15 |
| Figure 7  | Example of model compensation. . . . .  | 17 |
| Figure 8  | Block diagram of the model compensation procedure. . . . .  | 17 |
| Figure 9  | Standard feature transformation. . . . .  | 24 |
| Figure 10 | Computing a standard state output probability . . . . .   | 25 |
| Figure 11 | Example of feature transformation with two equiprobable points. . . . .   | 25 |
| Figure 12 | Computing a state output probability given two equiprobable observations  | 26 |
| Figure 13 | Example of decoding with $m$ equiprobable points. . . . .   | 26 |
| Figure 14 | Computing the output probability of a stochastic feature . . . . .  | 28 |
| Figure 15 | Block diagram of the Uncertain Observation decoding procedure. . . . .  | 28 |
| Figure 16 | Block diagram of “Oracle” recognition experiments. . . . .  | 45 |
| Figure 17 | Oracle 1 experiment results on TIMIT. . . . .   | 47 |
| Figure 18 | Oracle 2 experiment results on TIMIT with time varying distortion. . . . .  | 48 |
| Figure 19 | Oracle 3 experiment results on TIMIT with erroneous variance information.   | 50 |
| Figure 20 | Oracle 4 experiment comparing feature covariance matrix representation.   | 52 |
| Figure 21 | Example of a reliability mask used for Missing Feature robust ASR . . . . .   | 55 |
| Figure 22 | Example of a fuzzy mask used for Missing Feature robust ASR . . . . .   | 55 |
| Figure 23 | Degenerate example of Uncertainty Decoding. . . . .   | 57 |
| Figure 24 | Plot showing local SNR across time. . . . .   | 62 |
| Figure 25 | Plot showing local SNR across frequency. . . . .  | 63 |
| Figure 26 | Time-Frequency comparison of clean and noisy speech. . . . .  | 64 |
| Figure 27 | Block diagram of the Gibbs Sampler to implement the autoregressive constrained Markov Chain Monte-Carlo speech enhancement algorithm. . . . . | 70 |
| Figure 28 | Machine gun bursty noise waveform. . . . .  | 74 |

|           |   |     |
|-----------|---|-----|
| Figure 29 | Flowchart of a speech compression system . . . . .                                | 76  |
| Figure 30 | VQ-UO training and classification visualization . . . . .                         | 80  |
| Figure 31 | Training UO parameters for VQ-UO-Internal systems . . . . .                       | 81  |
| Figure 32 | Flowchart of the $C_3$ -Internal-UO system incorporating UO . . . . .             | 81  |
| Figure 33 | Training UO parameters for $C_3$ -External-UO systems . . . . .                   | 82  |
| Figure 34 | Flowchart of the $C_3$ -External-UO system incorporating UO . . . . .             | 82  |
| Figure 35 | Performance of the $C_3$ -Internal-UO and $C_3$ -External-UO algorithms . . . . . | 85  |
| Figure 36 | Histograms of MFCC cepstral distortion . . . . .                                  | 86  |
| Figure 37 | Motivation for the $C_4$ -External-2-State algorithm . . . . .                    | 89  |
| Figure 38 | Histogram of MELP coder cepstral distortion . . . . .                             | 92  |
| Figure 39 | Digit performance of $C_4$ -location UO decoding of coded speech . . . . .        | 94  |
| Figure 40 | Phoneme performance of $C_4$ -location UO decoding of coded speech . . . . .      | 95  |
| Figure 41 | Gilbert model of burst packet loss . . . . .                                      | 97  |
| Figure 42 | Comparison of Viterbi alignments used in UO training . . . . .                    | 115 |
| Figure 43 | Training results . . . . .  | 117 |

## SUMMARY

This research presents a new technique for adapting Hidden Markov Model (HMM) automatic speech recognition (ASR) systems when there is a mismatch between training and testing conditions. In standard ASR systems, each observation vector is assumed completely reliable, and implicitly weighted equally during decoding. The new method, referred to as the Uncertain Observation technique, leverages the knowledge that all feature extraction has an inherent amount of uncertainty. In noisy environments, for example, different regions in the time frequency plane have vastly different local signal to noise ratios, and thus significantly different reliabilities. The Uncertain Observation decoder derived for this research presents a strong probabilistic framework, integrating the likelihood distribution of each feature vector into the decoding process. This results in a continuously adaptive recognition system that integrates Bayes Predictive Classification into a HMM recognition system.

The technique compares favorably in stationary noise to model compensation methods such as Parallel Model Combination (PMC), and outperforms simpler feature transformation. In time-varying environments, for which the algorithm is designed, Uncertain Observation techniques outperform both model compensation and time-varying feature transformations. Uncertain Observation methods are further applied to packet lossy channels, compressed speech, and feature vector quantization, showing a promising method that can be integrated into distributed speech recognition systems for robust performance. Finally, Uncertain Observation methods are applied during training to aid speech model creation or adaptation when the available data contains a distortion.

# CHAPTER I

## INTRODUCTION

Although automatic speech recognition has made noticeable strides, it has yet to make the significant impact other recently developed technologies have made on people's daily lives. One reason is that while experimental systems in the laboratory are capable of serving as good man-machine interfaces, they have yet to demonstrate the ability to be robust and versatile in actual applications. One major deterrent to widespread application is that today's systems have trouble recognizing speech captured in an environment that is different from which it was trained.

Many factors contribute to a mismatch between training and operating environments. Differences can be due to the speaker (such as vocal tract length, dialect, and emotional state) or to the audio capture conditions (such as room reverberation, transducer nonlinearities, transmission channel effects, compression artifacts, and background noise). Various algorithms have been discussed in the literature to compensate for many of these effects, operating as either a feature transformation or as model compensation.

There are many techniques in the Automatic Speech Recognition (ASR) community that address the issue of additive background noise. In general, once a feature set has been chosen, robust techniques fall into two categories, feature transformation or model compensation. Feature transformation aims to adjust the observation sequence to approximate what it would be if the noise were not present during audio capture. An example is the well known spectral subtraction algorithm [7] or the more recent SPLICE algorithm [14]. Model compensation is a more advanced method, that aims instead to adjust the speech models so that they appear they were trained using speech collected in the current environment. The best known example of this class of algorithms is Parallel Model Combination, (PMC) [24]. Each of these two general approaches has pros and cons. Feature transformations are easier to perform, requiring no changes to the recognition engine, and can be quickly

adapted to changes in the background environment. Model compensation methods are better at improving recognition accuracy, as it is possible to update both the model means and variances. It has been difficult, however, to incorporate model compensation techniques into many real-world systems [25, 37, 36, 55, 60], since they operate by updating the means and variances of the recognition engine's Hidden Markov Models (HMMs). These computations are nontrivial, and up to several million parameters of the model must be re-adapted each time the background noise changes. Further, sufficient memory to hold an entire new version of the HMM is required, which can be a burden on speech servers that may be operating tens or hundreds of channels at a time.

In this thesis a new technique is derived: the Uncertain Observation (UO) decoding algorithm, which extends feature transformation to include the positive benefits of model compensation. It allows fast adaptation when the source of the mismatch changes, and the improved performance of updating both mean and variance parameters.

The basis of the algorithm is to give not only the most likely estimate of the clean feature for each frame (as in feature transformation), but also to give the variance of this estimate. It will then be shown how the standard HMM decoding algorithm can be extended in order to use these observation variances. With this uncertainty passed into the HMM decoder, model means and variances are updated as in model compensation, albeit indirectly. However, since the uncertainty is now modeled with the features instead of the model parameters, it is easy to have the uncertainty change with time. In effect, the one-time, global update of all parameters in model adaptation has been exchanged for a continual set of smaller updates for every frame, spreading the adaptation computation cost into a continual smaller cost across time. This allows model compensation accuracy during time varying distortions.

It is worth noting that the UO technique can be applied to any distortion. One must only have a feature transformation algorithm that is extended to include an estimate for the variances of the compensated features. Rather than expressing frames as single points in (typically) 39-dimensional space, frames are expressed as 39-dimensional PDFs, describing the likelihood for all points in space of being the unobserved clean feature vector. One obvious application is to handle background noise, but other scenarios are also examined in

this thesis. Packet lossy channels, feature vector compression, and adaptation to compressed speech are considered in turn, and all show some measure of improvement in recognition accuracy. Each of these techniques is useful for distributed speech recognition (DSR), which involves the capture of speech at one location, and recognition at a some remote network location.

The uncertain observation technique will also be extended to the training of HMM parameters. Collection of training data has always proved difficult and time consuming for speech scientists. If training data is collected in environments with background noise that is not necessarily matched to recognition conditions, then models will be suboptimal. By incorporating knowledge about which parts of the training data are better known than others, training of robust models for multiple noise conditions is possible. For this thesis, the standard Baum-Welch parameter reestimation algorithm will be extended to use uncertain observations, resulting in a technique that is significantly more successful at recognizing clean data, given that only noisy data was available for training.

In summary, in this dissertation we:

- develop a feature extraction method that models each speech frame by a PDF in the speech feature domain,
- derive a new HMM decoding algorithm that uses these PDF features to handle time-varying mismatches between training and testing environments,
- compare, in terms of recognition performance and computational costs, this new algorithm to feature transformation and model compensation in stationary noise,
- demonstrate the proposed algorithm in time-varying noisy environments that cannot be handled by current state-of-the-art model compensation algorithms,
- apply the new algorithm to also handle dropped frames, feature vector compression, and speech compression, and
- propose that speech models appropriate for clean environments can be trained using noisy data by applying uncertain observation techniques.

This thesis is organized as follows. Chapter 2 will present the baseline recognition system used throughout this research. Also, the concepts required as building blocks needed to understand UO techniques will be presented. Chapter 3 develops the theory of UO decoding, showing how standard HMM decoding can be extended to the case of speech frames

described by PDFs instead of single deterministic points in space. Several interpretations of the algorithm and experiments showing an upper bound on performance available with this technique are documented. Chapter 4 applies the decoding algorithm to additive noise environments, and Chapter 5 show applications to distributed recognition environments. In Chapter 6, Baum-Welch reestimation of HMM parameters is extended to use observation uncertainty, resulting in the novel BW-UO training algorithm for HMMs. Experimental results are shown demonstrating significantly improved recognition performance for models trained with noisy speech. Finally, a summary of the work, conclusions, and a set of future related research areas are presented in Chapter 7.

# CHAPTER II

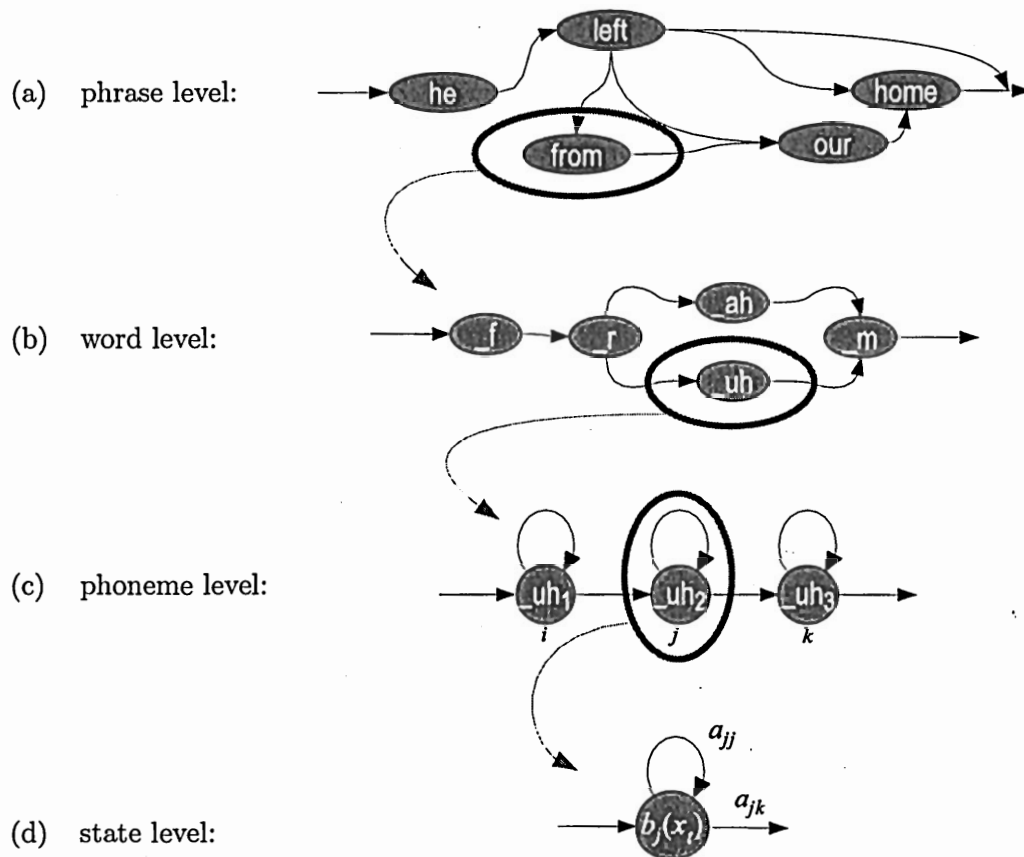
## BACKGROUND

This thesis will introduce a new technique for robust speech recognition that can adapt rapidly to time varying noise or other types of distortion. In order to better understand the new material, it is beneficial to first describe the baseline techniques that currently exist. This chapter will thus present a brief overview of speech recognition using Hidden Markov Models, the standard feature extraction techniques used for ASR, and the existing methods for robust recognition. This is by no means a comprehensive survey of existing techniques, and the interested reader will be referred to other locations for more in-depth descriptions and analysis where appropriate. Also covered in this chapter will be the shortcomings of current techniques, and the motivation for a new method. Finally, the baseline recognition experiments used throughout the thesis will be discussed.

### *2.1 Hidden Markov Model LVCSR*

Large Vocabulary Continuous Speech Recognition (LVCSR) [9, 34, 49, 51] requires several layers of processing to convert a speech waveform to text. At the top most level, a language model controls how words can be combined into phrases and sentences. As shown in Figure 1(a), this can be a finite state grammar that explicitly enumerates the possible paths and their probabilities. Such a language model is useful for highly constrained tasks, for example the input of credit card numbers. More general language models for dictation and transcription are  $N$ -grams and their many variants, which allow any word to follow any other words, with a probability determined by the previous sequence of  $N - 1$  words. For efficient implementation, low-likelihood paths can be screened out in advance, reducing complexity.

Below the phrase-level language model is a word-level model, as shown in the example of Figure 1(b) that describes how words are built from subword units, typically context



**Figure 1:** Topographical model of a the typical layers involved in HMM-based speech recognition.

dependent phonemes. Another common version for simpler tasks is the whole-word model, where each individual word is simply represented by its own unique HMM. Context dependent models for words can also be built using cross-word triphones, in which the exact set of subword units is chosen based on the pronunciation of the words that come before and after during decoding. This means that during decoding, to allow for  $J_1$  possible pronunciation paths into a word, and  $J_2$  possible pronunciation paths exiting a word, a total of  $J_1 \times J_2$  different subword pronunciations would be used for the current word.

Each subword unit is modeled by a single HMM. The most common form in the literature is a three-state, transition constrained left-to-right model that represents context dependent phonemes, such as triphones. A context independent example of this is shown in Figure 1(c). At each time index  $t$ , the system must transition from one state to another (possibly itself) following the paths allowed by the subword HMM units, the word models, and the language model. Each visit to a state will generate a single output vector, and it is this which is actually observed, rather than the sequence of state visits, which is hidden.

Within each subword unit HMM, each state transition has an associated transition probability,  $a_{ij}$ , that given the system is in state  $i$  at time  $t$ , it will transition to state  $j$  at time  $t+1$ , as shown in Figure 1(d). The complete set of  $a_{ij}$ 's for a given HMM is represented by the matrix  $\mathbf{A}$ . Transitions between subword units similarly follow an allowed set of paths, with likelihoods given by whatever language model is being used. Each state  $j$  also has an associated likelihood distribution  $b_j(\mathbf{x})$  that can be used to calculate the probability that the observation vector for frame  $t$ ,  $\mathbf{x}_t$ , was generated by this state. The complete set of parameters for these state output probability distributions for a subword unit's HMM is represented by the parameter vector  $\mathbf{B}$ . While many possibilities exist for the form of  $b_j(\mathbf{x})$  the most common in the literature is an  $n$ -dimensional,  $K$  mixture Gaussian mixture PDF, where  $n$  is the number of dimensions in the feature vector. Given a feature vector  $\mathbf{x}_t$ , the probability that this observation was generated by state  $j$  which has  $K$  mixtures each with

weight  $c_k$  is:

$$b_j(\mathbf{x}_t) = \Pr[\mathbf{x}_t | q_t = j] \quad (1)$$

$$= \sum_{k=1}^K c_k \mathcal{N}(\boldsymbol{\mu}_{jk}, \boldsymbol{\Sigma}_{jk}) \Big|_{\mathbf{x}_t} \quad (2)$$

$$= \sum_{k=1}^K \frac{c_k}{(2\pi)^{n/2} |\boldsymbol{\Sigma}_{jk}|^{1/2}} \exp \left\{ -\frac{1}{2} (\mathbf{x}_t - \boldsymbol{\mu}_{jk})^T \boldsymbol{\Sigma}_{jk}^{-1} (\mathbf{x}_t - \boldsymbol{\mu}_{jk}) \right\} \quad (3)$$

where  $\boldsymbol{\mu}_{jk}$  and  $\boldsymbol{\Sigma}_{jk}$  are the mean vector and covariance matrix of the  $k^{\text{th}}$  mixture of state  $j$ , respectively,  $c_k$  is the probability of mixture  $k$ , and  $T$  in this context is the linear algebra transpose. The final component of an HMM is the initial probability  $\pi_j$  for each state  $j$  of the model. For the standard left-to-right HMM used to describe the time evolution of speech, this corresponds to  $\pi = 1$  for the initial state of the model, and  $\pi = 0$  for all other states. The set of all parameters required to describe the HMM of the single subword unit  $w$  will be referred to as the parameter vector  $\lambda_w = (A_w, B_w, \pi_w)$ , and the set of all HMMs for all subword units will be referred to as  $\Lambda = (\lambda_0, \dots, \lambda_{W-1})$ , where  $W$  is the total number of subword units.

While the multiple levels above can be considered conceptually distinct, in implementation they can collapse into a single lattice structure, as shown in Figure 2. The horizontal axis indexes the feature vectors extracted from the speech signal, and the vertical axis indexes the complete set of all states in all subword unit HMMs. In order to perform speech recognition, the goal is to find the most likely path through the lattice, given the subword unit model parameters  $\Lambda$ , the models for generating words from these units, the language model for generating phrases and sentences from these words, and observation vectors  $\mathcal{O}_X = (\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_T)$ . Efficient methods exist for calculating this for the case of a single, large HMM. Because of language modeling, however, a new wrinkle exists that there are now multiple paths at the same time  $t$  through the same state  $j$  that correspond to different word sequences, and thus transition probabilities that are dependent upon the history of the preceding word sequence. The typical method for handling this situation is instead of keeping a compact score for each state in the lattice that gets updated for each new speech frame, a list of “traces,” each with their own history information (including

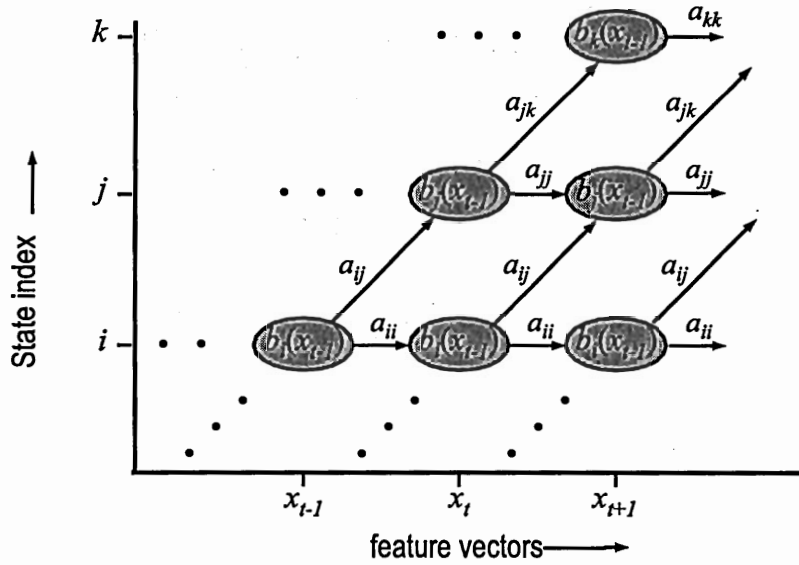


Figure 2: HMM lattice decoding used for trace projection speech recognition.

state and language model location) exist and are propagated forward for each new observation vector. By pruning this list of traces to keep only those that could reasonably be the correct word sequence, processing can be performed in an efficient manner.

## 2.2 Feature Extraction

The LVCSR development in Section 2.1 specifically does not mention the form of the feature vectors to be extracted from the speech waveform. The HMM is independent of the feature set used, and the choice depends upon that which can best represent the information in the speech signal, and be used in a compact fashion in computing the state output probabilities.

The basic feature vector extracted in most research speech recognition systems is the Mel Filterbank Cepstral Coefficient (MFCC) parameter set [48]. These have become the de facto standard over LPC, filterbank, and other feature sets, because they are

- a compact representation of the smoothed speech spectrum,
- are well modeled by Gaussians with diagonal covariance matrices, and
- experimentally have been found to provide the best recognition performance.

MFCCs are calculated by passing the time domain speech signal  $x[n]$  through a set of 20 to 30 bandpass filters with center frequencies spaced along the mel scale, such that center frequencies are spaced further apart as the frequency increases, as shown in the MFCC extraction process flowchart in Figure 3. This set of filterbanks is often implemented by

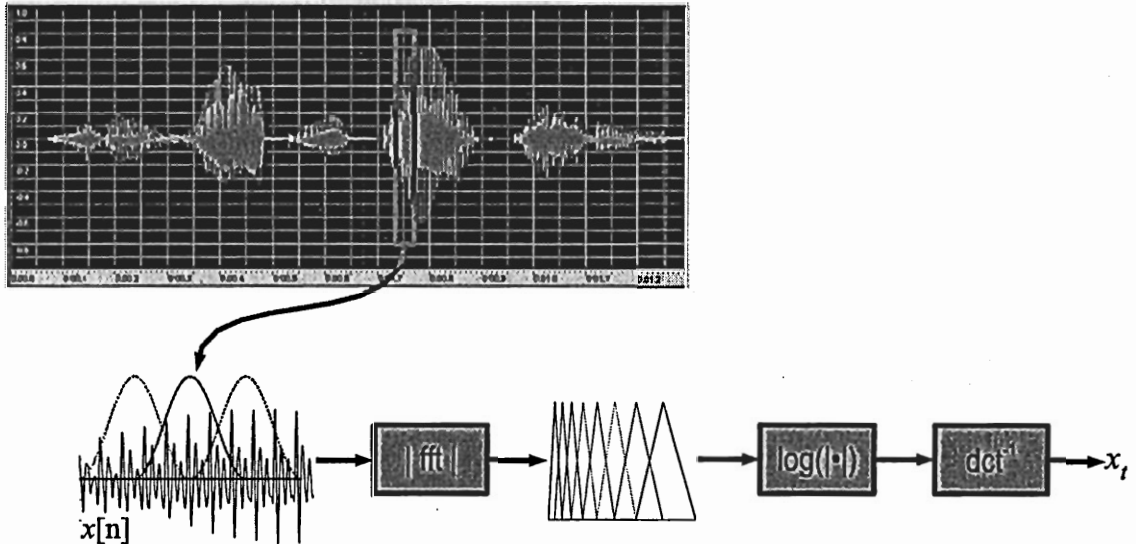


Figure 3: Flowchart for extracting an MFCC vector from one frame of speech

windowing the FFT magnitude. Roughly every 10 milliseconds (the frame rate), a snapshot is taken of the energy in each of these  $m$  filterbanks over an interval of 20 to 25 milliseconds (the frame size). This  $m$ -dimensional observation vector is the linear filterbank energy domain representation.

The logarithm is taken of each filterbank energy, such that by simple mean removal, any constant frequency response terms are removed. This corresponds to removing frequency shaping due to the transducer.

The final operation is to convert this representation to the cepstral domain by taking the inverse DCT. This serves to decorrelate the features [51], so that the covariance matrices used in Gaussian mixture models of feature clusters are closer to diagonal, greatly reducing both the amount of data required to train, and the computations required to decode. This final vector is truncated to  $n$  components, typically 12, since the higher indices are related

to the pitch, which is not important for recognition (at least for the English language). It is convenient to view this operation as multiplying the log filterbank energy vector by a  $m$ -by- $n$  component cosine matrix,  $C$ . It is the resulting  $n$ -dimensional output that is referred to as the MFCC feature vector. Traditionally the zeroth order MFCC coefficient is replaced by a log energy term, although in practice this has little effect on the recognition rate. Retaining cep[0] is useful if reverse conversion from the cepstral domain back to the filterbank log energy domain is needed, which is the case for several robust recognition algorithms in the literature.

Typically feature vectors are extended to include their first and second derivatives. The sequence of feature vectors extracted from a speech signal will be represented by  $\mathcal{O} = (\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_t, \dots, \mathbf{x}_T)$  where  $t$  indexes the frame of speech, and  $\mathbf{x}_t$  is the MFCC vector associated with this frame.

### ***2.3 Robust Speech Recognition***

There are many reasons for mismatch between speech observed for recognition and the conditions for which recognition models are trained [1, 22, 41]. Dialect, emotional state, Lombard effect, room response, background noise, transducer response, and the transmission channel are some of the sources of mismatch that affect the observed signal, as shown in Figure 4. In this system, speech that matches correctly with the models is the unobservable time domain sequence  $x[n]$ , which matches exactly the conditions present in the training data. If speech were collected by a close-talking microphone at the mouth, the signal received,  $z[n]$ , is still distorted from the matched signal, since it has a potentially different dialect, etc.

As a brief example of the detrimental effect on recognition that even seemingly minor training/testing mismatches can create, consider the following experiment that was run during the course of the research presented in this paper. The task was to test gender dependent speech models in a phoneme recognition system using the well-known TIMIT database. As expected, the results in Table 1 show that a gender dependent system outperformed a gender independent system. An second interesting result, however, was when

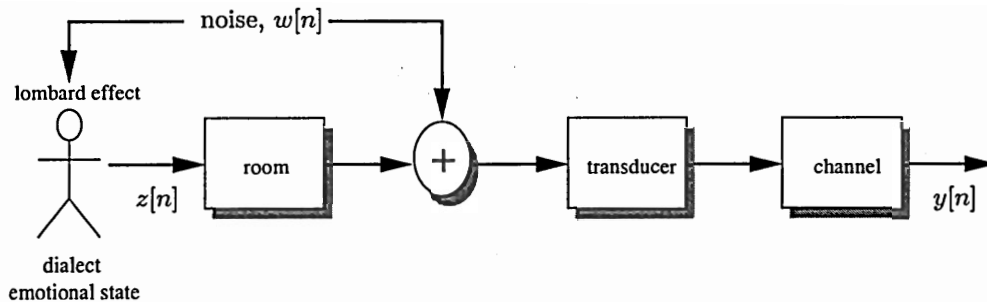


Figure 4: Block diagram of speech production mismatches

the gender-detection algorithm was accidentally swapped during decoding. This meant the system used male-trained models to recognize female speech, and vice-versa.

Table 1: Gender-dependent phoneme recognition experiments.

| Recognizer Type           | Phoneme Error Rate |
|---------------------------|--------------------|
| gender independent        | 52.9%              |
| gender dependent          | 51.6%              |
| gender dependent, swapped | 70.2%              |

The surprising result was that while gender-specific models slightly improved recognition performance, using models specifically mismatched for gender gave a much, much larger decrease in performance. This simple example shows the importance of robust recognition, because what would seem like a small mismatch between training and testing resulted in a recognizer that performed much poorer than the matched version.

Robust speech recognition thus has the goal of minimizing the errors due to the mismatch between training and testing environments. There are three standard approaches to this problem:

- robust feature sets,
- feature transformation, and
- model compensation.

It is assumed for the remainder of this document that the feature set being used is already appropriate for mismatched conditions. The remainder of this section will thus review feature transformation and model compensation techniques, as it is by analyzing the strengths and weaknesses of these methods that the need for a new algorithm will be seen.

Both of these methods require some knowledge of the specific mismatch conditions that they are facing. Thus prior to discussing these robust techniques, we will discuss methods for estimating a distortion model.

### 2.3.1 Estimating A Distortion Model

An important step in performing feature transformation or model compensation is first to obtain an accurate estimate of the background noise. The assumption is that while speech is pseudo-stationary for only 10 or 20 millisecond segments, the noise is stationary for a longer time – perhaps on a scale of seconds. Over the course of several seconds, it is almost certain that there will be pauses in the speech signal, for example between sentences. If these non-speech times can be identified, they can be used to calculate parameters for a noise model,  $\mathcal{W}$ . In environments with only small amounts of noise, the non-speech frames can be identified with a technique as simple as looking at signal energy. In environments with higher noise, a more sophisticated approach is required, using a voice activity detector (VAD) which has the ability to discriminate speech from non-speech intervals. Recent extensions allow for noise model updates even during continuous speech.

The model  $\mathcal{W}$  most frequently used for model compensation is a single Gaussian mixture in the MFCC domain. Feature transformations are often applied in the FFT or filterbank energy domain, and often are first-order statistics, such as the expected value of the absolute value of each FFT bin.

### 2.3.2 Feature Transformation

The goal of feature transformation is to observe a noisy sequence, estimate the background noise during this sequence, and generate a new sequence that more closely resembles the unobserved ideal clean sequence. Thus, given  $\mathcal{O} = (\mathbf{y}_1, \mathbf{y}_2, \dots, \mathbf{y}_T)$  and  $\mathcal{W}_t$ , generate  $\hat{\mathcal{O}} =$

$(\hat{\mathbf{x}}_1, \hat{\mathbf{x}}_2, \dots, \hat{\mathbf{x}}_T)$ , where

$$\hat{\mathbf{x}}_t = E[\mathbf{x} | \mathbf{y}_t, \mathcal{W}_t]. \quad (4)$$

A graphical example for a distorted, one dimensional observation vector  $y_t$  is shown in Figure 5. Using some knowledge of the environment conditions  $\mathcal{W}_t$ ,  $\hat{\mathbf{x}}_t$  is substituted during the decoding process. The transformation process, shown in Figure 6, can never be completely accurate, as any distortion of serious interest has a many-to-one reverse mapping. But if the estimate can at least be closer to the clean observation  $x_t$ , in terms of state output probabilities calculated for all the states of the model set, the acoustic matching scores will be closer to those of the unobserved clean data.

This transformation can be implemented in a wide variety of ways. The simplest method is to use a background noise estimator from the previous section and perform spectral subtraction [7]. A more thorough method is to use a technique such as the Ephraim-Malah optimal spectral estimation algorithm [16], or the Hansen-Clements constrained iterative Wiener filter [30].

An advantage of feature transformation algorithms is that since they typically operate on a frame-by-frame basis, it is entirely feasible to have different noise models  $\mathcal{W}_t$  specified for every frame  $t$ .

The major drawback with feature transformation techniques is that effectively only the model means are updated, and the variances are left unchanged. Thus, while recognition performance is improved, it is not taking full advantage of the available tools.

### 2.3.3 Model Compensation

Though there are several different algorithms to perform model compensation, they share a basic framework. First, a background noise model  $\mathcal{W}$  is generated during nonspeech times. This background model,  $\mathcal{W}$ , is then used to compensate the speech models to approximate what they would be if they had been trained with data from this background environment.

To perform the compensation step, each individual HMM state in the model set is considered independently. The goal is to update the state output probability distributions:

$$b'_j(\mathbf{x}_t) = f(b_j(\mathbf{x}_t) | \mathcal{W}) \quad (5)$$

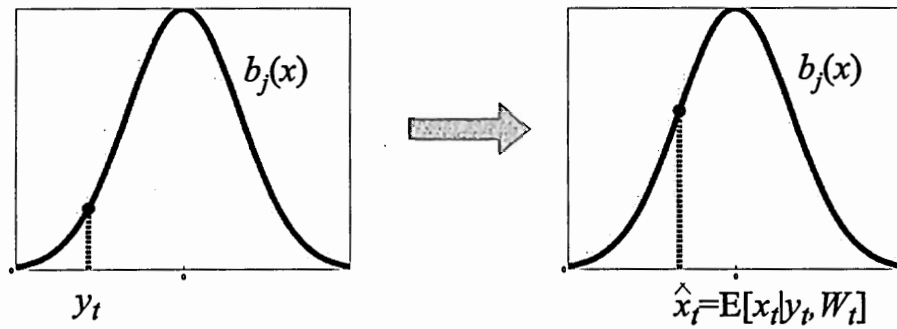


Figure 5: Example of feature transformation.

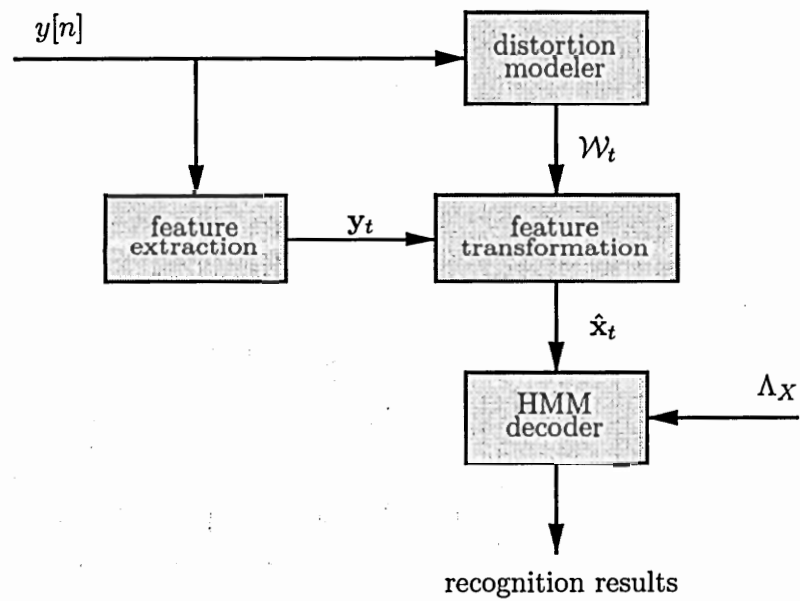


Figure 6: Block diagram of a feature transformation operation.

where  $b_j(\mathbf{x}_t)$  is the PDF giving the likelihood of state  $j$  generating observation vector  $x_t$ ,  $\mathcal{W}$  is the background noise model, and  $b'_j(\mathbf{x}_t)$  is a new PDF describing state  $j$ 's output given  $\mathcal{W}$ .

A graphical example for a distorted, one dimensional observation vector  $y_t$  is shown in Figure 7. Using knowledge of the environment conditions  $\mathcal{W}_t$ ,  $\hat{\lambda}_y$  is estimated prior to recognition begins. During decoding, this model is substituted in place of  $\lambda_x$ . The model compensation process, diagrammed in Figure 8, can be very accurate, nearly able to reach matched conditions of training on noisy data (which, obviously, will have a higher error rate than matched *clean* conditions). Thus by changing model parameters, improved recognition during mismatched conditions can be achieved.

The most popular model compensation method is the PMC algorithm [24]. Every state output probability PDF,  $b_j(\mathbf{x})$ , is transformed from a normal random variable (r.v.) in the cepstral domain to a normal r.v. in the filterbank log energy domain. This is then transformed into a lognormal r.v. in the linear energy domain. The same is done for a Gaussian that represents the noise estimate  $\mathcal{W}$ . This results in a lognormal mean and variance for the noise and for each Gaussian mixture in the model. Since speech and noise are approximately additive in this domain, a new lognormal for each model mixture can be generated by:

$$\hat{\mathbf{m}}_{jk} = \mathbf{m}_{jk} + \mathbf{m}_{noise} \quad (6)$$

$$\hat{\mathbf{S}}_{jk} = \mathbf{S}_{jk} + \mathbf{S}_{noise} \quad (7)$$

for each mixture  $k$  of each state  $j$  in the model set, where  $\mathbf{m}$  is a lognormal mean and  $\mathbf{S}$  is a lognormal variance. This new lognormal representation in the linear energy domain can then be transformed back into a normal representation in the log energy domain, and finally, transformed back into a new Gaussian PDF in the cepstral domain. Other model compensation techniques, such as Vector Taylor Series adaptation, are described in [2, 23, 31, 35].

After every Gaussian in all HMM models of the speech model set have been updated using this method, speech decoding (ASR) with the new HMM model set is carried out

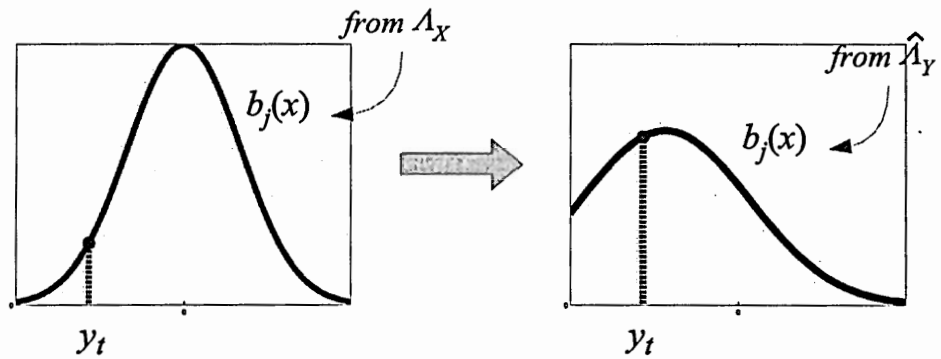


Figure 7: Example of model compensation.

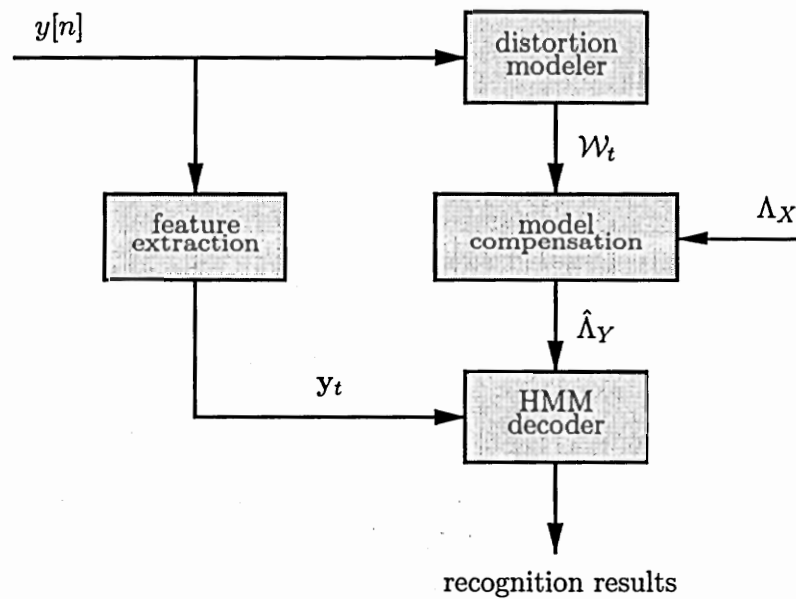


Figure 8: Block diagram of the model compensation procedure.

in the standard manner. There is no further computational overhead, but this entire new model must be stored for each input audio stream. The major drawback is that when the background noise conditions change, every Gaussian must be updated again. For a large vocabulary system, it is not unreasonable to have a total of 15,000 states, each modeled by 8 mixtures. With 39 dimensional feature vectors, a model update would require updating almost ten million mean and variance parameters.

## *2.4 Baseline Recognition System*

Section 2.1 described a lattice-based trace projection solution for Hidden Markov Model LVCSR. Trace scores in this lattice are a combination of acoustic match likelihoods, and language model match likelihoods. Given the assumption that the words, their pronunciation, and the sentence structure used by a talker are independent of the mismatch between training and testing environments, the problem of robust recognition can be decoupled from the language model used. While better performance can obviously be had by improving the language model (or even by weighting it differently), it is a reasonable expectation that improvements in the acoustic matching will be directly related to improvements in overall recognition, even for large vocabulary scenarios. For this reason, the work in this thesis will focus entirely on robustness at the acoustic matching level. Experiments use only the simplest of grammars, and test either:

- whole word digit recognition, or
- phoneme recognition.

Experiments such as these are smaller, faster, and were found easier to analyze than running large dictation experiments. Further, the direct effects of the robust recognition algorithms are easier to detect, as the final results are not “muddied” with side effects from other parts of the recognition system.

The baseline recognition system used for the work in this thesis is based upon the prototype speech recognition system freely available from the Institute for Signal and Image

Processing at Mississippi State University. The system uses a Baum-Welch training algorithm, and a trace-projection decoder. The source code for the ISIP recognizer is available for download, allowing the modifications necessary to implement the algorithms developed as a part of this research.

Whole word digit recognition was based upon the standard Aurora2 implementation. Aurora2 was designed for the HTK recognition system, a research package available from Cambridge University. However, the model parameters such as states per word and Gaussian mixtures per state were easily replicated using the ISIP system. The Aurora2 standard is 16-state HMMs for each of the 11 words of the TIDIGIT database, with three mixtures per state.

Phoneme recognition experiments used the standard TIMIT database, with phonemes converted to the 41 element CMU phoneme set, shown in Table A.1. This set was used instead of the standard TIMIT 52-phoneme set because a 130K+ dictionary using the CMU set was also available. While no experiments reported directly in this thesis used large vocabulary dictation, and thus needed such a dictionary, side experiments to satisfy the curiosity of the author were performed that required one. In order to allow sharing of models, training scripts, and transcripts between many experiments, then the TIMIT transcripts were converted to the CMU phoneme set, and used throughout this thesis. Phonemes were modeled by three state, left-to-right HMMs, with 32 mixtures per state.

In the ISIP recognizer, context-dependent triphones are possible, although the phoneme recognition experiments in this thesis used only context independent phoneme models. In particular, this was because cross-word triphone decoding is extremely computationally expensive, especially for phoneme recognition, where every word is only one phoneme long. The ability to better model speech comes at a steep price: the rapid expansion of traces that must be retained for accurate decoding slows recognition. It was found experimentally that by increasing the mixture count in phonemes to give HMM sets with a similar number of parameters as found in a triphone model set (i.e., 40 3-state HMMs with 128 mixtures apiece, instead of 1000 triphone HMMs with six mixtures apiece), running in context independent mode gave recognition accuracy within a few percent of the triphone system, in roughly one

thirtieth the time. To further speed development, the number of mixtures was cut to 32, trading again errors for speed. A summary of baseline results is given in Table 2.

**Table 2:** Baseline error rates for phoneme recognition experiments.

| Phoneme Type | Number of HMMs | Mixtures Per HMM | Phoneme Rate | Hours Required For Decoding |
|--------------|----------------|------------------|--------------|-----------------------------|
| triphone     | 800            | 8                | 34.2%        | 98                          |
| monophone    | 41             | 128              | 36.3%        | 3.9                         |
| monophone    | 41             | 32               | 38.7%        | 1.8                         |

The features extracted for this thesis are MFCC vectors, as described in Section 2.2. A preemphasis coefficient of 0.97 is always used, as is a set of 24 mel-spaced filterbanks that span 0-4kHz. Unless otherwise specified in a particular section of this thesis, the extraction method converts the 24 filterbanks to thirteen MFCC coefficients. The zeroth cep coefficient,  $\text{cep}[0]$ , is then replaced by the log energy of the frame. First and second derivatives are taken across multiple frames, giving a final feature vector of length 39.

## CHAPTER III

# THE UNCERTAIN OBSERVATION DECODING ALGORITHM

The background section presented an overview of current approaches to feature enhancements and model compensation for robust speech recognition. These two methods were shown to have opposite strengths and weaknesses in terms of their ability to handle time-varying noise, and their overall ability to improve recognition accuracy. This chapter will present the first major contribution of this thesis: the *Uncertain Observation* (UO) decoding algorithm [4, 5]. This method will combine the strengths of both feature enhancement and model compensation, resulting in a new, superior algorithm.

Uncertain Observation decoding is an extension of the standard HMM in which state output probability calculations are augmented to take PDF representations of each input frame, instead of single fixed points. By incorporating feature uncertainty, the method allows performance superior to that available by feature compensation alone. Further, since feature uncertainty is included with each feature frame, time-varying distortions can be handled easily.

This chapter will first review the shortcomings of current robust recognition techniques, after which the desired characteristics for a new algorithm to address these problems is discussed. The new technique will be motivated by increasingly more complex examples, resulting in the new general algorithm that uses Uncertain Observations. This is followed by a detailed description, and several interpretations of the derived equations. Special cases will then be covered, in particular the case where a single-mixture Gaussian PDF is used to model the likelihood of unseen clean features. This formulation allows simple and efficient integration into existing speech recognition systems. An analysis of this method is covered in detail, showing the accuracy improvements that it can provide while addressing the shortcomings of current robust recognition methods. Finally, similar techniques recently

described in the literature are compared to the UO method of this thesis.

### *3.1 Problem Description*

The goal of this thesis is to develop a new algorithm for robust speech recognition. Thus a quick review of the shortcomings to current techniques is appropriate. As discussed in the previous chapter, robust techniques fall into three major categories: robust feature extraction, feature transformation, and model compensation. The first of these, although important, is independent of the actual HMM decoder, and for this thesis, it will be assumed that a good feature set is already in use. The second and third categories directly affect the recognition engine, and it is at this level where the work for this thesis will focus.

In Section 2.3, feature transformation and model compensation were described in detail. Here, we will summarize the pros and cons found for these two techniques:

#### *Feature Transformation*

- ✓ Allows time varying updates of the distortion model,  $\mathcal{W}_t$ .
- ✗ Effectively only updates model means.
- ✗ Only mediocre performance improvement achievable.

#### *Model Compensation*

- ✓ Updates means and variances of  $\Lambda_Y$ , giving better performance.
- ✗ Must re-update all parameters of  $\Lambda_Y$  if  $\mathcal{W}$  changes.
- ✗ Requires storage for the new model  $\Lambda_Y$ .

Thus, the desired result of this research is a method that combines the positive characteristics of these two methods. The new recognition algorithm should:

- ⇒ allow for updates to both model mean and variance parameters,
- ⇒ these updates should be able to change rapidly, and
- ⇒ implementation should be efficient.

The remainder of this chapter will present and analyze a new algorithm that uses a model of feature vector uncertainty to meet these criteria.

### 3.2 *Motivating Example*

In order to develop the Uncertain Observation method for robust speech recognition, a motivating example will be used to demonstrate progressively more complex models of distortions. It will also show in a straightforward manner how these models can be handled in a frame-by-frame manner at the level of the state output probability calculation, as an extension of the feature transformation method of Section 2.3.2. The end result is a generic formulation of the UO algorithm for speech recognition decoding.

To begin this motivating example, recall from Chapter 2 that in a standard speech recognition system, a 20-30 millisecond long frame of speech is extracted approximately every 10 milliseconds, with a single observation vector  $\mathbf{x}_t$  being extracted from each of these frames, generating the observation sequence

$$\mathcal{O}_x = (\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_T) \quad (8)$$

for frames  $t \in [1, T]$ . The language model and HMM structure are used to prune the list of states in the model set to find those states that are sufficiently likely for frame  $t$ . For each of these states  $j$ , the likelihood that observation frame  $t$  could have been generated is calculated using the state output probability equation:

$$\Pr[\mathbf{x}_t | q_t = j] = b_j(\mathbf{x}_t) \quad (9)$$

where  $q_t = j$  denotes that the HMM is in state  $j$  during observation frame  $t$ . This simply corresponds to evaluating the PDF that describes the output probabilities of state  $j$  at the single point  $\mathbf{x}_t$ .

During a distortion, however, the observation speech sequence will instead be

$$\mathcal{O}_y = (\mathbf{y}_1, \mathbf{y}_2, \dots, \mathbf{y}_T),$$

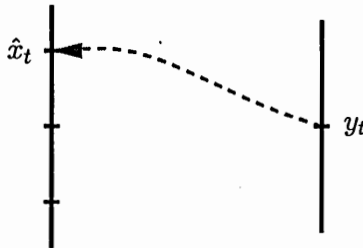
with each feature vector  $\mathbf{y}_t$  being some transformed version of the unobserved clean signal  $\mathbf{x}_t$  and the current distortion:

$$\mathbf{y}_t = g(\mathbf{x}_t, \mathcal{W}_t)$$

If information is known about how this transformation affects feature vectors in general, then this can be used to generate expected values for the clean speech vectors. This is standard feature transformation as described in Section 2.3.2. A one dimensional example of this is shown in Figure 9, where the observed feature  $y_t$  on the numberline on the right is mapped one-to-one to  $\hat{x}_t$ , on the numberline on the left. It is then this feature vector that is used in the state output probability calculation:

$$\Pr[y_t|q_t=j] = \Pr[\hat{x}_t|q_t=j] \quad (10)$$

$$= b_j(\hat{x}_t). \quad (11)$$



**Figure 9:** Feature Transformation of an observation distortion, showing a mapping between the observed, distorted point  $y_t$  (on the numberline to the right) transformed to  $\hat{x}_t$ , the expected location of the unobserved, clean observation point  $x_t$  (on the numberline on the left).

This is shown graphically in Figure 10. In this figure, the probability of the observation  $y_t$  is calculated for state  $j$  using the output likelihood PDF  $b_j(x)$  associated with state  $j$  evaluated at  $\hat{x}_t$ .

Consider now the case where the exact form of the transformation  $g(\cdot)$  is not completely known. In particular, suppose an observation frame can be represented by two equally likely “sub-observations.” A representation of this situation is shown in Figure 11. The single distorted observation  $y_t$  can be mapped to either location  $\hat{x}_{t,1}$  or  $\hat{x}_{t,2}$ , with equal probability. In this case, a reasonable method for calculating the state output probability for this frame would be to average the output probabilities of the individual sub-observations, as follows:

$$\Pr[y_t|q_t=j] = \frac{1}{2}b_j(\hat{x}_{t,1}) + \frac{1}{2}b_j(\hat{x}_{t,2}). \quad (12)$$

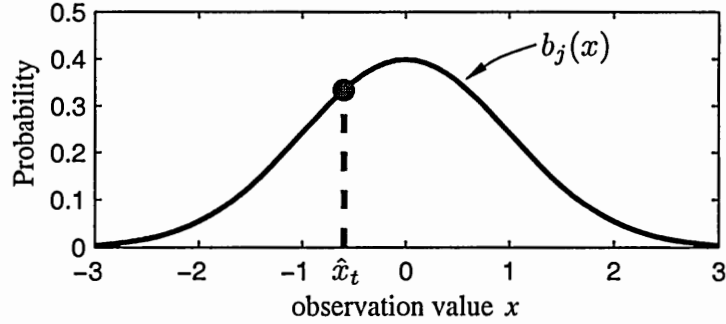


Figure 10: Evaluation of the state output probability of the distorted observation  $y_t$  shown in Figure 9, using  $\hat{x}_t$  inside  $b_j(x)$ .

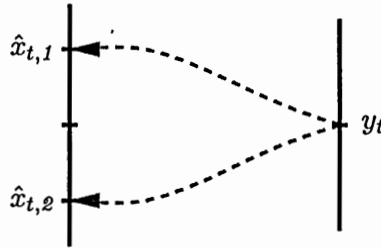
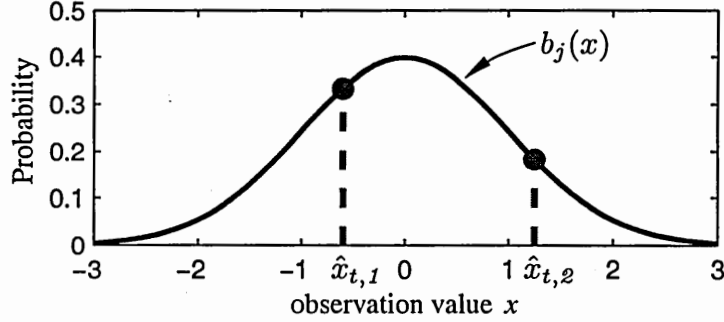


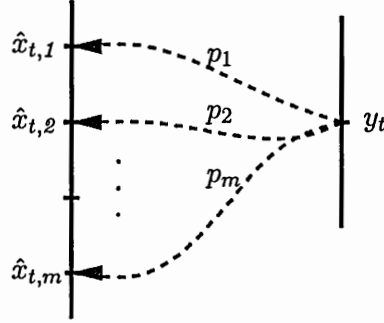
Figure 11: Example of feature transformation when the distortion maps to two equiprobable points in the clean feature domain.

An example of calculating this averaged output probability of Equation 12 is shown graphically in Figure 12. While this does not give the ideal results available in the previous one-to-one mapping case, it will give better expected performance overall than selecting either only one point to evaluate (giving either  $b_j(\hat{x}_{t,1})$  or  $b_j(\hat{x}_{t,2})$ ) or evaluating the state output probability of an average location,  $b_j(0.5(\hat{x}_{t,1} + \hat{x}_{t,2}))$ .

This method of averaging discrete state output probabilities can easily be extended in two ways. First, the number of sub-observations can be increased to some arbitrary value,  $M$ . Thus  $y_t$  will be represented as the set of sub-observations  $\hat{x}_{t,1}$  through  $\hat{x}_{t,M}$ . Second, the likelihood of each  $m^{\text{th}}$  sub-observation can be assigned an unequal probability  $p_m$ , with the constraint that  $\sum_1^M p_m = 1$ . This scenario is represented in Figure 13, with  $m$  sub-observations for  $y_t$ .



**Figure 12:** The output probability of distorted observation  $y_t$ , given equiprobable sub-observations  $x_{t,1}$  and  $x_{t,2}$ , is  $\Pr[y_t|q_t=j] = 0.5b_j(\hat{x}_{t,1}) + 0.5b_j(\hat{x}_{t,2})$ .



**Figure 13:** Example of decoding with  $m$  equiprobable points.

The output probability for  $y_t$  in this case is given by the following equation:

$$\Pr[y_t|q_t=j] = \sum_{m=1}^M p_m \cdot b_j(x_{t,m}), \quad \text{where } \sum_{m=1}^M p_m = 1. \quad (13)$$

This is the general formulation of the Uncertain Observation decoding algorithm for discrete feature transformation.

### 3.3 General Formulation

In reality, all observations ultimately are stochastic. In well controlled scenarios, the variance of the observations may be low, but especially as noise, compression, and other distortions are introduced, observation uncertainty increases. This leads to the representation of each observation vector not by a single deterministic point in space, but instead by a PDF describing the likelihood of any given point in feature space, given the distorted observation

and whatever is known about the existing distortion conditions, such as background noise.

Thus the discrete case of the Uncertain Observation algorithm described in the previous section can be expanded. Instead of a discrete set of possible observations, consider the case of a PDF that describes the likelihood of all possible observations. This might arise in a scenario where, due to background noise or other effect, the true observation  $\mathbf{x}_t$  cannot be observed, but a likelihood distribution  $f_t(\mathbf{x})$  can be found which describes the ideal observation  $\mathbf{x}_t$  by using the noisy observation  $\mathbf{y}_t$  and some model of the background noise  $\mathcal{W}_t$ . Thus, for each noisy observation  $\mathbf{y}_t$ , a PDF is generated to describe the original clean frame  $\mathbf{x}_t$ , giving the new observation sequence:

$$\mathcal{O}' = (f_1(\mathbf{x}), f_2(\mathbf{x}), \dots, f_T(\mathbf{x})), \quad (14)$$

where  $f_t(\mathbf{x}) = \Pr[\mathbf{x}|\mathbf{y}_t, \mathcal{W}_t]$  is a PDF whose parameters are some function of  $\mathbf{y}_t$  and  $\mathcal{W}_t$ . Various methods for finding this PDF will be discussed in Chapters 4, and 5. The probability of this observation, given the model is in state  $j$ , can now be written as an integration of all possible observations, weighted by their respective probabilities:

$$\Pr[\mathbf{y}_t|q_t=j, \mathcal{W}_t] = \int_{-\infty}^{\infty} f_t(\vartheta) \cdot b_j(\vartheta) d\vartheta. \quad (15)$$

This is the general formulation of Uncertain Observation decoding for continuous feature PDF extraction.

As an example, consider Figure 14. The original, undistorted vector  $x_t$  could not be observed due to background noise, and instead  $y_t$  was observed. What *is* known, however, is that the likelihood of  $x$  given  $y_t$  can be described by the uniform distribution  $f_t(x)$ , shown in Figure 14(a). Using Equation 15, the probability of of this observation being generated by state  $j$  can be calculated. Graphically, this is the integral of the curve illustrated in Figure 14(b).

A block diagram showing a typical implementation of this algorithm is given in Figure 15. This can be compared to the algorithm flowcharts for feature transformation and model compensation, given in Figures 6 and 8, respectively. Unlike feature transformation, variance information is being updated, in the form of the uncertainty included in the feature PDF  $f_t(\mathbf{x})$ . And, unlike model compensation, the baseline speech models  $\Lambda_X$  are used

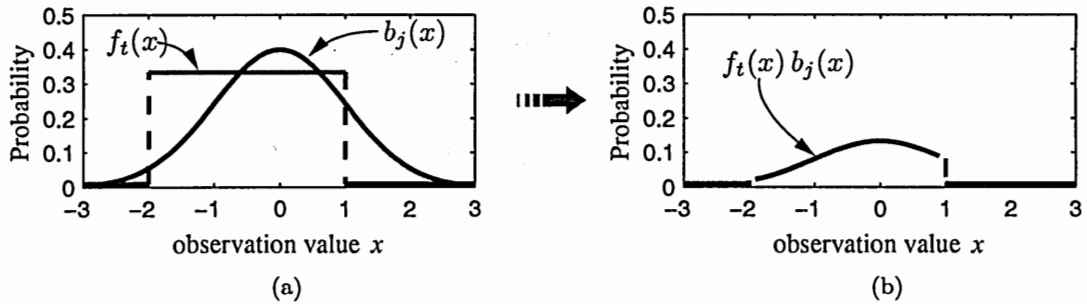


Figure 14: The output probability  $b_j(x)$  and observation probability  $f_t(x)$  shown in (a) are used to generate  $f_t(x)b_j(x)$  in (b), whose integral is the probability of frame  $y_t$  being generated by state  $j$ , given distortion model  $\mathcal{W}_t$ .

directly during decoding.

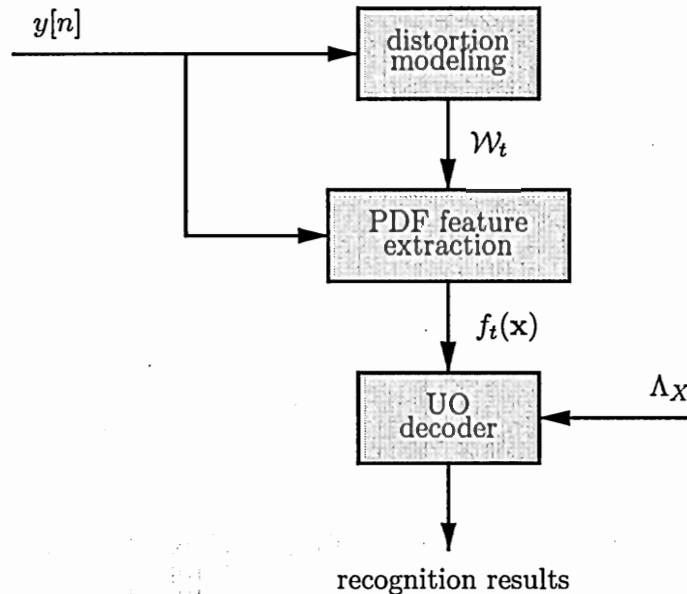


Figure 15: Block diagram of the Uncertain Observation decoding procedure.

### 3.4 Interpretations

Two interpretations of the Uncertain Observation decoding algorithm of Equation 15 are given in this section. First, it is shown that formulation is equivalent to an expected value of the state output probability conditioned on the distortion. Second is an interpretation as a Bayes decision theory problem, but with uncertain feature data.

### 3.4.1 Expected Value Interpretation

In the development of the UO algorithm in Section 3.3, the left side of the state output probability equations were shown to be  $\Pr[y_t|q_t=j, \mathcal{W}_t]$ . This section will present another representation that is important because of its intuitiveness: the expected value interpretation. The basis is to note that the right side of the Uncertain Observation decoding equations can be shown to be equations for conditional expected values.

The discrete feature transformation case (Equation 13), is easily related to the standard form of conditional expectation of discrete random variables, as shown below:

$$\Pr[y_t|q_t=j] = \sum_{m=1}^M p_m \cdot b_j(\mathbf{x}_{t,m}) \quad (16)$$

$$= \sum_{m=1}^M \Pr[\mathbf{x}_{t,m}|y_t, \mathcal{W}_t] \cdot b_j(\mathbf{x}_{t,m}) \quad (17)$$

$$= E[b_j(\mathbf{x}_{t,m})|y_t, \mathcal{W}_t]. \quad (18)$$

$$= E[\Pr[\mathbf{x}|q_t=j] |y_t, \mathcal{W}_t]. \quad (19)$$

A similar approach can be taken with the continuous PDF version of UO decoding given in Equation 15. Again, the right side of the equation can be written in the standard form of a conditional expected value (for example, [45], p. 169), this time for continuous random variables:

$$\Pr[y_t|q_t=j] = \int_{-\infty}^{\infty} f_t(\mathbf{x}) b_j(\mathbf{x}) d\mathbf{x} \quad (20)$$

$$= \int_{-\infty}^{\infty} \Pr[\mathbf{x}|y_t, \mathcal{W}_t] \Pr[\mathbf{x}|q_t=j] d\mathbf{x} \quad (21)$$

$$= E[\Pr[\mathbf{x}|q_t=j] |y_t, \mathcal{W}_t]. \quad (22)$$

For both the discrete and continuous cases, Uncertain Observation decoding can thus be considered equivalent to solving for the expected value of the standard state output probability of state  $j$ , given the current model for the distortion,  $\mathcal{W}_t$ , and the distorted observation vector,  $\mathbf{y}_t$ .

### 3.4.2 Bayes Decision Theory Interpretation

Another interpretation of the decoding algorithm given in Equation 15 is as a Bayes Predictive Classifier (BPC) [44]. This classification methodology, which has received little mention in standard pattern recognition texts, is similar to standard Bayes decision theory, except the parameters normally considered as being exactly known a priori are represented instead by distributions over another set of parameters. This section will present first the standard HMM decoding algorithm as a Bayes optimal classification problem, and then extend it to the BPC form when the observation data is uncertain. Credit for this interpretation goes to a soon to be published work [12] by researchers at Microsoft who worked in parallel (and independently) on Uncertain Observation decoding.

The basic form of a state-of-the-art speech recognizer uses training data to generate an acoustic model  $\Lambda_X$  and a language model  $\Gamma$  that completely describes human speech. The goal of recognition is then to find the optimal word sequence  $\hat{W}$  which, for a given observation sequence  $\mathcal{O}_Y$ , finds:

$$\hat{W} = \arg \max_W \Pr(W | \mathcal{O}_Y, \Lambda_X, \Gamma) \quad (23)$$

which, using Bayes' rule can be rewritten

$$\hat{W} = \arg \max_W \frac{\Pr(\mathcal{O}_Y | W, \Lambda_X, \Gamma) \Pr(W | \Lambda_X, \Gamma)}{\Pr(\mathcal{O}_Y | \Lambda_X, \Gamma)} \quad (24)$$

$$= \arg \max_W \Pr(\mathcal{O}_Y | W, \Lambda_X) \Pr(W | \Gamma) \quad (25)$$

This is known as the *maximum a posteriori* (MAP) decoding decision because it finds the sequence with maximum likelihood after the sequence  $\mathcal{O}_Y$  has been observed. It gives the optimal result, achieving the minimum expected word error rate assuming all parameters are completely known.

The result of Equation 25 is separated into two independent parts:  $\Pr(W | \Gamma)$ , the probability of the word sequence  $W$  given the language model  $\Gamma$ , and  $\Pr(\mathcal{O}_Y | W, \Lambda_X)$ , the probability of the observation sequence  $\mathcal{O}_Y$  given both the word sequence  $W$  and the acoustic model  $\Lambda_X$ . If the assumption is made that the language model  $\Gamma$  is independent of the environmental conditions, only the first term is going to be affected by the acoustic environment.

This first term is found by noting that for every state sequence  $\mathbf{q}$ ,  $\mathbf{q} = (q_1, q_2, \dots, q_T)$ , and using Markov state independence assumptions gives

$$\Pr(\mathcal{O}_Y, \mathbf{q} | \Lambda_X) = \Pr(\mathcal{O}_Y | \mathbf{q}, \Lambda_X) \Pr(\mathbf{q} | \Lambda_X) \quad (26)$$

$$= \left[ \prod_{t=1}^T \Pr(\mathbf{y}_t | q_t, \Lambda_X) \right] \Pr(\mathbf{q} | \Lambda_X) \quad (27)$$

The total probability can be found by summing over all possible state sequences:

$$\Pr(\mathcal{O}_Y | \Lambda_X) = \sum_{\text{all } \mathbf{q}} \Pr(\mathcal{O}_Y, \mathbf{q} | \Lambda_X) \quad (28)$$

The typical approach for speech recognition is to use Viterbi decoding to generate only the single most likely state sequence  $\hat{\mathbf{q}}$ , and compute  $\Pr(\mathcal{O}_Y | \Lambda_X)$  conditioned upon this sequence. Substituting in the state output probabilities  $b_j(\mathbf{y}_t)$  from Equation 9 and transition probabilities  $a_{ij}$ 's,

$$\Pr(\mathcal{O}_Y, \hat{\mathbf{q}} | \Lambda_X) = \pi_{q_1} b_{q_1}(\mathbf{y}_1) a_{q_1 q_2} b_{q_2}(\mathbf{y}_2) \dots a_{q_{T-1} q_T} b_{q_T}(\mathbf{y}_T). \quad (29)$$

From this the standard approach to speech recognition can be seen as the MAP classifier optimal in the sense of minimizing expected word errors. Since the model parameters  $\Lambda_X$  and  $\Gamma$  are found from training data, and are not necessarily true representations of the underlying distributions, this approach is often referred to as the *plug-in* MAP classifier, as the best guess at model parameters are plugged-in, in place of the true distributions.

This method can be extended to what is sometimes referred to as the Bayes Predictive Classifier (BPC). BPC also solves for a MAP solution, but allows for data (or model parameters themselves) to be specified by probability distributions, conditioned on some other set of (known) parameters, in this case the mismatch model  $\mathcal{W}$ . Following the analysis given in [53], the method of handling unknown parameters is to integrate them out from the conditional distribution, given the data. Performing this on Equation 25 yields

$$\hat{W} = \arg \max_W \int_{\mathcal{O}_X} p(\mathcal{O}_X | \mathcal{O}_Y, W, \Lambda_X, \mathcal{W}) p(\mathcal{O}_X | W, \Lambda_X, \mathcal{W}) \Pr(W | \Gamma, \mathcal{W}) d\mathcal{O}_X \quad (30)$$

$$= \arg \max_W \left[ \int_{\mathcal{O}_X} p(\mathcal{O}_X | \mathcal{O}_Y, W) p(\mathcal{O}_X | W, \Lambda_X) d\mathcal{O}_X \right] \Pr(W | \Gamma) \quad (31)$$

Only the bracketed term in Equation 31 is dependent upon on the mismatch between training and testing conditions,  $\mathcal{W}$ . This term can be found by summing the probability

of the single state sequence  $\mathbf{q}$  over all possible state sequences. As was done for standard method above, the probability of some state sequence  $\mathbf{q}$  can be found using Markov independence to be

$$\int_{\mathcal{O}_X} p(\mathcal{O}_X, \mathbf{q} | \mathcal{O}_Y, \mathcal{W}) p(\mathcal{O}_X, \mathbf{q} | \Lambda_X) d\mathcal{O}_X = \quad (32)$$

$$= \left[ \int_{\mathcal{O}_X} p(\mathcal{O}_X | \mathbf{q}, \mathcal{O}_Y, \mathcal{W}) p(\mathcal{O}_X | \mathbf{q}, \Lambda_X, \mathcal{W}) d\mathcal{O}_X \right] \Pr(\mathbf{q} | \Lambda_X) \quad (33)$$

$$= \left[ \prod_{t=1}^T \int_{\mathbf{x}} p(\mathbf{x} | y_t, \mathcal{W}_t) p(\mathbf{x} | q_t, \Lambda_X) d\mathbf{x} \right] \Pr(\mathbf{q} | \Lambda_X) \quad (34)$$

Recognition can then follow the same path as described in Equations 25 to 29, except that the state output probabilities are calculated as

$$\Pr[y_t | q_t = j, \mathcal{W}_t] = \int_{\mathbf{x}} p(\mathbf{x} | y_t, \mathcal{W}_t) p(\mathbf{x} | q_t = j) d\mathbf{x} \quad (35)$$

which is identical to that developed for UO decoding in Equation 15.

From this development, it is now clear that a BPC decoding formulation (which includes knowledge that the observation vectors themselves are uncertain) is identical to standard decoding, except that the state output probability calculation of Equation 9 is replaced with the UO state output probability of Equation 15. Thus UO decoding inherits the properties of a BPC classifier. UO is thus the optimal decoder, given model parameters  $\Lambda_X$ , mismatch model  $\mathcal{W}$ , and features described as conditional PDFs.

A method related to this was described in [31], except in that work it was the model parameters,  $\Lambda$ , that were considered unknown, as they had changed due to the mismatch between training and testing environments. Thus that method was another form of model adaptation.

### 3.5 Special Cases

Several special cases are worth considering. First is the case of absolutely *known* feature vectors (i.e., when no distortion is present). Second, is the case of completely *unknown* feature vectors. Finally, the important special case of Gaussian observation PDFs will be covered, as this form is how UO decoding will actually be implemented in the remainder of the thesis. Both single- and multi-mixture normal random variables will be considered.

Finally, uniform PDFs will be presented, as an example that any PDF could potentially be used to model observation likelihoods.

### 3.5.1 Known Undistorted Feature Vectors

An analysis of the decoding algorithm of this thesis shows that in the absence of noise, the recognizer will perform as well as a matched-condition system. This can be seen by noting that when it is known that there is no distortion present, the observed feature vector  $\mathbf{y}_t$  is exactly equal to the clean, ideal feature vector  $\mathbf{x}_t$ . This can be modeled in the PDF feature domain as a Dirac delta function:

$$f_t(\mathbf{x}) = \delta(\mathbf{x} - \mathbf{x}_t) \quad (36)$$

Plugging this into the UO integral gives

$$E[\Pr[\mathbf{x}|q_t=j] | \mathbf{y}_t, \mathcal{W}_t] = \Pr[\mathbf{y}_t|q_t=j, \mathcal{W}_t] \quad (37)$$

$$= \int_{-\infty}^{\infty} f_t(\boldsymbol{\vartheta}) \cdot b_j(\boldsymbol{\vartheta}) d\boldsymbol{\vartheta} \quad (38)$$

$$= \int_{-\infty}^{\infty} \delta(\boldsymbol{\vartheta} - \mathbf{x}_t) b_j(\boldsymbol{\vartheta}) d\boldsymbol{\vartheta} \quad (39)$$

$$= b_j(\mathbf{x}_t) \quad (40)$$

$$= \Pr[\mathbf{x}_t|q_t=j] \quad (41)$$

which is exactly the state output probability calculation for standard HMM decoding.

The conclusion that can be drawn from this special case is that in the absence of any distortion, application of the UO decoding algorithm will not adversely affect recognition, retaining optimal decoding for matched conditions.

### 3.5.2 Completely Unknown Feature Vectors

Another special case is when some distortion has completely obliterated a feature vector or series of feature vectors. This can be the case in extremely noisy environments with very low signal to noise ratio, or in a Voice over IP or wireless communication system where groups of packets of audio data can occasionally be delayed long enough to require being considered lost for the purposes of real-time communication. For very short bursts of noise

or dropped packets some type of interpolation of feature vectors is possible, as is done in current speech coding algorithms, such as G.729 and MELP. This is referred to as Packet Loss Concealment. For this section, however, the assumption is made that interpolation is not possible, and no information at all can be obtained about the audio signal.

The feature PDF for such a case can be represented as a uniform random variable

$$f_t(\mathbf{x}) = u(\mathbf{x}) = u_0 \quad (42)$$

across the entire range of the finite feature space  $\mathcal{R}$ , representing that any one feature vector is equally likely as any other. Using the identity that the integral of any probability density function (in this case  $b_j(\mathbf{x})$ ) across its entire range must equal to 1, UO decoding using Equation 15 gives:

$$E[\Pr[\mathbf{x}|q_t=j] | \mathbf{y}_t, \mathcal{W}_t] = \Pr[\mathbf{y}_t|q_t=j, \mathcal{W}_t] \quad (43)$$

$$= \int_{\mathcal{R}} f_t(\vartheta) \cdot b_j(\vartheta) d\vartheta \quad (44)$$

$$= \int_{\mathcal{R}} u(\vartheta) b_j(\vartheta) d\vartheta \quad (45)$$

$$= u_0 \int_{\mathcal{R}} b_j(\vartheta) d\vartheta \quad (46)$$

$$= u_0 \quad (47)$$

Thus, the score for every state  $j$  in the model will be equal to the same constant  $u_0$ . This implies that acoustic information will penalize every state and every phoneme equally in all projected paths internal to the HMM decoder. Decoding will thus rely solely on state duration and context information available from the language model, as would happen if a human were performing the recognition problem. Unreliable acoustic information is ignored, and this is the entirely desirable result.

Obviously the scenario of completely unknown vectors is somewhat unlikely. With interpolation, multiple description coding, or other packet loss concealment techniques, only in cases of multiple dropped frames spanning a phoneme changes would acoustic information be completely unknown. And then, only in cases where duration and language modeling could still span the lost acoustic information would robust recognition schemes even be relevant. Thus for many communication applications, a more appropriate model is one

showing information loss as a growing variance as length of a burst of unknown feature vectors grows in length. This subject is covered in much more detail in Section 5.3 for the specific case of bursts of lost packets on a communications channel.

### 3.5.3 Gaussian Observation PDFs

The typical modern HMM-based speech recognition system uses Gaussian mixture distributions to model state output likelihoods  $b_j(\mathbf{y}_t)$ , as described in Section 2.1. Given this, a very efficient implementation of UO decoding is to model observations PDFs as single mixture Gaussian of mean  $\boldsymbol{\mu}_t$  and covariance  $\boldsymbol{\Sigma}_t$ :

$$f_t(\mathbf{x}) = \mathcal{N}(\mathbf{x}; \boldsymbol{\mu}_t, \boldsymbol{\Sigma}_t). \quad (48)$$

For many distortions and feature extraction methods, this is a reasonable method of representation, as will be shown for various applications in Chapters 4, 5 that follow.

Substituting the single mixture Gaussian observation PDF into the UO decoding integral in Equation 15 gives

$$E[\Pr[\mathbf{y}_t | q_t = j] | \mathbf{y}_t, \mathcal{W}_t] = \int_{R^n} f_t(\boldsymbol{\vartheta}) \cdot b_j(\boldsymbol{\vartheta}) d\boldsymbol{\vartheta} \quad (49)$$

$$= \int_{R^n} \mathcal{N}(\boldsymbol{\vartheta}, \boldsymbol{\mu}_t, \boldsymbol{\Sigma}_t) \cdot \sum_{k=1}^K c_{jk} \mathcal{N}(\boldsymbol{\vartheta}, \boldsymbol{\mu}_{jk}, \boldsymbol{\Sigma}_{jk}) d\boldsymbol{\vartheta} \quad (50)$$

$$= \sum_{k=1}^K c_{jk} \int_{R^n} \mathcal{N}(\boldsymbol{\vartheta}, \boldsymbol{\mu}_t, \boldsymbol{\Sigma}_t) \mathcal{N}(\boldsymbol{\vartheta}, \boldsymbol{\mu}_{jk}, \boldsymbol{\Sigma}_{jk}) d\boldsymbol{\vartheta} \quad (51)$$

$$= \sum_{k=1}^K c_{jk} \mathcal{N}(\boldsymbol{\vartheta}; \boldsymbol{\mu}_{jk}, \boldsymbol{\Sigma}_{jk} + \boldsymbol{\Sigma}_t) \Big|_{\boldsymbol{\vartheta} = \boldsymbol{\mu}_t} \quad (52)$$

$$= \sum_{k=1}^K \left[ \frac{c_{jk}}{(2\pi)^{n/2} |\boldsymbol{\Sigma}_\tau|^{1/2}} \cdot \exp \left\{ -\frac{1}{2} (\boldsymbol{\mu}_t - \boldsymbol{\mu}_{jk})^T \boldsymbol{\Sigma}_\tau^{-1} (\boldsymbol{\mu}_t - \boldsymbol{\mu}_{jk}) \right\} \right] \quad (53)$$

where  $\boldsymbol{\Sigma}_\tau = \boldsymbol{\Sigma}_{jk} + \boldsymbol{\Sigma}_t$ , and  $k$  indexes mixtures within the  $j^{\text{th}}$  state's Gaussian mixture distribution.

To show how Equation 52 follows from Equation 50, begin with the well known equation

for the convolution of Gaussians, evaluated at zero:

$$c(0) = \int_{-\infty}^{\infty} f_1(x) \cdot f_2(u-x) dx \Big|_{u=0} = f_3(x) |_{x=0} \quad (54)$$

where

$$\begin{aligned} f_1(x) &\equiv \mathcal{N}(x; \mu_1, \Sigma_1) \\ f_2(x) &\equiv \mathcal{N}(x; \mu_2, \Sigma_2) \text{ and} \\ f_3(x) &\equiv \mathcal{N}(x; \mu_1 + \mu_2, \Sigma_1 + \Sigma_2) \end{aligned}$$

If we define  $f_4(x) = f_2(-x)$ , then  $f_4(x) \equiv \mathcal{N}(x; \mu_4, \Sigma_4)$ , where  $\mu_4 = -\mu_2$  and  $\Sigma_4 = \Sigma_2$ . Thus  $f_4(-x) \equiv \mathcal{N}(x; \mu_2, \Sigma_2)$ . Substituting this into Equation 54 gives

$$\int_{-\infty}^{\infty} f_1(x) f_2(u-x) dx \Big|_{u=0} = f_3(x) |_{x=0} \quad (55)$$

$$\int_{-\infty}^{\infty} f_1(x) f_4(x) dx = f_3(x) |_{x=0} \quad (56)$$

with  $f_3(x) \equiv \mathcal{N}(x; \mu_1 - \mu_4, \Sigma_1 + \Sigma_4)$  which is sufficient to show Equation 52.

The important consequence with this special case is that if a single mixture Gaussian can be used to represent the feature PDF, then the state output probability calculation itself will remain a  $K$  mixture Gaussian evaluation, and is thus identical in complexity to standard HMM decoding. The only extra computations required are only those to generate the new inverse covariance matrix  $\Sigma_{\tau}^{-1}$ , and the covariance determinant  $|\Sigma_{\tau}|$ . Finding these is required on a per-frame basis, and thus is a nontrivial extension, but the overall cost is still less than a factor of two. Section 3.6 will cover the computational complexity of this particular formulation in detail, including numerous shortcuts.

### 3.5.4 Gaussian Mixture Observation PDFs

The previous formulation using a single mixture Gaussian for the observation PDF can be extended to further cover the case of having a Gaussian mixture model (GMM) representation with  $L$  mixtures. The state output probability is

$$E \left[ \Pr [y_t | q_t = j] | y_t, \mathcal{W}_t \right] = \int_{R^n} f_t(\vartheta) \cdot b_j(\vartheta) d\vartheta \quad (57)$$

$$= \int_{R^n} \left( \sum_{l=1}^L c_{t,l} \mathcal{N}(\vartheta; \mu_{t,l}, \Sigma_{t,l}) \right) \left( \sum_{k=1}^K c_{j,k} \mathcal{N}(\vartheta; \mu_{j,k}, \Sigma_{j,k}) \right) d\vartheta \quad (58)$$

$$= \sum_{l=1}^L \sum_{k=1}^K c_{t,l} c_{j,k} \int_{R^n} \mathcal{N}(\vartheta; \mu_{t,l}, \Sigma_{t,l}) \cdot \mathcal{N}(\vartheta; \mu_{j,k}, \Sigma_{j,k}) d\vartheta \quad (59)$$

$$= \sum_{l=1}^L \sum_{k=1}^K c_{t,l} c_{j,k} \mathcal{N}(\vartheta; \mu_{j,k}, \Sigma_{j,k} + \Sigma_{t,l}) \Big|_{\vartheta = \mu_{t,l}} \quad (60)$$

There are two important issues with this GMM situation. First, is that in the case of the single Gaussian feature PDF the state output probability equation remained similar in computational complexity to the original decoding equation, requiring only the evaluation of a  $K$  mixture Gaussian. For the GMM formulation, a total of  $L$  such evaluations are required. Thus this is a computationally expensive extension to the basic single Gaussian evaluation.

Further, it will be shown later in this chapter that it is important that the feature PDF be accurate. As with any technique which uses a model to simplify the description of a real world scenario, when the model is inaccurate or used improperly, performance will drop. The GMM feature model requires more parameters, and thus is more susceptible to improper modeling unless caution is exercised in its application. In Section 4.3.2 we cover a novel method for PDF feature extraction that is actually appropriate for the GMM scenario.

### 3.5.5 Uniform Observation PDFs

The general form of the UO decoding equation allows for any observation PDF to be chosen. Thus, while most implementations would no doubt use the Gaussian or mixture Gaussian

versions discussed above, it is of course possible to use other forms. One obvious example is the uniform distribution,

$$f_t(\mathbf{x}) \sim \mathcal{U}(\mathbf{a}, \mathbf{b}) \quad (61)$$

Consider, for example, the case of a vector quantization system. The full feature space is broken into  $I$  regions and during compression each observation  $\mathbf{x}_t$  is assigned to one of these regions. During transmission, only the index  $i$  of the region is sent. If this index were used as input to a UO decoder, and the assumption is made that any point within the region is equally likely, then the appropriate observation PDF is a uniform random variable covering region  $i$  of the feature space.

The state output probability for  $d$ -dimensional feature vectors can, for the case of a uniform observation RV that is separable across dimensions, be written as

$$E\left[\Pr[y_t|q_t=j] | y_t, \mathcal{W}_t\right] = \int_{R^n} f_t(\vartheta) \cdot b_j(\vartheta) d\vartheta \quad (62)$$

$$= \int_{\mathbf{a}}^{\mathbf{b}} \frac{1}{\mathbf{b} - \mathbf{a}} b_j(\vartheta) d\vartheta \quad (63)$$

$$= \sum_{d=1}^D \frac{1}{b_d - a_d} (F_j(b_d) - F_j(a_d)) \quad (64)$$

where  $d$  indexes the dimension, and  $F_j(x)$  is the cumulative distribution function for state  $j$  evaluated at  $x$ .

### 3.6 Complexity Analysis

There are several components required to analyze the computational complexity of the UO decoding algorithm. First, the model for the current distortion must be generated. For some scenarios this model does not change, for example, a global model of a speech compression algorithm. For cases scenarios that truly harness the full capabilities of UO decoding, however, this distortion model is time varying. The best example of this is speech recognition in the presence of nonstationary background noise. For such a scenario, the distortion model

$$\mathcal{W} = (\mathcal{W}_1, \mathcal{W}_2, \dots, \mathcal{W}_T) \quad (65)$$

and some amount of computation is undoubtedly required to generate the new noise model  $\mathcal{W}_t$  for each frame  $t$ . This computation of course is dependent upon the type of distortion, the nature of the model, and the particular implementation chosen, and thus further general analysis cannot be made here.

Similarly,  $\mathcal{W}$  must be used in order to generate the observation PDF sequence,  $\mathcal{O}' = (f_1(\mathbf{x}), f_2(\mathbf{x}), \dots, f_T(\mathbf{x}))$ , and the complexity of this operation is implementation dependent. Several methods for computing such a sequence will be covered in this thesis. The first methods, described in Chapter 4, give several techniques for use during nonstationary additive background noise. The methods range in computational complexity from being an extremely efficient mapping lookup, to being an iterative Monte-Carlo based estimation scheme requiring more computations than speech recognition itself. Other methods described in Chapter 5 highlight PDF sequence generation to handle distortions encountered in distributed speech recognition. Compressed feature vectors, compressed speech waveforms, and packet-lossy channels are modelled as uncertain features, and compensated using UO decoding. The computational complexity for generating an observation PDF sequence in each of these cases will be shown to be minimal in relation to other aspects of recognition.

As for the actual decoding process, the amount of computation is related to the choice of the form of the observation PDF. For example, a Dirac delta function described in Section 3.5.1 implies no extra processing. For the uniform distribution of Section 3.5.5, a large number of Gaussian cumulative distribution functions must be evaluated.

Probably the most useful observation PDF is the single mixture Gaussian described in Section 3.5.3. Depending on whether a full or diagonal covariance matrix is used, the amount of extra computations required by UO decoding can be quite limited. The remainder of this section will look at the Gaussian case in more detail.

For the Gaussian observation PDF, the state output probability computation remains just the evaluation of a single Gaussian. Due to UO, however, there are two changes. First is the evaluation location, which shifts from  $\mathbf{y}_t$  to the mean of the observation Gaussian,  $\hat{\mathbf{x}}_t$ . This obviously requires no extra processing. Second is the change to the covariance

matrix. The new covariance matrix,  $\Sigma_\tau$ , is a function of both the covariance of the state being evaluated,  $\Sigma_{jk}$ , and the covariance of the observation PDF,  $\Sigma_t$ . Being a function of the observation, the new covariance cannot be calculated until it is time to decode the frame. Assuming that speech is modeled with a diagonal covariance Gaussian, as is almost always the case,  $\Sigma_\tau$  can be generated by  $N$  additions regardless of the type of observation covariance matrix.

Because a new covariance matrix is generated for each frame,  $\Sigma_\tau^{-1}$  must be recomputed. For the general case of full covariance matrices, this is a computationally complex evaluation, although the positive-definite symmetric form of covariance matrices does make the problem much smaller than general matrix inversion. Using a linear algebra package such as LAPACK, this is accomplished by a Cholesky factorization (`spotrf` in LAPACK), followed by a Cholesky-factorization-specific matrix inverse (`spotri`), giving an  $\mathcal{O}(N^2)$  operation. If the observation PDF also uses a diagonal representation, the inverse only requires  $N$  divides.

A further amount of computation comes from needing a new determinant,  $|\Sigma_\tau|$ . In any current ASR system, the determinant of  $|\Sigma_{jk}|$  is a precomputed value stored inside the acoustic model. For UO decoding, this is no longer possible and must also be computed for each frame. In both the full and diagonal covariance cases, this requires another  $N - 1$  multiplies, multiplying along the Cholesky factorization diagonal for full covariances, and along the diagonal for diagonal covariances.

If the implementation-specific computation of the observation PDF is ignored, the resulting computational burden of the UO state output probability calculation with diagonal covariance observation PDFs is roughly 5 floating point operations for every 3 required by standard decoding. In a typical large vocabulary system, these Gaussian evaluations are roughly 50% of the processing burden [50] (with language modeling being the majority of the rest), resulting in the extra processing is expected to be approximately 33% of the overall system load. Experiments timing only the output probability calculation verified this value, as is shown in Table 3. Also shown are experimental results for full covariance Gaussians in UO decoding when using 13-dimensional feature vectors.

**Table 3:** Computational overhead for UO Decoding, with 13-dimensional feature vectors.

| Observation PDF Model                            | Covariance Structure | Order of UO Overhead | Experimental Results, Timing |
|--|----------------------|----------------------|------------------------------|
| Dirac delta, $\delta(\mathbf{x} - \mathbf{y}_t)$ | N/A                  | N/A                  | 1.0 $\times$                 |
| single mixture Gaussian                          | diagonal             | $\mathcal{O}(N)$     | 1.3 $\times$                 |
| $K$ mixture Gaussian                             | diagonal             | $\mathcal{O}(KN)$    | 1.3 $K$ $\times$             |
| single mixture Gaussian                          | full                 | $\mathcal{O}(N^2)$   | 32.2 $\times$                |

This extra computational burden is, however, not necessarily required. Consider the case of ASR in the presence of additive background noise. The dynamic range of speech is large enough that the local SNR of frames, even closely related in time, can vary significantly. While some low energy frames will be completely lost and many frames somewhat affected by the noise, some high energy frames occurring during voiced speech will only be slightly affected by the noise. For frames such as these, the observation PDF  $f_t$  will have component variances along the covariance matrix diagonal small enough that the PDF can be effectively reduced to a Dirac delta function. As shown in Section 3.5.1, this reduces to nothing more than using the standard decoding output probability equation given in Equation 9. Covariance matrix updates are no longer necessary, and the extra computational burden due to UO is zero for this type of frame. In moderate noise, a similar situation can occur for nonspeech frames. In such a scenario, a voice activity detector can be quite successful at finding near enough the onset and end of speech activity. Nonspeech frames can thus be easily replaced with by a “typical” nonspeech feature vector, and the Dirac delta approximation again used.

An example of the computational complexity required during additive noise environments can be seen in Table 4. The experiments in this table are based on the 0dB SNR subset of the Aurora2 noisy digit experiments described in detail later in Section 4.3.1. Four recognition systems are used. First is a baseline recognizer using mismatched noisy observation with clean-trained models. Second is the same system, but using the MCMC-AR feature transformation method that will be described in Section 4.3.1. Third is the same as the second, but now using UO on all frames during decoding. The observation PDFs

are single mixture Gaussians with diagonal covariance matrices. Fourth, is the UO system with the ability to replace certain frames with Dirac delta PDFs as described above. Note that these experiments are for a small vocabulary example with no language model (any digit is allowed to follow any other digit with equal likelihood). Computational overheads would thus proportionally decrease in LVCSR systems employing extensive language modeling during the search. If fifty percent of the computation is again assumed to be language modeling, this shows that UO can, in a noisy environment, be implemented with only a 15% continuous computational overhead when compared to the standard speech decoding method. The tradeoff is performance. While the baseline UO method is able to remove around 10% of the errors after feature transformation, the frame selective method is only half as effective, removing around 5% of the errors.

Table 4: Computational burden of UO decoding in an additive noise environment with no language modeling.

| Recognition system     | Time required (minutes) | Word Error Rate | Computational Overhead |
|------------------------|-------------------------|-----------------|------------------------|
| standard               | 9.7                     | 84.7%           | —                      |
| feature transformation | 9.7                     | 37.6%           | —                      |
| baseline UO            | 14.9                    | 34.3%           | 53%                    |
| frame-selective UO     | 12.6                    | 35.8%           | 30%                    |

Another implementation issue related to algorithm complexity is that of memory. Since in a modern recognition server there might perhaps be hundreds of input streams being decoded simultaneously, it is beneficial to reduce the amount of stream-dependent memory required. While a few parameters are of little consequence (for example, the single number representing the vocal tract scaling factor, for Vocal Tract Length Normalization), typical model compensation techniques such as PMC require completely new sets of model parameters. In a large vocabulary system, this can be several million new parameters that must be stored for each stream, resulting in potentially several gigabytes of stream-specific model data for a speech server. The UO framework, however, does not require any updates to the original speech models  $\Lambda_x$  during decoding. Thus the same speech models can be shared among all input streams. In this respect, the UO technique has significant memory benefits

over other model compensation techniques.

An final issue with UO complexity is that there is an extra bandwidth cost associated with distributed speech recognition environments. This cost is in the feature extractor, which now must output two parameters (observation PDF  $f_t(\mathbf{x}) \sim \mathcal{N}(\mathbf{x}; \boldsymbol{\mu}_t, \boldsymbol{\Sigma}_t)$ ) for what used to be just one (speech feature vector  $\mathbf{y}_t$ ), assuming a single-mixture Gaussian as in Section 3.5.3.

### *3.7 Experiments with UO Decoding*

The basic performance of the Uncertain Observation decoding algorithm will be demonstrated through several experiments in this section. First, the recognition system used throughout the experiments of this thesis, both in this chapter and in later application chapters, is described in detail in Section 3.7.1. This is followed by discussions of several types of baseline experiments. Artificial distortions are used in these experiments in order to allow the performance of UO decoding to be simulated in a controlled manner under several conditions.

#### **3.7.1 The Recognition Engine**

The source code for the training and decoding algorithms was originally taken from Mississippi State University’s publicly available prototype research recognition platform for large vocabulary speech recognition created by Institute for Signal and Information Processing (ISIP) department [49]. The Uncertain Observation decoding algorithm was integrated into this framework, allowing several types of observation PDFs:

- multimixture delta functions,
- single mixture, diagonal covariance Gaussians,
- multimixture, diagonal covariance Gaussians, and
- single mixture, full covariance Gaussian.

The specific model chosen from this set is flexible, and can be changed on a per-frame basis. While other options for observation PDFs such as uniform, Laplacian, truncated Gaussian,

or other random variables could certainly be implemented, this set was deemed rich enough for the research of this thesis.

### 3.7.2 Experimental Setup

In order to test the UO decoding algorithm, an experiment that deals only with decoding based on acoustic information was chosen. Phoneme recognition of TIMIT data [26] was the goal, with no language modeling allowed, thus giving a loop grammar with any phoneme allowed to follow any other phoneme, and with no phrase length constraints.

Phonemes were modeled by three state, 16 mixture Gaussian, in a left to right model. These models were trained with the Baum Welch algorithm in the usual manner using the ISIP tools.

The ISIP recognition system is feature-set independent. That is, feature extraction happens separately from training or recognition, and any reasonable set can be used. For the first three experiments of this section, log energy plus 12 MFCCs generated from a 24 band mel spaced filterbank, with first and second derivatives were extracted, giving a feature dimension of 39. The final experiment deals compares the performance of full covariance matrices to diagonal covariance matrices, and for these experiments, only the static features were used, both in the filterbank log energy domain and in the MFCC domain, giving feature dimensions of either 24 and 13, respectively.

### 3.7.3 Oracle Experiment 1: Stationary distortion

A first set of experiments was designed to show the ideal operating point of UO decoding. This was accomplished by generating a disturbance vector  $\mathbf{n}_t$  pulled from a Gaussian distribution of known mean and variance,  $\mu_t$  and  $\Sigma_t$ , respectively, and adding this disturbance directly to clean cepstral parameters, as shown in Figure 16, rather than adding noise to the time domain speech signal. The mean and variance parameters  $\mu_t$  and  $\Sigma_t$  are then shown exactly to the decoder (although *not* the disturbance vector itself,  $\mathbf{n}_t$ ). Since the characteristics of the disturbance sequence are known exactly, these are referred to as “Oracle” experiments.

For this experiment, some choice had to be made for the mean  $\mu_t$  and diagonal covariance

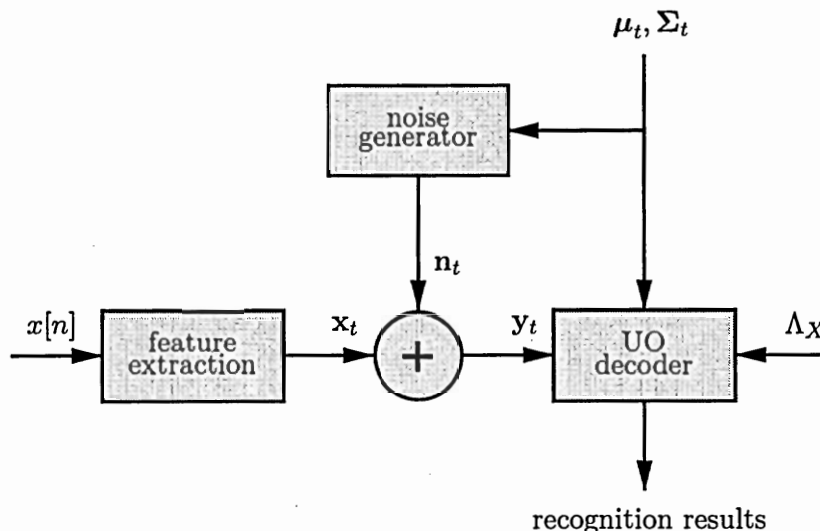


Figure 16: Block diagram of “Oracle” recognition experiments. Gaussian noise  $w_t$  of known mean and variance is added to clean cepstral vector  $x_t$ . Recognition then uses resulting sequence  $y_t$ .

$\Sigma_t$ . In order to pick reasonable values, the stereo Aurora2 database (described in detail in [47]) was used. MFCC feature extraction was performed on both the clean data and all of the 0dB SNR subset of the noisy data. The global mean vector  $\mu_a$  and variance vectors  $\Sigma_a$  of the difference between these two sets were then calculated, and  $\mu_t$  and  $\Sigma_t$  were set to these vector constants.

The baseline for this experiment is training and testing using clean, unaltered data. This matched condition scenario shows a 38.7% phoneme error rate (PER). A second baseline matched condition comes from using noisy data for both the training and testing. This is a representation of an ideal form of model adaptation, allowing all parameters of the HMM models (including transition matrix components  $a_{ij}$ 's) to be updated. This matched condition gives only a slightly lower error rate, at 54.7%. Because of the structure of this problem, it is not possible to perform PMC-style model adaptation, since there is no explicit model for the noise in the time domain.

Testing noisy data using clean-speech-trained models shows the difficulty of mismatched conditions. For this case, the error rate jumps to 100.6%. The first cut at improvement is to perform feature transformation. From a feature transformation perspective, this experiment

scenario allows an optimal  $\hat{\mathbf{x}}_t$  to be found as follows,

$$\hat{\mathbf{x}}_t = E[\mathbf{x}_t] \tag{66}$$

$$= E[\mathbf{y}_t - \mathbf{n}_t] \tag{67}$$

$$= \mathbf{y}_t - \boldsymbol{\mu}_a, \tag{68}$$

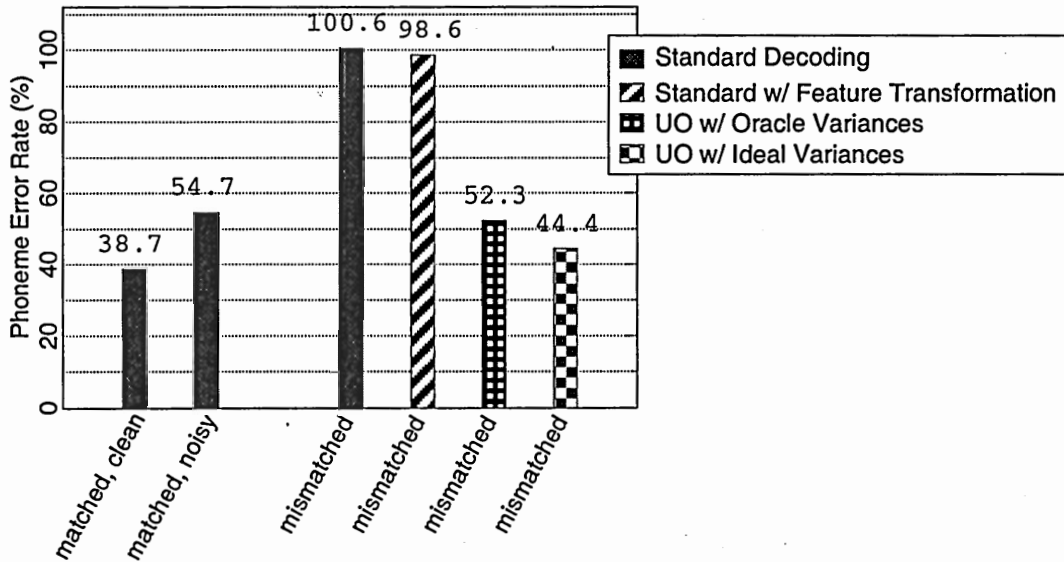
since  $E[\mathbf{n}_t] = \boldsymbol{\mu}_a$  for the disturbance is known exactly.

While this form of ideal feature transformation shows an improvement, it is minimal. The PER drops to 98.6%, still far from the matched condition level. This is of course because the disturbance for this experiment was almost zero mean, and therefore feature transformation has little effect. However, it does show that variance information can certainly hurt performance if not accounted for.

Finally, UO decoding was implemented using all of the known information: the observation  $\mathbf{y}_t$ , the disturbance mean  $\boldsymbol{\mu}_a$ , and the disturbance's diagonal covariance matrix,  $\boldsymbol{\Sigma}_a$ . The simple inclusion of the feature variance information showed a dramatic drop in the error rate, down to 52.3%. This approaches the matched condition case of training on noisy data. The slightly lower performance (compared to matched noisy) is completely expected, since the matched noisy case has also allowed for transition matrix updates, as well as state output probability parameter updates.

A different form of ideal UO decoding was also implemented for comparison. This method will be described throughout this thesis as the "Ideal Variance" method. An ideal variance is a covariance matrix with diagonal entries calculated by squaring the difference between the clean and distorted feature vectors. Thus for this method  $\boldsymbol{\Sigma}_t = (\mathbf{y}_t - \mathbf{x}_t)^2 = \mathbf{n}_t^2$ , meaning both the clean features and distorted features must both be given to the decoder. For this ideal form of UO decoding, the error rate drops significantly, down to 44.4%.

The experiment results, shown in Table 17, show clearly that feature transformation only partially compensates for noisy conditions. Uncertain Observation decoding, however, can almost completely eliminate the mismatch if observation PDFs can be accurately found.



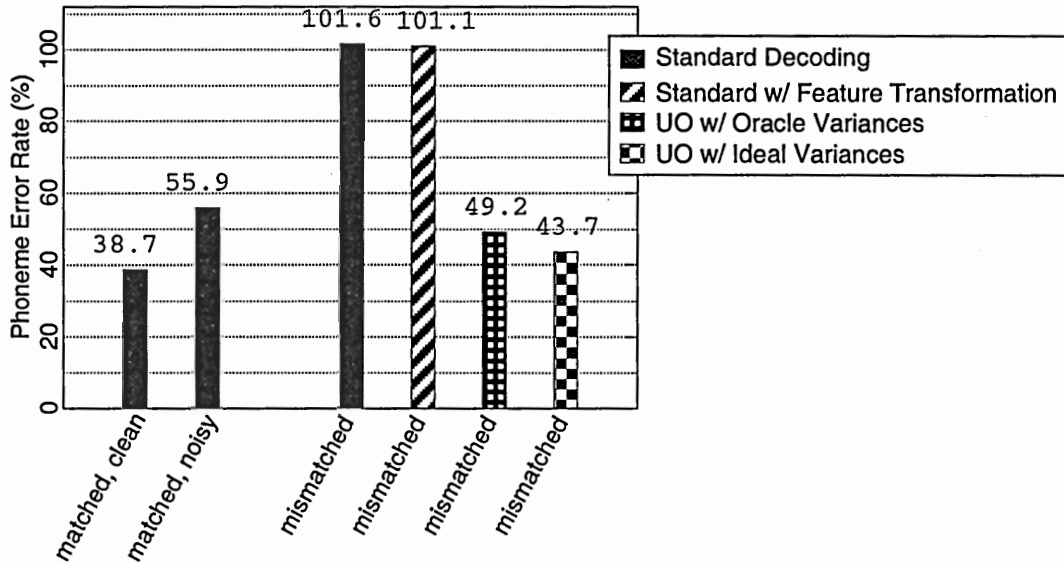
**Figure 17:** Oracle 1 experiment results for phoneme error rates on TIMIT data. Noisy, in this case, refers to a disturbance vector,  $\mathbf{n}_t$ , drawn each frame  $t$  from a known Gaussian distribution, and being added to the clean MFCC vectors.

### 3.7.4 Oracle Experiment 2: Time-varying distortion

In the previous experiment, UO decoding was shown to significantly reduce recognition error rates in the case where the parameters of a Gaussian disturbance vector were completely known. The disturbance for the experiment, however, did not change with time – the same mean and covariance were used for all frames. In this section, the case of a time varying distortion are examined. By showing that UO decoding can still be successful in such time varying environments, it can be seen that successful decoding of speech in nonstationary noise can be successfully accomplished by good estimation of feature PDFs.

The same experiment setup was used as in the previous section, but this time the parameters of the Gaussian from which  $\mathbf{n}_t$  is drawn were time varying (albeit still known, in a time varying manner, to the UO decoder). For this test, both the mean and variance were themselves drawn from a uniform random distribution,  $\mu_t \sim U[0, 2\mu_a]$ , and  $\Sigma_t \sim U[0, 2\Sigma_a]$ , respectively, rather than being set to constant vector values as in the previous experiment. The results are summarized in Table 18, showing that even with a time varying distortion,

UO allows for frame-by-frame updates to the decoding process, allowing significantly improved recognition accuracy that cannot be matched by simple feature transformation, and would be impractical for model compensation techniques.



**Figure 18:** Oracle 2 experiment results for phoneme error rates on TIMIT data. Noisy, in this case, refers to a disturbance vector,  $\mathbf{n}_t$ , drawn each frame  $t$  from a known Gaussian distribution with time varying parameters, and being added to the clean MFCC vectors.

The important consequence of this experiment is to observe that UO decoding is just as successful as in the first Oracle experiment in Section 3.7.3. For the first experiment, because the statistics of the distortion were not time-varying, UO decoding could just as easily have been implemented in a manner as model compensation. That is, given the distortion mean and variance  $\mu_a$  and  $\Sigma_a$ , and single new updated HMM model set  $\Lambda_Y$  could have been calculated prior to standard decoding, and would have been equally successful (as well as more efficient computationally). For the experiment in this section, however, because the distortion is rapidly time-varying, there is no equivalent model compensation method that simplifies to a *single* optimal updated model set  $\Lambda_Y$ . An optimal model compensation implementation would necessitate a new  $\Lambda_Y$  for each speech frame. This would require an impractical amount of computational overhead, eliminating it as a potential implementation.

UO decoding, however, gives model-compensation-quality results, and the overhead is no more than that of the first Oracle experiment.

### 3.7.5 Oracle Experiment 3: Erroneous covariance estimates

A potential pitfall to using observation uncertainty is when the variance estimates for a feature vector are incorrect. This section will briefly study the consequence of such an occurrence.

To examine this, an Oracle experiment setup identical to the Oracle 1 experiment in Section 3.7.3 was performed, but erroneous covariance information was passed to the UO decoder. Thus, the disturbance vectors  $\mathbf{n}_t$  are drawn from the distribution  $\mathcal{N}(\mathbf{x}; \boldsymbol{\mu}_a, \boldsymbol{\Sigma}_a)$ , but the decoder will be given distribution parameters  $\mathcal{N}(\mathbf{x}; \boldsymbol{\mu}_a, \boldsymbol{\Sigma}'_t)$ . The sequence of disturbance vectors was chosen to be exactly identical to that used for the Oracle 1 Experiment. For UO decoding, the mean  $\boldsymbol{\mu}_a$  was given correctly to the decoder, but the variance parameters given for the observation PDF were generated as a random variable,  $\boldsymbol{\Sigma}_t \sim \mathcal{U}(0, c\boldsymbol{\Sigma}_a)$ , with  $c$  being a scaling constant in order to evaluate a range of errors.

Results for scaling factors in the range  $c = 0$  to  $c = 10$  are shown in the graph of Figure 19. When  $c = 0$ , the system is operating as a standard decoder, using feature-transformed data, and thus gives identical results to that seen in Figure 17. When  $c = 2$ , and since the new variances are chosen using a uniform random variable, the mean variance  $E[\boldsymbol{\Sigma}_t]$  equals the Oracle variance,  $\boldsymbol{\Sigma}_a$ , and it is near this location that the error rate is near its minimum, although it never reaches as low as either the Ideal Variance performance (44.4% PER) or the Oracle performance (52.3% PER) found in the Oracle 1 experiment.

A potential problem is clearly identified: if the variance information is incorrect by a large enough amount, the error rates from the decoder will be larger than if no robust processing were done at all. It is a typical problem: the more complex the model used for a system, the more unstable the system becomes when conditions outside the model are encountered. This concept needs to be remembered when designing a PDF feature extractor, as will be done for various applications in Chapters 4 and 5.

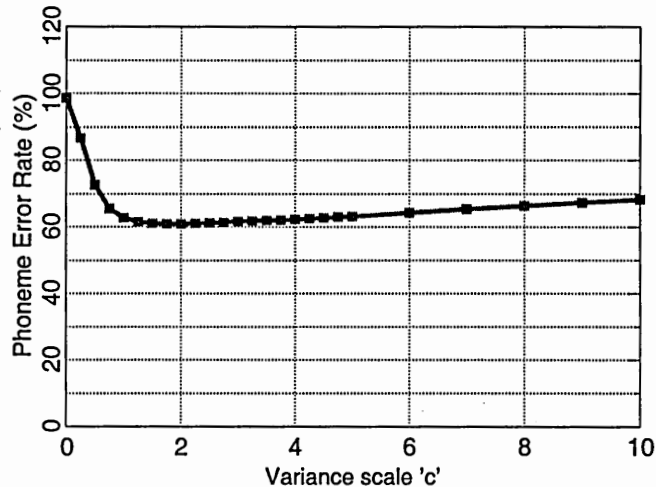


Figure 19: Oracle 3 experiment results for phoneme error rates on TIMIT data showing the sensitivity to incorrect knowledge of observation PDF variance.

### 3.7.6 Oracle Experiment 4: Comparison of full and diagonal observation covariance matrices

One potential variance estimation problem alluded to in the analysis section 3.6 is the case of an observation PDF that is not well represented by a Gaussian with diagonal covariance matrix. An especially relevant case for speech recognition is additive noise. In this case, there are two feature domains with significantly different properties: mel filterbank log energy features, and mel filterbank cepstral coefficient (MFCC) features.

Feature vectors in the filterbank log energy domain can be considered as having a reasonable amount of statistical independence due to additive noise. Since the transformation from the log energy domain to the MFCC domain uses a discrete cosine transform, and therefore a full matrix multiplication, to decorrelate speech features,  $\Sigma_t$  is now best described by a full covariance matrix. This is exactly opposite the standard argument made for modeling speech features in HMMs, as shown in Table 5.

In this section, an experiment similar to the Oracle approach presented in the previous sections is designed to show the effect of diagonalizing, in the MFCC domain, a feature PDF that is better represented as being diagonal in the filterbank log energy domain. Unlike the previous Oracle experiments, only static features are used, which means the baseline

Table 5: Covariance matrix forms in order to well-represent speech models and noisy features in the filterbank log energy and MFCC domains.

| Feature Vector<br>Domain | $\Sigma_{jk}$ format for<br>speech models | $\Sigma_t$ format for<br>noisy speech |
|--------------------------|---|---------------------------------------|
| Filterbank log energy    | full                                      | diagonal                              |
| MFCC                     | diagonal                                  | full                                  |

error rates are higher, however comparisons of different techniques within this experiment are valid.

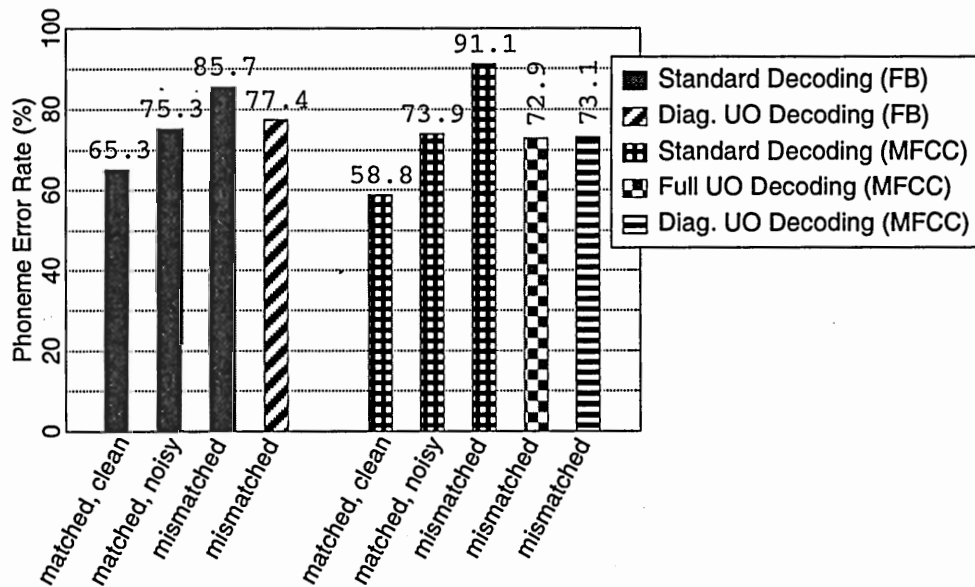
The clean dataset is the TIMIT data as used in previous Oracle experiments. To generate a distorted dataset, the same technique as as for the first Oracle experiment was used, that is, the global mean and variance of the noise signal of the 0dB SNR portions of the Aurora2 dataset were calculated. This mean and variance were then used to generate an artificial distortion vector  $\mathbf{n}_t$  to be added to the clean signal,  $\mathbf{x}_t$ , for each frame  $t$ . Unlike the first Oracle experiment, however, the feature vector domain is the filterbank log energy domain. The standard baseline experiments were run giving matched clean error rates of 65.3%, and matched distorted error rates of 75.3%. Converting from these feature vectors to the MFCC domain, a second set of baseline error rates were observed, of 58.8% for matched clean conditions, and 73.9% for matched distorted conditions. The results for the MFCC domain show lower error rates as expected, since speech models with diagonal covariance matrices are more appropriate.

To test mismatched conditions, models were trained with clean speech and tested with distorted data. In the filterbank domain, the error rate increased to 85.7%. The MFCC domain showed more sensitivity to the distortion, with the error rate increasing to 91.1%.

The distortion PDF  $\mathbf{n}_t$  was drawn from a multidimensional Gaussian with diagonal covariance matrix, as in the previous Oracle experiments. Thus optimal Oracle UO decoding for the filterbank data uses a feature PDF representation that has the same diagonal covariance matrix. For this case, the error rates drop to a 77.4% phoneme error rate.

In the MFCC domain, there are two ways that Oracle UO decoding can be tested. The optimal method requires that the diagonal covariance matrix representing the filterbank

domain distortion be transformed into a full covariance matrix in the MFCC domain. An approximation of this is to then reduce this to only a diagonal MFCC feature covariance. For the optimal version, the error rate drops to 72.9%. The diagonal approximation shows an error rate slightly higher, of 73.1%. The full set of results are reviewed in the graph of Figure 20.



**Figure 20:** Oracle 4 experiment comparing results for full and diagonal observation covariance matrices for a distortion that is independent on filterbank log energy (FB) features.

It is worth noting the processing time required for each of the MFCC experiments, as this gives further insight into the recognition problem. Each experiment was run on an otherwise idle 1GHz x86 PC, and the times were required recorded. The relative speeds, as a function of real time, are given in Table 6.

A simple scaling of time required based upon the state output probability calculation was actually not observed. This is because for all experiments, the pruning parameters were fixed, and thus the unexpected time differences are because of some conditions requiring the evaluation of more paths through the lattice than others.

There are several notes to observe from this set of experiments. First, is that as expected, the MFCC domain gives superior matched-condition performance compared to the

**Table 6:** Comparison of the relative speeds of the various MFCC tests within the Oracle 4 Experiment

| Decoding Method | Testing Data | Speed (times real time) | Speed (rel. to standard/clean) |
|-----------------|--------------|-------------------------|--------------------------------|
| standard        | clean        | 0.093                   | $\times 1.00$                  |
| standard        | distorted    | 0.076                   | $\times 0.82$                  |
| diag UO         | distorted    | 0.180                   | $\times 1.93$                  |
| full UO         | distorted    | 2.26                    | $\times 24.5$                  |

filterbank log energy domain. This would lead, of course, towards the adoption of MFCC features. The next observation is that the MFCC domain is unfortunately less robust to the distortion used in this experiment than the filterbank domain, leading to the conclusion that in at least certain noisy environments, the filterbank domain may be more appropriate. The implication for standard decoding is thus that while MFCC features are superior under clean conditions, there does come a point where the filterbank domain is preferable.

Applying optimal UO decoding, however, it can be observed that the MFCC domain again becomes the more desirable. Since UO decoding during non-distorted speech simplifies to just standard decoding, the implication is now that across all conditions, the best results will be obtained with MFCC feature extraction. The unfortunate news now is that optimal UO decoding for additive distortions in the filterbank domain (and thus distortions similar to additive background noise to the audio waveform) requires a full covariance observation PDF, and the  $24.5\times$  increase in computations that accompanies this feature PDF representation.

The final observation from these experiments, then, is that there is only a small difference between MFCC domain full covariance feature PDFs and the diagonal covariance PDF simplification. Thus the final implication of this experiment is thus that even for distortions that are additive in the filterbank log energy domain, if UO decoding is performed, MFCC domain feature extraction will give better results than using filterbank energies directly. Further, when implementing MFCC domain UO decoding, observation PDFs can be simplified to have diagonal covariance matrices, with only a small negative impact on overall recognition performance.

### 3.8 Comparison to Related Work

There are three branches of published research similar to the work of this thesis. First is the topic often referred to in the literature as the “Missing Data” problem. Second, is a special case of UO decoding applied to the speaker identification task, published under the title “Weighted Viterbi Decoding”. More closely related is a recently published method for including feature variance information called “Uncertainty Decoding”. While each of these techniques is similar to that described in this chapter, differences exist, and will be examined in detail in this section.

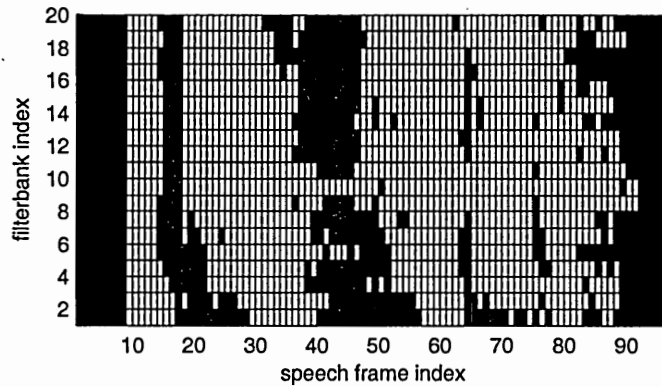
#### 3.8.1 The Missing Data Problem

A standard problem of vision systems in image processing is how to handle recognition of objects that are partially occluded by other objects. This is referred to as the Missing Data problem. A number of researchers have applied the techniques developed in image processing to noise robust speech recognition [8, 29, 33]. Many researchers have reported that if some regions in the spectral domain are completely removed, it is still possible to correctly recognize speech [39, 59], in essence viewing only “stripes” in time across the time-frequency plane.

The framework used by missing data research has been to generate a tiling in the time-frequency plane and heuristically determine whether a tile has reliable information or not. This generates a mask, as depicted in Figure 21. The state output probability is then broken into separate calculations for the reliable and unreliable data components.

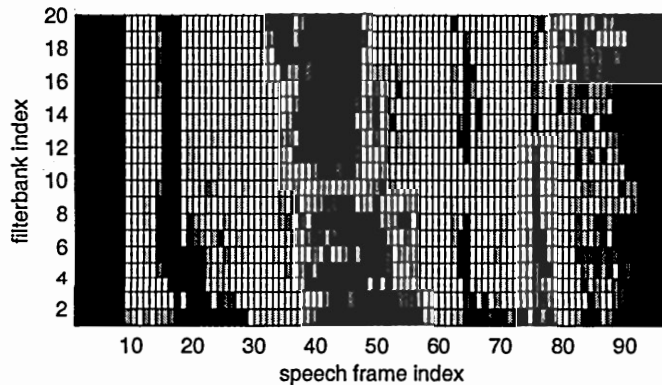
The drawback of this method is obvious. In vision systems, objects causing occlusion are almost always opaque, causing a complete loss of visual information behind them. In auditory systems, signals combine additively, thus the hard decisions pictured in Figure 21 will end up including some tiles that include unreliable information, and throw out other tiles that contains at least *some* information. The result is that unreliable information is still included, and other potentially valuable information is discarded.

In order to address this shortcoming, missing data techniques have been extended by



**Figure 21:** Example of a reliability mask used for Missing Feature robust ASR

several researchers [3, 6, 28, 42, 52] to allow for soft decisions in the reliability mask. This results in using “fuzzy masks,” as shown in Figure 22. Each tile is given a weight between 0.0 and 1.0, similar to a probability that the point is dominated by the speech signal, rather than the background noise, however the weightings are found by trial and error using a number of different weighting functions.



**Figure 22:** Example of a fuzzy mask used for Missing Feature robust ASR

Soft decisions at the mask level bring this branch of research quite close to the Uncertain Observation algorithm. In fact, it can be shown that many of the missing data techniques can be shown as special cases of UO decoding. However, these techniques lack the strong probabilistic framework that is the strength of the research for this thesis.

### 3.8.2 Stochastic Weighted Viterbi Decoding

A special case of UO decoding was published under the name *Stochastic Weighted Viterbi Decoding* in 2002 [61], although it was applied to speaker identification in noise, rather than speech recognition. This work approached the problem of robust recognition by beginning with the standard state output probability calculation given in Equation 9, expands the diagonal covariance normal r.v. evaluation in terms of HMM parameters, and takes an expected value, giving  $E[b_j(\mathbf{x}_t)]$ . The result is dependent upon the mean and variance of the observation vector, and gives the same result as the special case of the UO formulation when observations are modeled by single mixture, diagonal covariance Gaussian PDFs.

The drawback of this work is that it lacks the concept present in the integral formulation of UO Decoding in Equation 15. The weighted Viterbi solution is presented directly for the special case, and implicitly assumes diagonal covariance normal r.v.'s. There is no conceptual description from which the more general UO form to be inferred. Thus, for single mixture diagonal Gaussian observation PDFs, Stochastic Weighted Viterbi Decoding and UO Decoding are identical. However, UO Decoding describes a more general solution that allows complex, arbitrary observation PDFs.

### 3.8.3 Uncertainty Decoding

Another set of work similar to this thesis was recently published at *ICASSP 2002* [13, 38], named by its authors *Uncertainty Decoding*. In this work, the joint PDF  $\Pr[\mathbf{x}, y|q_t=j, \mathcal{W}_t]$  is marginalized over all possible clean observation vectors,  $\mathbf{x}_t$ :

$$\Pr[y_t|q_t=j, \mathcal{W}_t] = \int_{-\infty}^{\infty} p(y_t|\mathbf{x}, \mathcal{W}_t)p(\mathbf{x}|q_t=j)d\mathbf{x}. \quad (69)$$

This is used in place of Equation 15. Similar to UO decoding described in this thesis, the second term in the integral,  $p(\mathbf{x}|q_t=j)$ , is the standard HMM state output probability calculation. Different, however, is the first term, with Uncertainty Decoding using  $p(y_t|\mathbf{x}, \mathcal{W}_t)$  instead of  $p(\mathbf{x}|y_t, \mathcal{W}_t)$ . This can be found using Bayes' rule:

$$p(y_t|\mathbf{x}, \mathcal{W}_t) = \frac{p(\mathbf{x}|y_t, \mathcal{W}_t)p(y_t|\mathcal{W}_t)}{p(\mathbf{x}|\mathcal{W}_t)}. \quad (70)$$

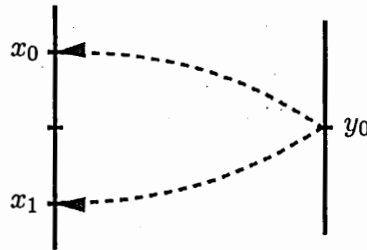
The terms of the numerator are easily found, leaving the prior  $p(\mathbf{x}|\mathcal{W}_t)$ . The approach taken in [13] is to model  $p(\mathbf{x}, \mathbf{y}|\mathcal{W}_t)$  as a mixture Gaussian, and use

$$p(\mathbf{x}|\mathcal{W}_t) = \int_{-\infty}^{\infty} p(\mathbf{x}, \mathbf{y}|\mathcal{W}_t) d\mathbf{y} \quad (71)$$

which can be carried out in a straightforward manner, giving a Gaussian prior.

Although conceptually the UO decoding and Uncertainty Decoding are quite different, the resulting implementations are similar. Both are perfectly valid interpretations, and are capable of handling time-varying mismatched conditions. The findings by the authors of Uncertainty Decoding are consistent with those of this research, showing significant recognition improvement.

An interesting comparison between UO decoding and Uncertainty Decoding is to consider the example below in Figure 23. Suppose some distortion exists so that clean obser-



**Figure 23:** Degenerate example of Uncertainty Decoding.

vations  $x_0$  and  $x_1$  both get mapped to distorted observation  $y_0$ . In this case, we can write the following probabilities:

$$p(x_0|y_0) = 1/2$$

$$p(x_1|y_0) = 1/2$$

$$p(y_0|x_0) = 1$$

$$p(y_0|x_1) = 1$$

Upon observation of the distorted feature vector  $y_0$ , expanding the UO and Uncertainty Decoding equations yields an interesting result. Substituting into the state output probability

for UO decoding will give:

$$\int_{-\infty}^{\infty} p(x|y_0)p(x)dx = p(x_0|y_0)p(x_0) + p(x_1|y_0)p(x_1) \quad (72)$$

$$= \frac{1}{2}p(x_0) + \frac{1}{2}p(x_1) \quad (73)$$

$$= \frac{1}{2}(p(x_0) + p(x_1)). \quad (74)$$

Uncertainty Decoding, however, gives a rather different answer:

$$\int_{-\infty}^{\infty} p(y_0|x)p(x)dx = p(y_0|x_0)p(x_0) + p(y_0|x_1)p(x_1) \quad (75)$$

$$= 1 \cdot p(x_0) + 1 \cdot p(x_1) \quad (76)$$

$$= p(x_0) + p(x_1) \quad (77)$$

Both methods are valid interpretations, with UO giving the expected value of the state output probability conditioned on the distortion and Uncertainty Decoding being the joint PDF of the clean and noisy observations marginalized over all clean observations. However, for this degenerate case, UO decoding gives an answer that matches better with what one would intuitively expect.

### 3.9 Summary and Discussion

This chapter has presented a new method for performing environmentally roust speech recognition using *Uncertain Observations*. The basis of the method is the idea that in many distortive environments, there is actually more information available from the audio signal than just the extracted feature vectors. In particular, information about which observation vectors are more reliable than others, and which dimensions of a particular vector are more reliable than others. For example, speech observed during stationary noise will show regions during high energy vowels that have high signal to noise ratio, and are very similar to speech vectors observed without the interfering noise present. Low energy speech, however, would be completely obscured by the same noise level. Standard HMM decoding weights observations from both of these scenarios equally. The method of this thesis accounts for the varying amount of uncertainty of each feature vector during decoding on a frame by frame basis.

The basic requirement in order to implement UO techniques during speech recognition is to extend feature extraction to generate a PDF over the feature space giving the likelihood that any particular point was generated, given the current knowledge of the distortive environment. Thus whereas standard speech recognition extracts a set of feature vectors, one for each frame, UO decoding requires a set of feature PDFs, one for each frame. This effectively allows for a time-frequency local specification of the reliability of the observed data.

This chapter showed how this feature PDF sequence can then be used in a strong probabilistic framework during decoding. A general form for this decoding framework was shown for both discrete and continuous random variable feature PDF representations, and was shown as calculating the expected value of the state output probability of an observation conditioned on the noisy observation and the current distortive environment. Further, the technique was shown to be the Bayes probabilistic classifier given the feature uncertainty. Thus decoding is the optimal MAP classifier, but now conditioned on the new knowledge of the feature observation PDFs.

To show the feasibility of UO applied to speech decoding, an Oracle experiment was developed, with noise of known mean and variance added to the feature vectors (MFCCs in this case, although the choice was arbitrary). Thus in this example, although the original clean signal was not known, clean feature PDFs are exactly known. During mismatched conditions, standard decoding shows dramatically reduced recognition performance. UO decoding, however, shows recognition performance only slightly lower than the clean, matched condition case.

This method has several advantages over other state of the art techniques in the literature. Feature transformation techniques, for example, can be seen as a special case of UO, that models observation PDFs as a Dirac delta function at the expected value of the clean speech vector. Further information, such as the variance of the Gaussian random variable, will only serve to improve the results, so long as that information is accurate. Model compensation techniques allow for variance updates, however, for computational reasons they are impractical for real-world situations where the distortion, such as background noise, is

time varying.

There are several caveats to note before applying UO techniques to speech recognition decoding. First, there is a continuous computational burden introduced by UO decoding. For the case of a single Gaussian observation PDF, the extra computations are solely those required to generate new covariance matrices and determinants. This step is required for each frame. It was shown that for large vocabulary speech recognition, this amounts to a 15% continuous burden. For a GMM feature PDF, the computations required for Gaussian evaluations scales linearly with the number of mixtures.

A second caveat is that the PDF models need to be accurate in order for UO decoding to improve performance. For example, using bad variance information can lead to worse performance than simply using feature transformation. With reasonable care, however, significant speech recognition accuracy improvements are possible.

## CHAPTER IV

### UO DECODING OF NOISY SPEECH

The new speech recognition decoding algorithm presented in Chapter 3, *Uncertain Observation Decoding*, describes a general technique for robust recognition during time varying distortions. The technique makes no assumptions about the type of distortion, only that some feature extraction algorithm exists that estimates PDF representations of the underlying clean speech, conditioned on some knowledge of the current distortion. This chapter will consider in detail the application of the UO decoding algorithm described in Chapter 3 to the specific problem of robust recognition in additive background noise environments, with the goal of showing that any general feature transformation algorithm can be extended to use the UO framework to improve recognition performance. Examples will be given that show that feature transformation plus UO decoding (FT+UO) roughly equals the performance of model compensation techniques, with the added benefit that FT+UO can be implemented in a time varying manner, unlike model compensation.

The order of discussion in this chapter is as follows: first, an analysis of additive background noise is presented. This is followed by a method for estimating feature variances which is applicable to any feature transformation algorithm. Experimental results are shown for several databases. After this, a second method for estimating feature variance is shown by extending a recently developed Monte Carlo speech enhancement algorithm. Several possible extensions to this algorithm are also proposed. Finally, the extension of PDF feature extraction to first and second derivatives, and the difficulties involved, is presented, along with a summary of the findings of this chapter.

#### *4.1 Analysis of Feature Extraction in Additive Noise*

Speech observed in an additive noise environment has some interesting consequences. In particular is that clean speech itself, being only pseudo-stationary, covers a large dynamic

range across both time and frequency. When even stationary noise is added, the local SNR of the result can be expected, then, to have a wide range of local SNRs in the time-frequency domain. Thus feature extraction techniques based upon time-frequency local energy, such as the windowed filterbank used in MFCC computation, will exhibit a high accuracy in some areas of the time-frequency plane, and lower accuracy in others. Consider, for example, the plots of the log energy feature shown in Figure 24. This is one of the standard 13 MFCC vector components, plotted across time for several signals: black is the energy of the original clean speech signal, dark gray is the energy in a white stationary noise waveform, and light gray is the energy of the sum of these two waveforms. The local SNR across time for this feature is the difference between the clean and noise energies, which is easily seen to change rapidly from well below zero during softer portions of speech, such as between words, to well above +20dB during voiced regions of the signal. And all this happens several times in a second and a half. An obvious consequence is that this speech feature

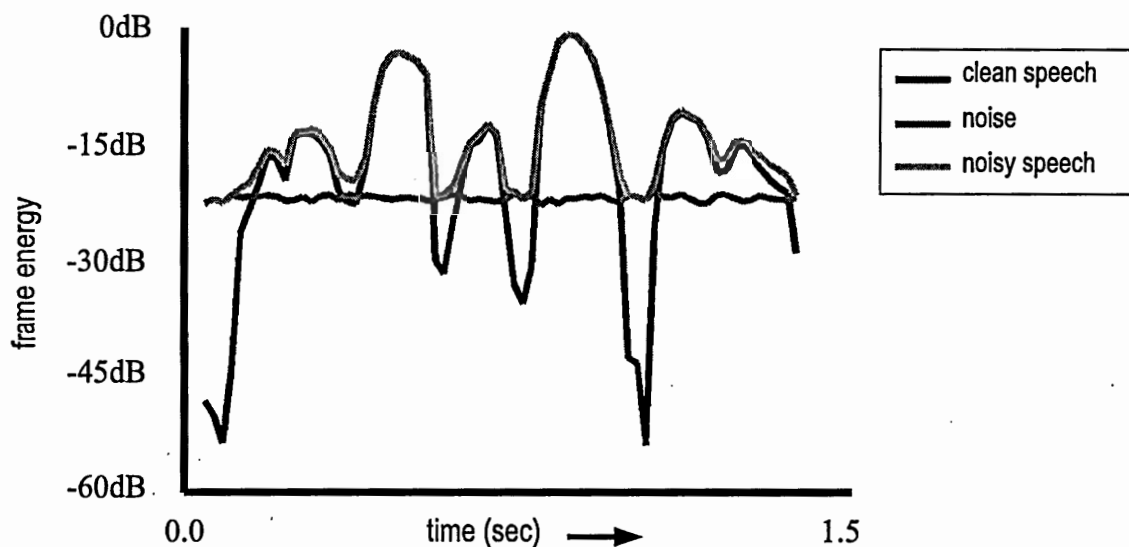


Figure 24: Plot of the log energy speech feature across time for clean speech, a noise signal, and the sum of these two, showing the rapid change possible in speech feature reliability.

shows rapid fluctuations in reliability, despite a standard speech recognition decoder treating all observation frames equivalently.

A similar plot can be drawn looking across frequency at any instant in time. Consider the plots in Figure 25 of the smoothed LPC spectrum for one particular frame extracted from the signal used in the previous figure. Black is from the clean waveform, dark gray is from the noise waveform, and light gray is from the sum. Now it can be seen that overall frame energy is not enough to decide whether information is reliable or not, as even though the overall SNR for this frame is relatively low, there exist frequency bands with SNR high enough for near-perfect feature extraction.

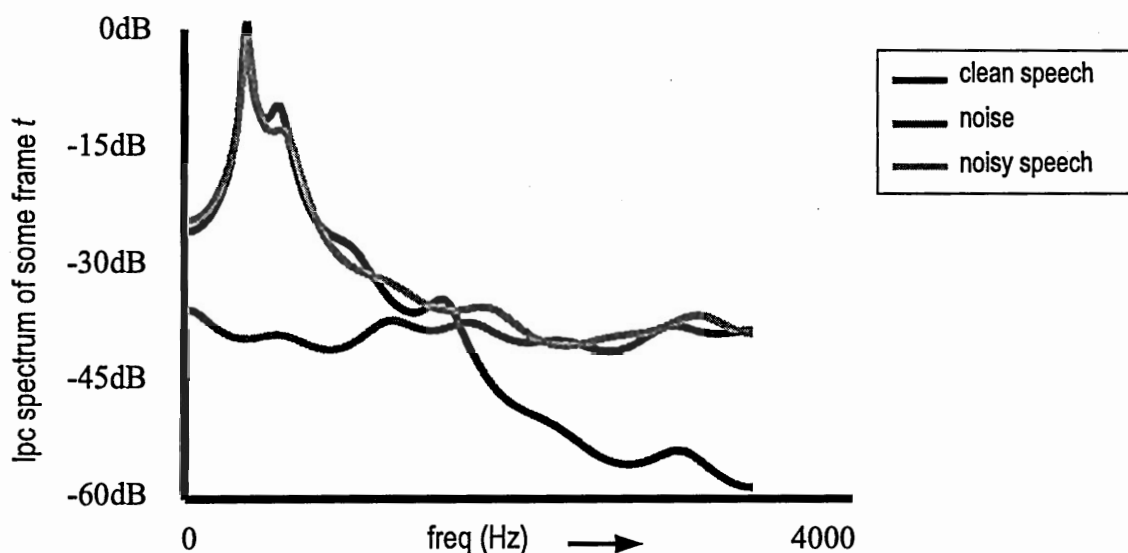
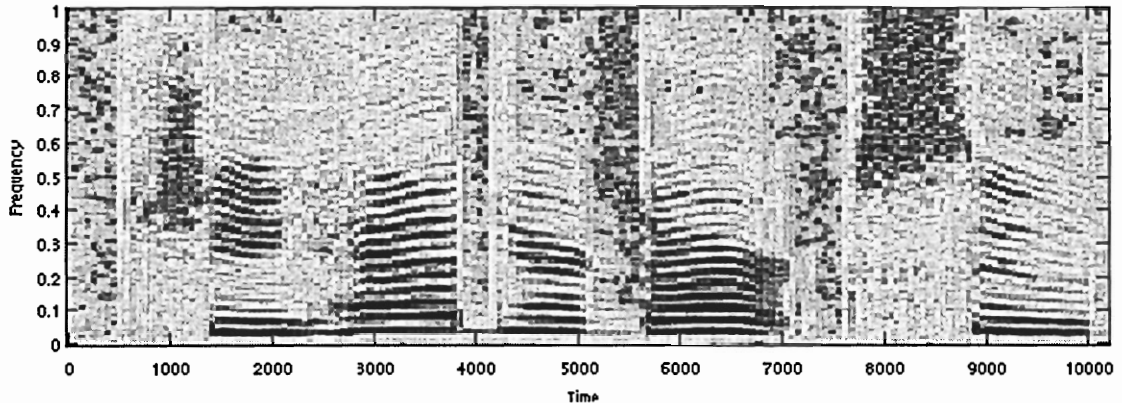


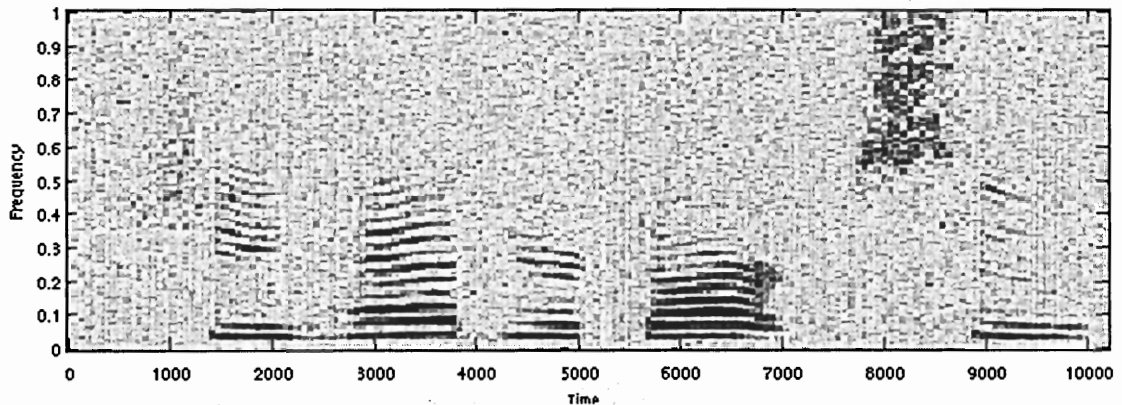
Figure 25: Plot of the LPC smoothed spectrum across time for clean speech, a noise signal, and the sum of these two, showing the rapid change possible in speech feature reliability.

Put together, the rapidly changing local reliability across time and frequency gives frame-based filterbank energy features a wide range of local SNR, as shown by the spectrograms of Figure 26. The clean speech signal, shown in part (a), has had F-16 cockpit noise obtained from the Noisex database [57] added with an overall signal to noise ratio of 0dB. The speech plus noise signal is shown in part (b). Even in these extreme conditions, some locations in the time frequency plane are unaffected, such as the pitch tracks during voiced regions. It is fairly obvious from this comparison which filterbank energies will be most reliable. The transformation from filterbank energies to MFCC features adds a

level of obfuscation to the local SNRs, however it can still be expected that the various cep coefficients will be affected by varying amounts in a predictable way based upon the filterbank time-frequency local SNRs.



(a)



(b)

Figure 26: Time-Frequency comparison of clean and noisy speech.

#### 4.1.1 Nonstationary background noise estimation

A problem of speech recognition in additive background noise is that it is difficult to extract the speech signal reliably from the noise. The other side of this problem is that it can be difficult to extract the characteristics of the noise reliably because of the interfering speech signal, and each of the speech feature PDF extraction methods described later in this section require knowledge of the background noise characteristics. For stationary

noise, simple techniques can be used, such as gathering noise statistics during non-speech, which is relatively easy, since even “continuous” speech has plenty of gaps and very low energy events. Many authors have recently published methods for nonstationary noise, based upon continuous gathering of statistics, with a decay to allow older information to be discarded [15, 19, 21, 40, 56]. Thus noise estimates can be updated even during voiced segments. A better solution is an iterative method involving specific modeling of the interfering speech [11]. This would entail a speech recognition pass allowing time-varying noise estimates can be generated, so that noise robust speech recognition can be run again, so that new noise estimates can be formed, etc.

The work in this thesis is devoted to robust recognition algorithms, and it is considered prerequisite that effective noise estimation algorithms already exist. Two strategies are thus used in the experiments that follow later in this chapter. The first method, used for the experiments that test stationary background noise, is to use the first forty frames of each noisy utterance, assume they are prior to any speech has begun, and average over them to find noise mean and variance in each filterbank. The second method, used for experiments with time-varying noise, directly estimates noise statistics from the noise signal itself, which is only available because stereo clean/noisy databases are being used.

## *4.2 Extending feature transformation to PDF feature extraction*

In order to implement a UO solution for robust recognition of noisy data, the knowledge of the noisy observed signal, and the effect of the estimated noise shown in the previous section now must be converted into a PDF representing each frame of speech in the feature vector domain.

In this section, a novel technique for extending any filterbank domain feature transformation method to give single mixture, diagonal covariance Gaussian PDF features for UO decoding is presented.

Many recent methods for robust speech recognition in additive noise environments operate in the filterbank log energy domain [17, 64]. Using the same basic principle as spectral

subtraction, these methods have shown improved performance with the following steps:

- estimate the noise log energy  $w_{t,k}$  in each filterbank  $k$  of the  $t$  frame of speech, as described in the previous section,
- observe  $y_{t,k}$ , the log energy of the  $t^{\text{th}}$  frame of noisy speech signal in each filterbank  $k$ , and
- estimate the log energy of the clean speech signal  $x_{t,k}$  in each filterbank based upon some mapping  $x_{t,k} = f(y_{t,k}, w_{t,k})$ .

Many different functions could be used for  $f(\cdot)$ , but whatever form is chosen, the parameters can be determined using a stereo clean/noisy training dataset.

Up to this point, the algorithm implemented is nothing more than feature transformation. This is extended here to also estimate the variances for each filterbank. The result is a single mixture, diagonal covariance normal PDF representation of each frame of speech, which can now be used directly in a UO decoder. While this formulation is in the filterbank domain, the transformation to the MFCC domain is linear, allowing the mean and variance to be computed in the MFCC domain, if desired. For this section, only static filterbank features were used, however.

Feature enhancement in the noise environment depicted in Figure 26 would expect that little change would be necessary for features representing areas of the time-frequency plane that have high SNR, while areas of low SNR would require a larger amount of change. For example, consider the difference between LPC smoothed spectral representations of a clean frame, to a noisy version of that same frame. At the first and second format locations, the local SNR is high enough that the observed energy is extremely close to the energy of the clean speech signal. This is, however, not the case for other filterbank outputs for this frame. Spectral subtraction removes a significant amount of energy at these other locations. It is thus reasonable that the expected error after the speech enhancement algorithm in a given region of the time-frequency plane is directly related to the amount of energy removed, i.e., the more energy removed in a particular filterbank, the more likely the output is wrong. If  $y_{t,k}$  is the observed energy in the  $k^{\text{th}}$  filterbank of frame  $t$ , and  $\hat{x}_{t,k}$  is the same energy

after running the speech enhancement algorithm, this gives something along the lines of Equation 78:

$$E[x_{t,k} - \hat{x}_{t,k}] \sim y_{t,k} - \hat{x}_{t,k}. \quad (78)$$

After experimental testing, the method chosen was to model the above relationship as an independent second order polynomial for each filterbank  $k$ :

$$\hat{\sigma}_{t,k} = b_{0,k} + b_{1,k}z_{t,k} + b_{2,k}z_{t,k}^2 \quad (79)$$

where  $z_{t,k}$  is a post-feature transformation estimate of the local signal to noise ratio of each filterbank  $k$  in each frame  $t$ ,

$$z_{t,k} = y_{t,k} - \hat{x}_{t,k} \quad (80)$$

For the implementation in this thesis, the coefficients  $b$  of these polynomials were found using stereo training data generated by adding white noise to the TIMIT database. Data was added at several signal to noise ratios, generating a very large set of clean frame, noisy frame pairs. This large set was then used to solve for the  $b$ 's in the standard MMSE way.

To further improve this estimator, a three state classifier was used to separate the incoming noisy speech frames into three categories: known speech, known non-speech, and unknown. Known speech frames were identified by their energy relative to the overall energy of the signal. Unknown frames were identified by those with lower relative energy, and also the lack of being near to known speech frames. Those frames that were close to known speech frames, but didn't have high energy were classified as unknown. Three separate mapping polynomials as in Equation 79 were then trained for each of these three cases. The motivation for this extra complexity was that it was noted that nonspeech frames were being handled very poorly.

#### 4.2.1 Application to Stationary Noise

To show the performance of this variance estimation method, an experiment was created to compare feature transformation, feature transformation plus UO decoding, and model compensation. Stationary white noise was added to the TIDIGITS database with a signal to noise ratio of 3dB, which corresponded to a peak SNR of 10dB. The experiments used

only static filterbank features, and in matched clean conditions this gives a 2.1% error rate. Matched noisy conditions results in an error rate of 12.6%. Mismatched conditions showed a dramatic drop, with an error rate jumping to 84.1%.

Since the noise added for these experiments is stationary, model compensation is a valid technique. The method chosen was the log-add implementation of the Parallel Model Combination algorithm. Two forms were implemented. First, was the full PMC which updates both means and variances of each HMM in the full model set. The second implementation was to approximate feature transformation by only updating the model means, and not the variances. These two techniques resulted in word error rates of 20.5% and 35.6%, respectively.

To implement feature transformation, a filterbank log energy mapping algorithm was implemented. This basic technique improved the error rate to be similar to the mean-only version of PMC, at 34.8%. The final experiment was to extend this version of feature transformation with the variance estimation technique described in this section. The resulting single mixture, diagonal covariance Gaussian features were then used in UO decoding. The results of feature transformation plus UO decoding dropped the word error rate to 21.6%.

**Table 7:** Digit recognition results for UO decoding in stationary additive noise.

| Conditions     | Decoding Method                | Error Rate |
|----------------|--------------------------------|------------|
| matched, clean | standard                       | 2.1%       |
| matched, noisy | standard                       | 12.6%      |
| mismatched     | standard                       | 84.1%      |
| mismatched     | PMC (means only)               | 35.6%      |
| mismatched     | PMC (full)                     | 20.5%      |
| mismatched     | feature transformation         | 34.8%      |
| mismatched     | feature transformation plus UO | 21.6%      |

This set of experiments has thus shown, by comparing performance in a stationary white noise environments, the major goal of this chapter: that feature transformation techniques extended to handle variance information via the UO algorithm perform similar to full model compensation techniques. However, while model compensation cannot be implemented in

a time-varying manner, UO decoding allows the background noise model to change, even as often as every frame.

### ***4.3 Direct estimation of observation PDFs***

The previous section described a generic method for estimating feature vector variances that can be used to improve the performance of any feature transformation technique. This section deals with more specific methods of finding feature PDFs that are tied to particular speech enhancement algorithms.

Two methods are discussed in this section. The first is an extension of a recently developed Monte Carlo method used for speech enhancement. The second is an extension of the Ephraim-Malah speech enhancement algorithm.

#### **4.3.1 Markov Chain Monte-Carlo (MCMC) Enhancement**

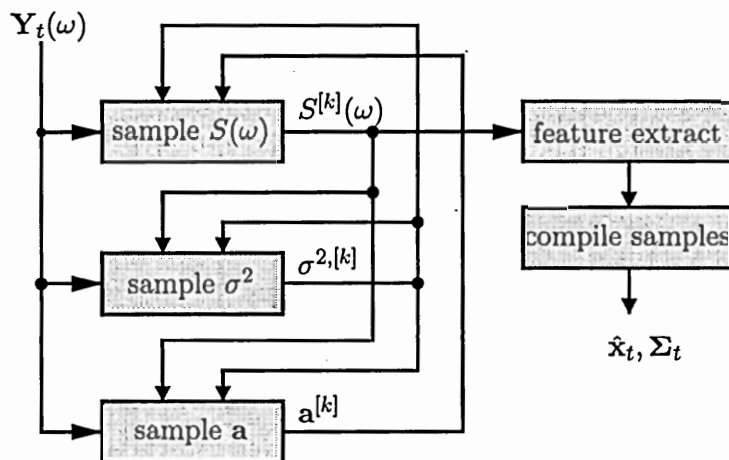
With the advances in computing power that have occurred in the past decade, many algorithms once considered inappropriate or wasteful are becoming viable techniques. One class of methods that have recently begun to be considered in a more favorable light are Monte-Carlo methods. That is, methods in which a large number of trials are extracted from a random system in order to characterize the performance of the system, rather than the classical method of finding parameterized methods of characterization.

Three methods of Markov Chain Monte-Carlo (MCMC) speech enhancement have recently been proposed. A time domain method was presented in [20] for the purpose of listening and was not used for this thesis. More recently, two frequency domain methods were developed at Georgia Tech by Robert Morris, and presented in [43]. These methods showed improved recognition performance in both an unconstrained, non-parameterized case (MCMC-NP), and further improvement by assuming an underlying autoregressive speech model (MCMC-AR). A brief look at the MCMC-AR method will now be given, followed by its application to UO decoding. A more detailed description of this technique is available in [43].

The framework for the AR frequency domain method is the Gibbs Sampler shown in Figure 27. The system operates independently on each frame  $t$ , generating outputs randomly

drawn from the posterior  $S(\omega)|\mathbf{a}, \sigma, Y_t(\omega)$  with each of many iterations. After a large number of these samples have been collected, and the initial samples while the system was converging discarded, a standard speech feature enhancement algorithm can then be implemented by running whatever feature extraction method is desired on all of the randomly generated frames of speech, finding the expected value of this set feature vectors, and using this as the enhanced speech feature vector,  $\hat{\mathbf{x}}_t$ . At this point, MCMC-AR robust recognition is another form of feature transformation.

An extension explored as a part of the work for this thesis has been presented in [43] that allows MCMC-AR to be applied to UO decoding. This extension involved finding second order statistics of the collected set, generating single mixture, diagonal covariance Gaussian PDF features appropriate for UO decoding. Accumulation of these statistics is only possible because of the Monte-Carlo nature of the underlying MCMC-AR speech enhancer, because it makes available a large number of possible clean speech frames, rather than giving a hard-decision single output, as in other traditional speech feature transformation algorithms.



**Figure 27:** Block diagram of the Gibbs Sampler to implement the autoregressive constrained Markov Chain Monte-Carlo speech enhancement algorithm.

The MCMC-AR method, extended to UO, was tested on two databases. First was a test using additive white, stationary noise added to the TIMIT database with a peak SNR of 10dB, corresponding to a standard signal to noise ratio of 3dB. The feature set used was that standard 13 dimension static features, cep[1] through cep[12] plus log energy. Only

static features were used.

Recognition results, summarized in Table 8, show that matched clean performance gives an phoneme error rate of 40.4%, with matched noisy performance at 54.6%. Training on clean and testing on noisy speech increased the error rate to 86.6%.

The MCMC-AR speech enhancement algorithm was implemented with a burn-in run of 500 iteration for each frame, followed by a further 2,000 iterations that each generated a hypothesized clean speech feature vector. Using just the mean of this set gives a baseline performance of MCMC-AR as a feature transformation algorithm. This dropped the the error rate to 81.6% in the mismatched case. Compiling both first and second order statistics, in order to feed diagonal covariance Gaussians to a UO decoder dropped the error rate further, down to 78.6%. In order to observe the limits of UO decoding for this case, the “Ideal Variance” analysis was then performed, replacing the MCMC-AR observed variance with  $\Sigma_t = \text{diag}((y_t - \hat{x}_t)^2)$ . This showed that if the difference were exactly known, UO decoding could in the limit obtain an error rate of 75.4%.

**Table 8:** Phoneme error rates for UO decoding in stationary additive noise using the MCMC-AR feature transformation algorithm.

| Conditions     | Decoding Method                | Error Rate |
|----------------|--------------------------------|------------|
| matched, clean | standard                       | 40.4%      |
| matched, noisy | standard                       | 54.6%      |
| mismatched     | standard                       | 86.6%      |
| mismatched     | feature transformation         | 81.6%      |
| mismatched     | feature transformation plus UO | 78.6%      |
| mismatched     | ideal variance UO              | 75.4%      |

To test the MCMC-AR/UO combination in nonstationary noise, the Aurora2 database was used. For background noise estimates, as described in Section 4.1.1, it was assumed that a good algorithm already exists. Thus, noise statistics were gathered by observing the unknown noise signal itself, and collecting statistics over a twenty frame range around the current observation location.

Figure 9 shows results across all SNR levels in the Aurora2 database. Important to

note is that as expected, in nearly clean environments, UO decoding does not degrade performance at all. In the high-noisy -5dB environment, MCMC-AR feature transformation reduced the error rate due to noise by 26.3%, from 92.2% to 65.9%. UO decoding reduced the error a further 3.8%, to 62.1%. Ideal Variance analysis showed that the limit for UO decoding in this case was an error rate of 41.9%.

**Table 9:** Phoneme error rates for UO decoding in stationary additive noise using the MCMC-AR feature transformation algorithm.

| Decoding Method | PER $\infty$ dB | PER 20dB | PER 15dB | PER 10dB | PER 5dB | PER 0dB | PER -5dB |
|-----------------|-----------------|----------|----------|----------|---------|---------|----------|
| standard        | 3.1             | 6.0      | 15.7     | 36.6     | 64.3    | 84.7    | 92.2     |
| FT              | 3.1             | 4.4      | 6.1      | 9.7      | 17.9    | 37.6    | 65.9     |
| FT+UO           | 3.1             | 4.3      | 5.8      | 9.0      | 16.4    | 34.3    | 62.1     |
| ideal UO        | 3.1             | 4.1      | 4.7      | 6.3      | 9.4     | 18.8    | 41.9     |

The computational cost of the MCMC-AR method as described is quite large. The implementation used in [43] required two to six thousand iterations through the Gibbs Sampler of Figure 27, with each iteration being larger than standard ASR feature extraction. The result was a front-end that had a computational burden around two orders of magnitude greater than the entire standard HMM decoding process. Further, the Gibbs Sampler is iterative, and thus not easily amenable to parallelization. Regardless, Moore’s Law allows the exploration of such techniques, as it will not be long before MCMC-AR is a viable front-end feature extractor for real time systems.

#### 4.3.2 MCMC Non-Gaussian methods

In this thesis, it is proposed that the previous method can further be extended to allow for non single-mixture diagonal covariance Gaussian representations of the clean speech PDF. Examples of such PDFs would be full covariance Gaussians, multi-mixture Gaussian Mixture Models, Uniform, or other standard random variable representations that better model the uncertainty of a frame of noisy speech. A further, extremely inefficient algorithm is to find the state output probability in the standard manner for each of the  $K$  iteration

outputs from the Gibbs Sampler. The overall state output probability can then be computed as the expected value of this set. In terms of UO decoding, this is modeling the observation PDF as  $K$  delta functions of equal height. While very inefficient, this method does allow for arbitrary PDF representations.

The most reasonable of these proposed methods is the mixture Gaussian observation PDF. This method does give an extra computation burden to the decoder, but it has a distinct advantage. In the filterbank log energy domain, used in Section 4.2, it is somewhat reasonable to make the assumption that the clean speech feature PDF can be effectively represented by a Gaussian with a diagonal covariance matrix. In the cepstral domain, however, this assumption is no longer valid. Rather than moving to a full covariance matrix, a diagonal covariance GMM model of the collected samples allows modeling the feature PDF by multiple mixtures, allowing a more compact representation, while still retaining the modeling benefits available due to the Monte-Carlo method. Further, the GMM model is superior when the filterbank log-energy assumption of independence is invalid. Due to the computational resources required to implement these and other ideas mentioned in this subsection, testing experiments have not been run, and have been left for future work.

### 4.3.3 Other techniques

Several other speech enhancement techniques can be extended to generate PDF features. During the course of this research, one such method attempted was to extend the Ephraim-Malah minimum mean square error log spectral estimator [16] to produce second order statistics. The basic Ephraim-Malah system finds the expected value of the clean speech log spectrum for each Fourier transform index independently, and can be extended, albeit in a somewhat complex manner, to also generate the variance of each log FFT bin. An inherent problem with this technique is the assumption of independence between all Fourier transform indices. While this is not a major stumbling block for standard speech enhancement, MFCC methods require the summation of many adjacent indices. At this level, then, the index dependence effectively renders this technique unusable. Similar results were found when developing an FFT-based mapping method akin to Section 4.2 above. In general,

FFT-index based methods were ruled out after this portion of the research.

#### 4.4 *Bursty Noise*

Another interesting problem is additive bursty noise, such as the machine gun waveform shown in Figure 28. The same techniques developed in this chapter for generating observa-

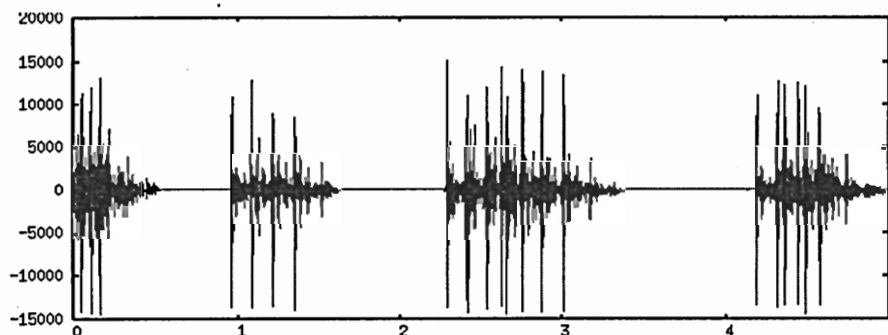


Figure 28: Machine gun bursty noise waveform.

tion PDFs for additive noise can be again applied to this problem. In this bursty scenario, however, many frames will exist that are completely corrupted by noise. This is similar to the special case in Section 3.5.2, where no acoustic information is available. Decoding can thus take advantage of the shortcut of not using acoustic information during bursts, which should be easily detectable. Although experiments were not performed for this type of noise, this is a common DARPA task that can be considered during future work.

#### 4.5 *Summary and Discussion*

This chapter has presented several methods for generating PDF representations of speech observed in additive background noise environments, and given recognition results when incorporating these features into an Uncertain Observation decoder. The methods all have the common dependence upon first finding an accurate estimate of the spectral shape of the background noise.

Two classes of PDF estimators were described. First was a generic algorithm that mapped time-frequency-local SNR estimates into filterbank log energy variances. This

generic method has the benefit that it can be applied to any existing filterbank domain feature transformation method. One drawback, however, is that it is limited in producing only single mixture, diagonal covariance Gaussian representations in the filterbank domain.

The second class of PDF estimators were speech-enhancement-specific methods. Two specific cases were considered. One was an extension of the existing Markov Chain Monte-Carlo MCMC-AR speech enhancement algorithm. This method allowed for arbitrary feature PDFs generated directly in any speech feature extraction domain, improving performance. This method suffers from an extreme computational burden, far greater than that required for recognition itself. For future applications, as Moore's Law continues, this type of feature estimation algorithm may become more practical. A second case of speech enhancement algorithm specific PDF estimation was an extended version of the Ephraim-Malah speech enhancer, but the shortcomings of this method lead to little or no improvement in recognition performance.

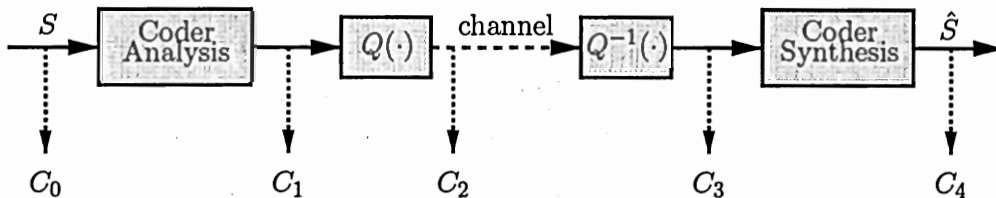
Both classes of these methods were shown to improve recognition accuracy over non-UO systems in a variety of noisy databases, using artificially generated data. Further, the algorithm was shown to work on both continuous digits, and more broadly, on phoneme identification, using noisy TIMIT experiments.

An important observation made was that it is extremely important to have accurate estimates of the background noise in order for the PDF feature extraction methods to properly work. A further observation was that features based upon FFT-indices are not practical, due to correlation between adjacent indices. Further, calculating derivative PDF features as linear combinations of static feature PDFs was not practical. Work in this direction towards direct estimation methods for derivative features is considered a promising field for further study.

## CHAPTER V

### APPLICATION OF OBSERVATION UNCERTAINTY TO DISTRIBUTED SPEECH RECOGNITION

A topic that has generated a lot of recent work is Distributed Speech Recognition, or DSR. Simply, this is the capture of speech at one location, transmitting the signal (or some representation of the signal) across a network, and performing recognition at a remote server. What makes this topic interesting beyond just transferring bits is that network latency, channel artifacts, and compression need to be handled in an intelligent manner by the ASR engine on the server.



**Figure 29:** Diagram of a communication system that includes a speech compression algorithm, with potential access points for ASR labeled (adapted from [32]).

The first consideration is the access point of the ASR server in the overall communication system. Consider the diagram shown in Figure 29. The various locations where feature extraction for ASR could be performed are labeled as  $C_0$  through  $C_4$ .

$C_0$ : Original speech waveform  $S$

$C_1$ : LSF and voicing information, MFCCs, or other coder parameters

$C_2$ : Quantized version of  $C_1$

$C_3$ :  $C_2$  through a non-ideal channel

$C_4$ :  $C_3$  with artificial synthesis (output waveform  $\hat{S}$ )

DSR implies that the recognition server is connected at either the  $C_3$  or  $C_4$  location. Looking at DSR from the point of view of robust speech recognition, there are two new

potential sources of mismatch between the speech models and the observed data: the effects of compression, and effects of the channel.

First, this chapter will consider alleviating the effects of compression. Sections 5.1 and 5.2 look at two aspects of compression. First, is when the datastream is tapped at the  $C_3$  location, giving raw access to vector quantized MFCC, LSF or other similar spectral features. This location is of special interest to both central office recognition systems that have access to the raw bit-stream, and “recognition only” systems that transmit data for recognition but no other listening or archival purpose. The second type of compression considered is when the datastream is not tapped until the  $C_4$  location. This corresponds to compressing speech with an advanced coder such as MELP or G.729, transmitting the bit-stream, and then decompressing the speech such that the recognition engine has access only to the resynthesized waveform. An example of this would be recognition of an incoming cellular telephone call at a call center. The only available data is a waveform that at some stage passed through an aggressive, advanced speech coder.

Many types of speech compression are known to create only minor degradations in terms of human listening intelligibility, yet have more detrimental effects on speech recognition. By modeling compression at the  $C_3$  and  $C_4$  locations as generating feature vectors with inherent uncertainty, the UO decoding algorithm can be applied to improve recognition performance.

The second source of mismatch considered in this chapter is the channel. For the two compression-adaptation methods, the transmission channel is assumed to be ideal, with the bit-stream transmitted perfectly. In Section 5.3, however, it is the compression that is considered ideal, and the channel that creates mismatch, in particular due to dropped packets. Such a scenario is common in cellular or other Voice over IP networks, where due to low delay, retransmit requests for dropped packets are not practical. This section will consider in detail the application of UO decoding to this problem by modeling feature vectors extracted during bursts of lost packets as PDFs in order to denote that while normal features are very well known, in-burst features are unreliable.

## 5.1 MFCC VQ Adaptation

This section now describes using UO techniques to improve recognition performance when feature extraction occurs at the  $C_3$  location of Figure 29.

### 5.1.1 Problem Description

There are two common scenarios in which speech recognition would take place at the  $C_3$  location of Figure 29. The first is if a recognizer is operating at, for example, a cellular central office, and has access to the raw bitstream prior to speech synthesis. It has been shown that recognition using these parameters prior to synthesis can give slightly improved recognition performance [32]. The second scenario is when parameters for speech synthesis might be destined for one location (such as a human on the other end of a phone connection), while a second stream of recognition parameters are destined for a different location (such as a speech server). The recognition stream is independent of the speech coder itself, and may consist of MFCC features extracted from the original waveform. These are then highly compressed using VQ and transmitted to the speech recognition server. Both of these scenarios describe a DSR environment.

One established standard for DSR is the ES-202-050 Advanced Front-End Feature Extraction Algorithm [18] from ETSI, the European Telecommunications Standards Institute. It is a standardized method for extracting and compressing MFCC vectors for telecommunications networks. One shortcoming, however, is its rather lackluster VQ compression of MFCC parameters. The standard, by default, yields a bit-rate of 4.8kb/sec, almost on the order of the parameters required for speech synthesis itself.

For this thesis, the ETSI front end was extended to perform a more aggressive VQ step, reducing the distributed ASR stream bit-rate to 3.3 kb/sec. The trade-off for transmitting this more compact representation of the MFCC vectors is a drop in recognition performance. The next section will describe several ways the UO methodology can be applied to improve recognition performance, gaining back a significant amount of the performance lost due to aggressive MFCC compression.

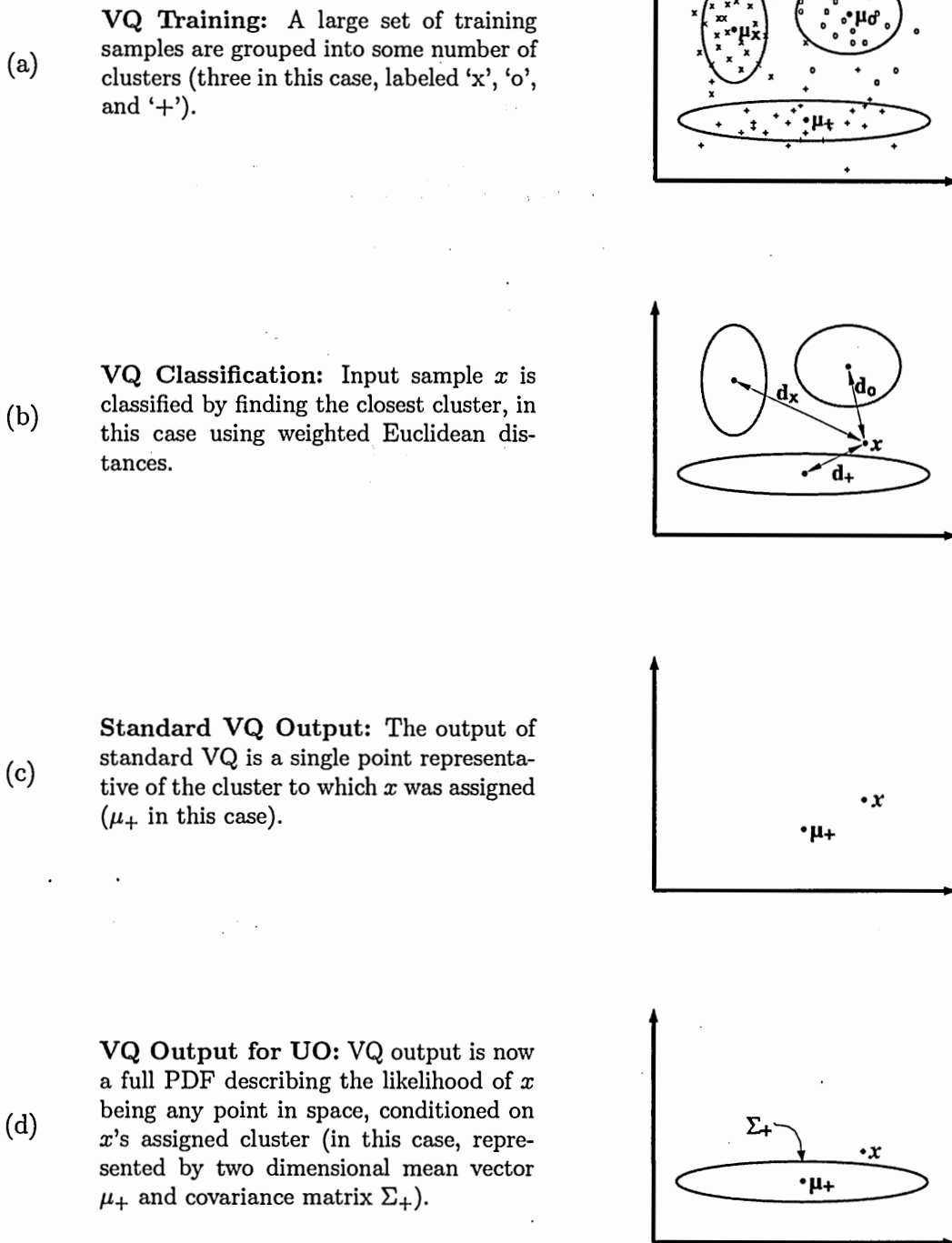
### 5.1.2 Two Proposed Solutions

Compensation of MFCC vector quantization is a rather ideal application of UO decoding. The goal of a VQ system is to reduce the possible feature space down to a small set of  $I$  discrete points. The criteria used is to take a large training set, and find a set of  $I$  clusters, as depicted in Figure 30(a). During compression, an input sample is classified by identifying the nearest of these clusters (Figure 30(b)). Transmission is now of this cluster's index  $i$ , rather than the MFCC vector. The MFCC vector is reconstructed at the receiver by using the mean of the cluster as in Figure 30(c), as this is known to the receiver a priori. These values comprise a *codebook*.

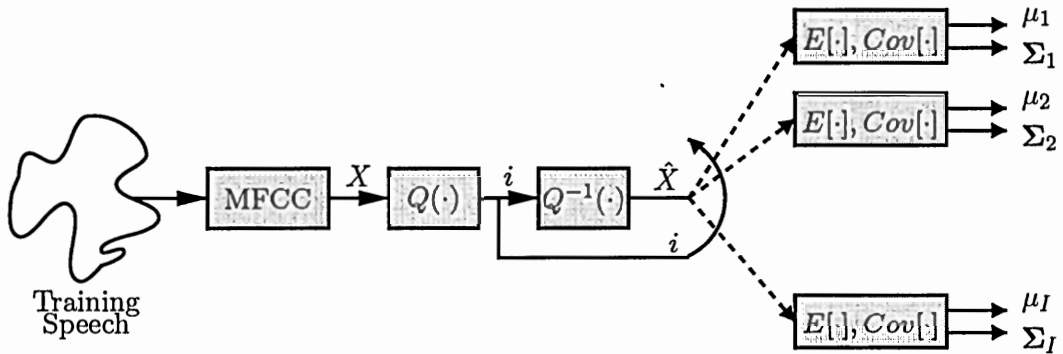
Vector Quantization can be extended, however, when its output is destined for use in a statistical classifier that allow for feature uncertainty. During VQ training, complete statistics about each cluster are known, but normally discarded with only the cluster centroids being retained. The full statistics can, of course, be retained, and embedded into the VQ decoder at the time of its design. Upon receiving a VQ codebook index, the receiver can now output more than just a single point. It can output a full PDF representing the cluster as found in the training data, as depicted in Figure 30(d). Most uses of VQ have no need for this extra information, however it can be used as a compression-robust input to the UO decoder developed in Chapter 3 to improve recognition.

Using this method in UO decoding does have the drawback that these extended cluster statistics must be generated during the VQ codebook design stage. When working with an existing VQ system, however, access to this data will not be available. In this case, an alternate method is to use a large database of samples, observed them both before and after compression, and note the codebook index of each. Statistics can then be gathered for each index. A flowchart depicting this procedure is shown in Figure 31. Whether the statistics are gathered at codebook design time, or after the fact using example data, access to the internals of the VQ system is required because the codebook indices must be known to generate the PDF information.

The resulting system is shown in Figure 32 for a codebook with single mixture Gaussian outputs. Notice from this system that since the variance information is found during the

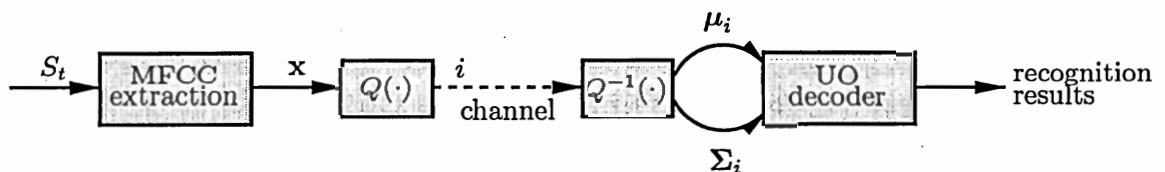


**Figure 30:** Visualization of VQ training, VQ classification, standard VQ output, and VQ-UO output. Training and classification (steps (a) and (b)) are the same for both standard VQ output (shown in (c)) and VQ-UO output (shown in (d)).



**Figure 31:** Training UO parameters for VQ-UO-Internal systems. The set of  $T$  MFCC vectors  $X = \{x_1, \dots, x_T\}$  in a speech database are quantized, and statistics gathered for all vectors belonging to each codebook index  $i$ , resulting in the set of variances  $\Sigma_1, \dots, \Sigma_I$  for the UO decoder.

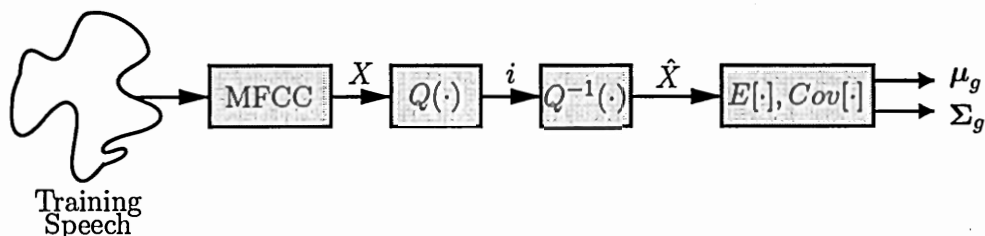
training of the VQ system (just as the codebook entry cluster means), it is assumed known at the receiver, and need not be transmitted. Thus the network bandwidth required for this system is identical to that required for the standard VQ system alone. Further, observation PDFs are found doing nothing more than table lookup. The only cost for the new implementation is simply that required for implementing the UO decoding, determined earlier (see Section 3.6) to be around 67% processing overhead for the state output probability calculations, or roughly a 33% overhead overall for continuous speech recognition. This method will be referred to from here as the  $C_3$ -Internal-UO algorithm.



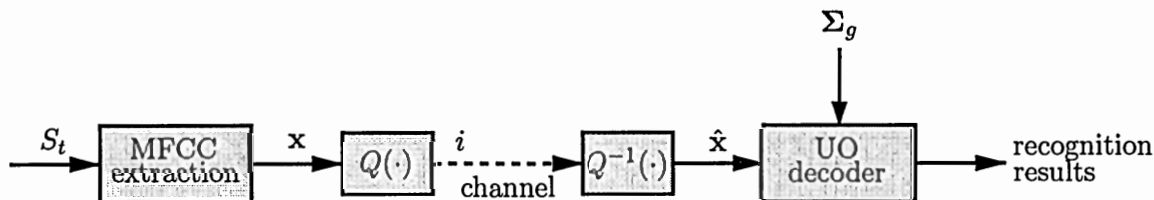
**Figure 32:** Flowchart of the  $C_3$ -Internal-UO system for robust recognition of  $C_3$  features. Each VQ codebook entry has its own variance information  $\Sigma_i$ .

A second, simpler method can also be implemented, and will be referred to as the  $C_3$ -External-UO algorithm. It can be performed as a post- $Q^{-1}$  stage, and thus does not require *any* internal information from the quantization system. This method can be implemented

as a post-processing step even on proprietary distributed ASR systems. Rather than determining a separate PDF for every codebook entry, it can be assumed that the codebook generation process inherently gives each cluster similar statistics. This is a reasonable assumption, as larger clusters with higher variance will typically be the first to be split. Thus, every codebook entry can be assumed not only to be zero mean, but to also all share the same global diagonal covariance matrix. The variance information can then be found by treating the quantization system as a black box, and using a corpus of clean speech as training data. Denoting the unquantized MFCC vectors as  $\mathbf{x}_t$ 's and quantizer output vectors as  $\hat{\mathbf{x}}_t$ 's, the zero-mean assumption can be verified by showing  $E[\hat{\mathbf{x}}_t - \mathbf{x}_t] \approx 0$ , and the global variance information necessary for UO decoding can be found as  $\Sigma_g = E[(\hat{\mathbf{x}}_t - \mathbf{x}_t)^2]$ . Figure 34 shows a flow diagram for implementing the  $C_3$ -External-UO algorithm.



**Figure 33:** Training UO parameters for  $C_3$ -External-UO system. The set of  $T$  MFCC vectors  $X = \{\mathbf{x}_1, \dots, \mathbf{x}_T\}$  in a speech database are quantized, and statistics gathered for all vectors globally, resulting in the single covariance matrix  $\Sigma_g$ . Note that  $\mu_g$  is not necessary.



**Figure 34:** Flowchart of the  $C_3$ -External-UO system for robust recognition of  $C_3$  features. All VQ codebook entries share the same covariance,  $\Sigma_g$ .

For this second method, the cost is reduced even further. Again, as the variance information is kept at the receiver, it does not need to be transmitted, thus no extra network

bandwidth is required. Further, since the observation variance  $\Sigma_g$  is constant for all codebook entries, the UO combined covariance matrix for mixture  $k$  of state  $j$ ,  $\Sigma_\tau = \Sigma_{jk} + \Sigma_g$  is identical for every possible observed feature  $\mathbf{x}_t$ . The  $\Sigma_\tau$  inverses and determinants can all be calculated *a priori* to create  $\hat{\Lambda}_Y$ , in essence generating a model compensation system. This will be identical in both network bandwidth and computation as a standard ASR decoder, but has improved recognition performance.

### 5.1.3 Experiment and Results

In order to test the  $C_3$ -Internal-UO and  $C_3$ -External-UO algorithms, the Aurora2 digit framework was used. Since the focus of this technique is to overcome distortion due to vector quantization of MFCC vectors, and not distortion due to noise, only the clean speech portions of the test data were used. The recognizer used was the ISIP recognizer, altered to accept feature uncertainty, using the same model parameters and training techniques as the HTK system distributed with the Aurora2 database.

The vector quantization system used was a modified version of the ETSI feature extraction algorithm for distributed speech recognition, described in [18]. The baseline operation of this quantizer operates at 4.8kb/sec. Recognition using this bit-stream shows almost no degradation in recognition performance. In order to demonstrate UO robust recognition, it is thus necessary to decrease the bit-rate to a level that will introduce recognition errors. Rather than training a new VQ system from scratch in order to achieve a lower bit-rate, the source code for the ETSI coder was altered. The ETSI coder compresses pairs of MFCC and energy components in seven independent streams using between 5 and 8 bits per stream. For each of these pairs, three quarters of the codebook entries were chosen at random and removed. The original coder used 44 bits per frame, and the newer, more aggressive coder used 30 bits per frame, yielding a bit-rate of 3.3kbits/sec. This newer system will be shown to create a significant increase in recognition errors. While this is not an optimal method for finding a new low bitrate VQ system, it was effective for illustrating the  $C_3$ -Internal-UO and  $C_3$ -External-UO algorithms.

For both the Internal and External implementations, single mixture, zero-mean, diagonal

covariance matrices were assumed to model the feature uncertainty. Since the VQ system was not retrained for the Internal system, the covariance matrices  $\Sigma_i$  were calculated by using a training database, and by collecting statistics on the input feature and output features, and codebook indices. The database used was the TIMIT database, decimated to 8kHz, in order to match the sampling rate required by the ETSI coder. The internal structure of the coder was known to be in seven independent streams, each relating directly to two dimensions of the MFCC vector. This greatly simplified the training process for the  $\Sigma_i$ 's. If this internal structure were not known, this type of training would be difficult due to the possible VQ codebook entries,  $2^{30}$  in this case. To train the External form, the same training speech was used, but all codebook indices were ignored and *only* the global statistics were measured to find  $\Sigma_g$ .

To generate baseline performances, a system was first trained using uncompressed MFCCs from the Aurora2 training set, and testing was likewise performed on uncompressed MFCCs from the test set. A second system was trained using VQ-compressed MFCCs, and tested on VQ-compressed MFCCs. The results are shown in Figure 35, with uncompressed matched conditions giving a 2.1% error rate, typical for a digit system using only static cepstral parameters. The matched conditions using compressed speech show a significant drop in performance, with a 5.7% error rate. When the clean-trained system is tested on VQ-compressed data, the error rate drops to 3.6%. Note that for this case, training on the distorted data gives performance lower than testing in mismatched conditions. Two hypotheses for this behaviour are first, that the data required to jointly model both the VQ and speech is significantly larger than that required to model only the speech, and thus it is an issue of lack of training data. A second hypothesis is that the quantization step yields a finite set of points for training, which conflicts with the choice of a continuous density HMM.

It is informative to examine the distortion caused by the VQ of MFCC vectors. Figure 36 shows a histogram for each dimension of the static MFCC components of the TIMIT data used to train the  $C_3$ -External covariance matrix  $\Sigma_g$ . For each dimension, the histograms have been scaled such that  $\pm 1$  is the standard deviation over all of the training data. Each

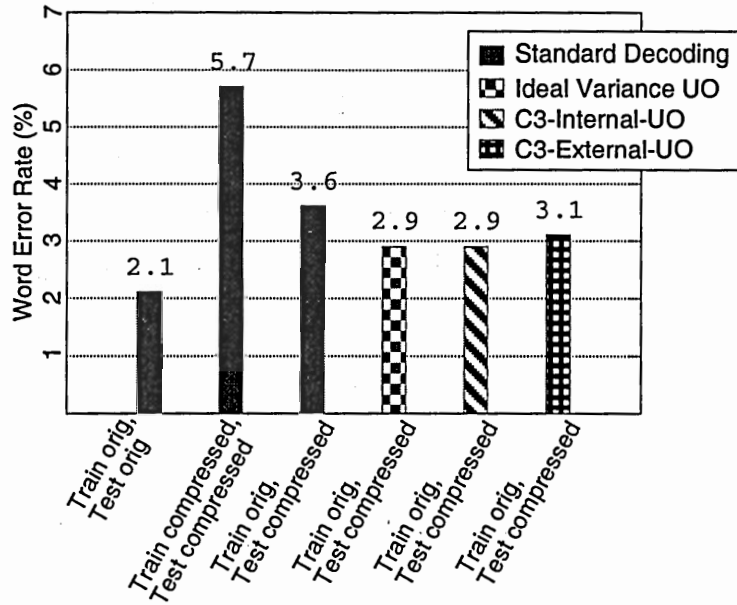
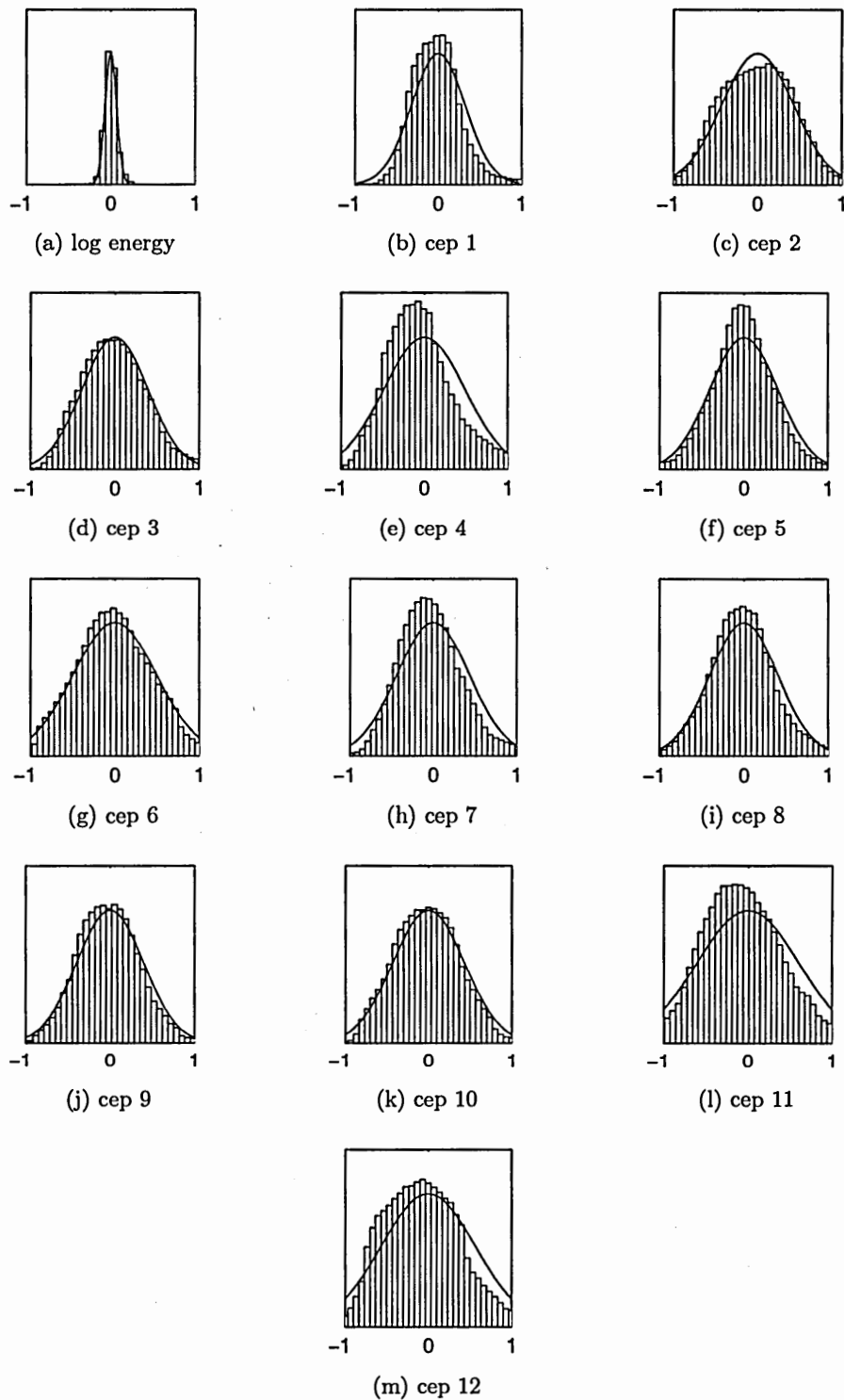


Figure 35: Digit recognition experiments showing the performance of the  $C_3$ -Internal-UO and  $C_3$ -External-UO versions of the VQ adaptation algorithm on the clean-speech portions Aurora2 digit database.

histogram has been overlaid with the normal r.v. used to generate  $\Sigma_g$ . A first observation is that the zero-mean assumption is accurate for MFCC vector quantization. A second observation is that in relation to the original cepstral vectors, the compressed versions do have a rather large variance along any particular dimension except log energy, thus it is no surprise the increase in the error rate.

A first test of the UO application to MFCC vector VQ is to use “ideal” variances during decoding. A stereo database of MFCC features was generated for pre- and post-VQ compression, giving vector sets  $X$  and  $Y$ , respectively. For any single feature vector at time  $t$ , the ideal variance was calculated as  $\sigma_t^2 = (\hat{x}_t - x_t)^2$ . Performing UO decoding in this ideal manner reduced the digit error rate to 2.9%, showing a 47% overall reduction in the errors due to the VQ compression is possible using UO.

Testing of the  $C_3$ -Internal-UO and  $C_3$ -External-UO algorithms show that a significant portion of the errors induced by the VQ compression can be removed. For the Internal algorithm, the error rate drops to 2.9%. Given that the ideal matched performance is



**Figure 36:** Histograms of MFCC VQ cepstral distortions for each of the static MFCC vector components, overlaid with the normal r.v. used for  $C_3$ -External-UO UO decoding. The x-axes have been normalized to the overall variance of each component.

2.1%, this translates to a 47% reduction in errors due to VQ compression of MFCC vectors. The performance of the External algorithm is slightly lower, giving an error rate of 3.1%, removing only 33% of the VQ compression errors.

An overview comparison of the algorithms is given in Table 10.

**Table 10:** Comparison of recognition systems using vector quantized MFCCs and single mixture Gaussian observation PDFs for UO.

| Recognition System | Internal VQ Access Required? | Computational Overhead | WER Reduction |
|--------------------|------------------------------|------------------------|---------------|
| Standard           | no                           | none                   | —             |
| $C_3$ -Internal-UO | yes                          | 33%                    | 47%           |
| $C_3$ -External-UO | no                           | none                   | 33%           |

## 5.2 *Speech Coder Adaptation*

This section now describes using UO techniques to improve recognition performance when features are extracted from the synthesized decompressed waveform, the  $C_4$  location of Figure 29.

### 5.2.1 Problem Description

Robust recognition at  $C_4$  using UO is a more difficult problem than the  $C_3$  scenario described in the previous section. In the  $C_3$  section, the assumption that the observation PDFs were modeled by Gaussians was a relatively good one, because of the inherent design of vector quantization training. At  $C_4$ , this is not quite as good an assumption. In particular, this is because the  $C_3$  parameters have been altered by the artificial synthesis stage, and features are now being extracted directly from a synthesized speech waveform. The  $C_4$  observation vectors thus contain an extra degree of distortion. Further, for this problem, it can be assumed that low-level access of internal state information of the VQ system is not accessible.

### 5.2.2 Two Proposed Solutions

Because information internal to the VQ system is not accessible, the Internal solution given in Section 5.1.2 is not applicable. The External method is, of course, still possible, and it

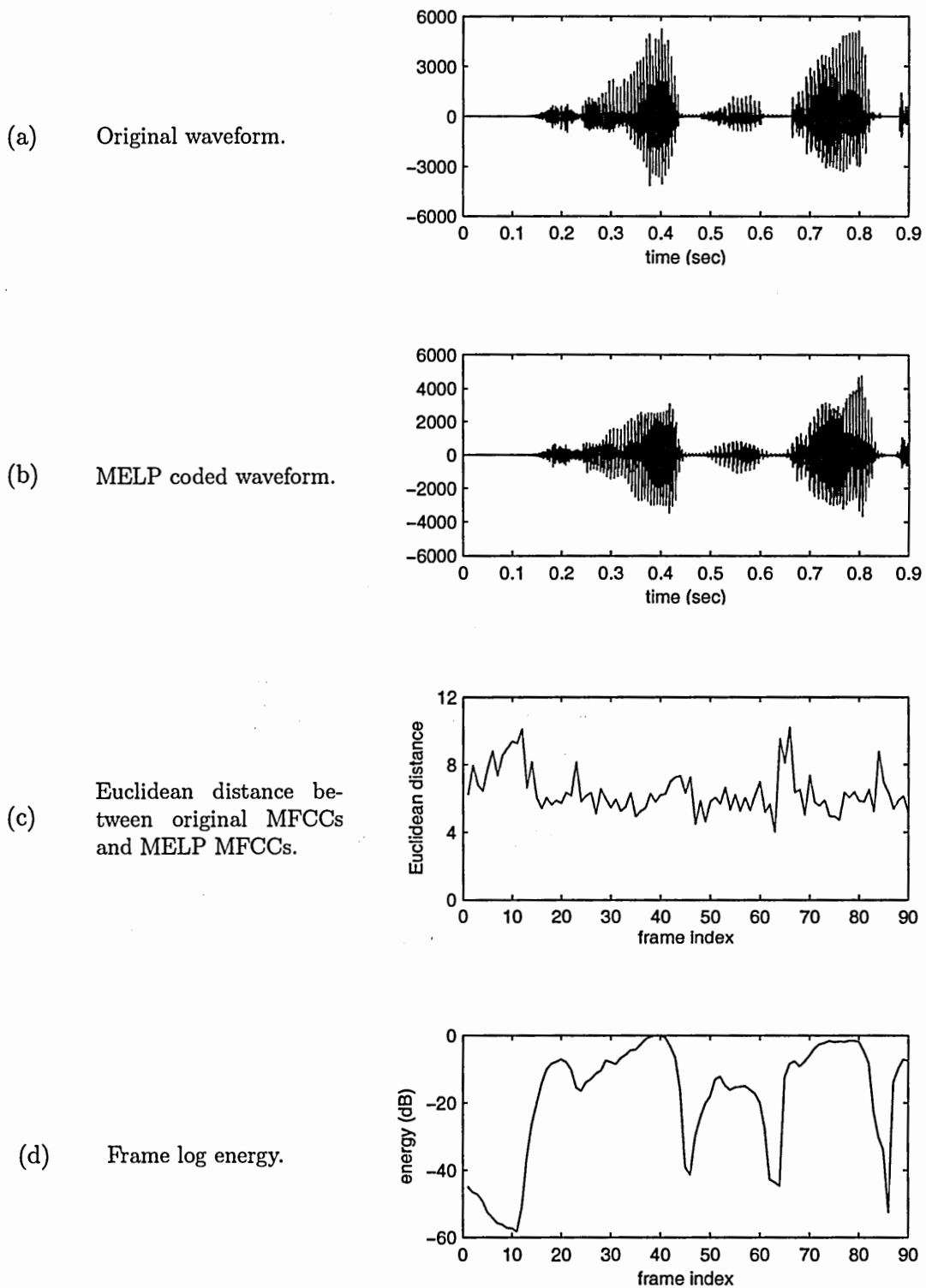
is along these lines that a solution will be developed. First, the External method will be implemented exactly as in Section 5.1.2. Second, a multiple-state version, based upon input feature energy, will be implemented. Finally, it will be shown how more general multiple state methods could be implemented to approximate the Internal solution.

The basic  $C_4$ -External method is implemented exactly as for the  $C_3$ -External method. A set of training data is passed through the coder, and MFCCs are extracted from the waveforms both before and after the compression step. The mean of the feature vector differences should be verified to be near zero, and then the global covariance matrix  $\Sigma_g$  can be found. One difference from the previous implementation is that here, the speech compression almost certainly induces some amount of delay in the signal. This delay should be accounted for, in order to compare input and output features, or else the variances will be artificially large.

While this method, referred to from here as the  $C_4$ -External-Global, does not have the codebook-specific covariance matrices as in the Internal solution, it does have the benefit that the feature extraction comes directly from a time domain waveform. Thus, the system is not dependent upon the feature set, frame size, or frame rate used in the VQ system.

It is now interesting to look at how well a global variance does at describing feature uncertainty. An easy method for doing this is to consider the original and MELP-compressed waveforms shown in parts (a) and (b), respectively, of Figure 37. Static MFCC features were extracted from both waveforms, after correction for coder-induced delay, and the Euclidean cepstral distance between frames of the two sets is shown in part (c). Comparing this to the plot of frame log energy plot in part (d), a roughly inverse relationship can be noted. The MFCC distortion is somewhat higher for the lower energy portions of the speech signal. While one example is not enough for conclusion, this trend continues across the entire compressed speech database.

This leads to a second technique, the  $C_4$ -External-2-State method, that segments frames based upon their relative energy levels, and uses different covariance matrices for the respective feature vector reliabilities. A two-state system was implemented with frames being assigned to be either high-energy or low-energy. This was first done during the training



**Figure 37:** Motivation for the  $C_4$ -External-2-State algorithm. MFCCs extracted from the original and MELP waveform show that distortion is related to signal energy. This motivates the use of the  $C_4$ -External-2-State energy-dependent distortion model.

phase, with log energy within 15dB of the maximum frame energy of the signal being classified as “high-energy.” The remainder were then put in a “low-energy” cluster. Covariance matrices were found for each cluster, yielding  $\Sigma_{\text{high}}$  and  $\Sigma_{\text{low}}$ , respectively.

Each of these covariances were then stored in the receiver, allowing it to choose the appropriate matrix at runtime, based upon the frame energy (which is quite accurately known, since these tests are on clean, albeit compressed speech). This method can be efficiently implemented in much the same way as the Global method. In the Global case, since the same observation covariance matrix is used for every observation vector, the entire model can be updated *a priori*, just as for standard model compensation. This same idea can be used here to generate two updated models, with the correct model chosen at run time based upon frame energy. Computation overhead is thus negligible, at the expense of memory.

Of course, extensions beyond the two state method implemented in this thesis are certainly possible. For example, another possibility is to perform a Viterbi recognition pass to align states to frames, and have new covariances for each phoneme, or even each state of each phoneme. Other clustering methods are also certainly possible. A multi-state External method such as this with a large number of states is in essence an approximation to the Internal solution that could not be directly implemented.

### 5.2.3 Experiment and Results

The two  $C_4$ -External algorithms described above were tested in a manner similar to that used for the  $C_3$  tests of the previous section. The clean training and testing data from the Aurora2 database was again used for training speech models and running recognition experiments, and one hour of randomly selected speech data downsampled to 8kHz from the TIMIT database was used for training the observation variances. The recognition system was again the ISIP recognizer extended for UO input, and the HMM parameters chosen were identical to those specified by the Aurora2 baseline system. In order to isolate the effects of the distortion, only static cepstral features are considered throughout this set of experiments.

The speech compression algorithm tested was the MELP coder. This compression algorithm was chosen because first, it is known to mildly degrade speech recognition, plus it is a common coder, being the U.S. federal standard for 2400 bit/sec communications.

The baseline performance for training and testing on the uncompressed speech shows a word error rate of 2.1%, typical for an application using only static cepstral features. The mediocre performance is, again, due to the lack of derivative features. In order to illustrate the use of UO to  $C_4$  speech recognition, however, this system is sufficient.

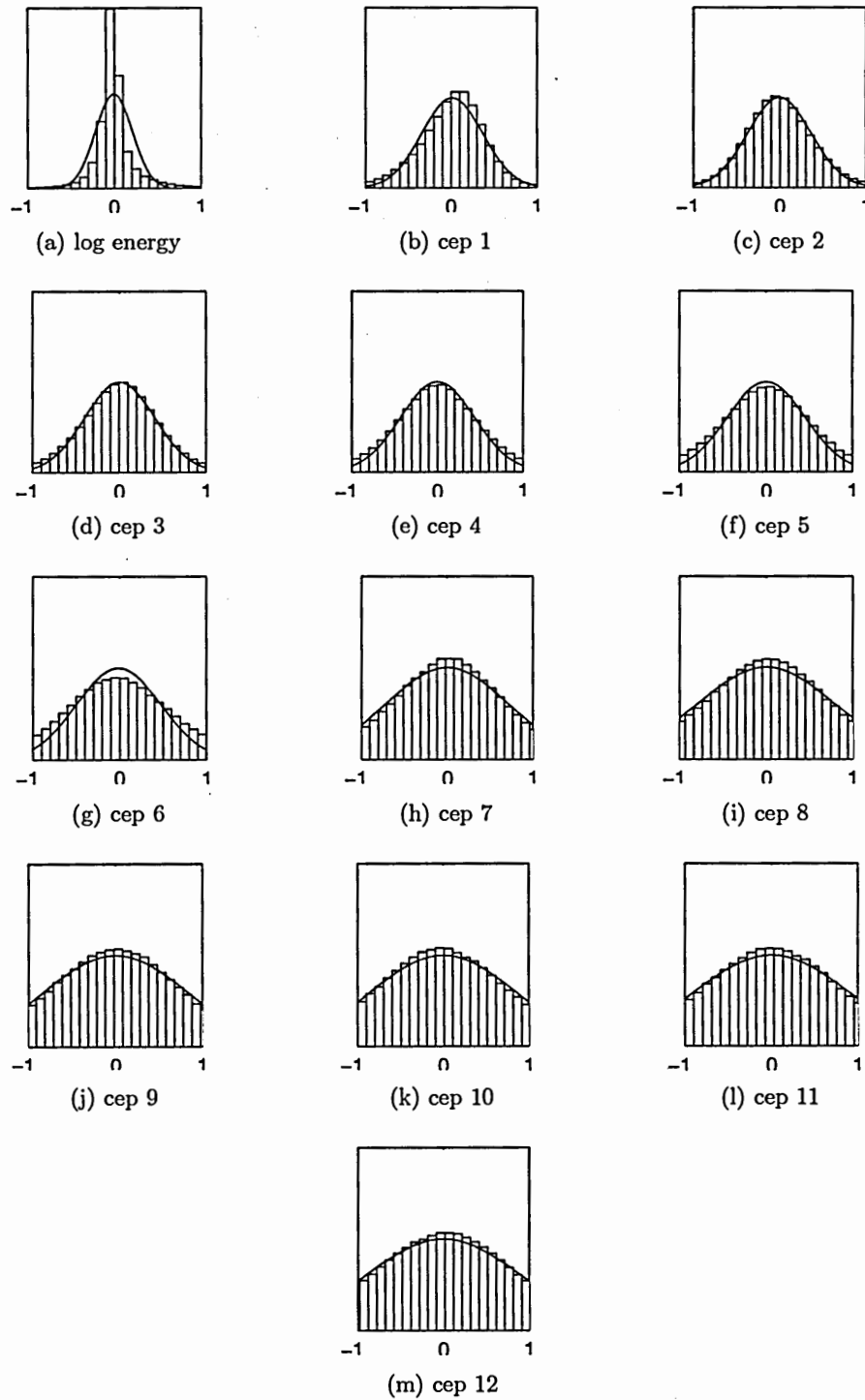
To test the matched condition case for compressed speech, all of the training and testing data was processed by MELP prior to feature extraction, and a new HMM set trained for this condition. This case shows a drop in performance, down to 3.3%. The most significant drop comes from training on uncompressed speech, but testing on MELP speech. For this case, the error rate rises to 4.8%. Thus the goal of the UO experiments is to bring the 4.8% error rate closer to 2.1% under mismatched testing conditions.

In order to train the  $C_4$ -External-Global variance  $\Sigma_g$ , it was first necessary to process the decimated TIMIT data using MELP, to generate a stereo training database. The roughly two frame delay induced by the MELP coder was accounted for prior to feature extraction, so that the distortion due to the coding process was measurable by simply taking a difference in the cepstral domain.

The first step is to verify that the mean of the feature distortion is zero. Figure 38 shows histograms of the distortion of each dimension of cepstral features found in the training database. Superimposed on these plots are normal PDFs fitted to each histogram. A conclusion from this diagram is that the distortion is relatively well modeled by a zero mean Gaussian with diagonal covariance matrix.

UO decoding using the diagonal global covariance matrix  $\Sigma_g$ , as represented by the individual normal r.v.s in Figure 38, shows a drop in the word error rate from 4.8% to 4.0%, as shown in the graph in Figure 39, removing 29.6% of the errors induced by the compression algorithm.

The second method analyzed was the  $C_4$ -External-2-State algorithm, as described previously. All frames with an energy within 15 dB of the maximum for the phrase were clustered



**Figure 38:** Histogram of MELP coder cepstral distortions for each of the static MFCC vector components, overlaid with the normal r.v. used for UO decoding. The  $x$ -axes have been normalized to the overall variance of each component.

into one group, and the low energy frames were clustered in a second group.

Using the statistical F-Test, hypothesis testing can be done to determine if the variances of these two groups of speech are indeed different. For this test, each dimension of the cepstral feature was assumed independent, and all frames were considered independent trials. Under these conditions, there is a much greater than 95% likelihood that the variances of the two sets should be modeled differently. This held true for all dimensions.

Decoding with the  $C_4$ -External-2-State algorithm caused the error rates to drop again slightly to 3.7%, removing 40.7% of the errors induced by compression. All results are given in Figure 39, where it can be seen that while an improvement is certainly possible for compressed speech, the improvements are not as dramatic as for the  $C_3$  MFCC VQ system for distributed speech recognition shown in the previous section.

An interesting question now is how much more can be gained by moving to a phoneme specific, or other high-cluster method to approximate an Internal solution. To answer this question, the Ideal Variance method was used. In this technique, the actual variance for each sample (i.e.,  $\Sigma_t = \text{diag}((\hat{x}_t - x_t)^2)$ ) is used during decoding, giving an optimal performance for the UO algorithm. This test found that the word error rate dropped to 3.6%, removing 44.4% of the errors caused by compression. This improved recognition rate is nearly identical to the  $C_4$ -External-2-State method. Thus, while a phoneme-specific (or other multi-state) system certainly could be implemented, it is unlikely to see any significant performance gain over the 2-State system when weighed against the added complexity cost.

The same experimental framework was also used to test phoneme recognition using the TIMIT database, in order to give a data point that is on something more than just digit recognition. The results, shown in the graph in Figure 40, show a similar pattern. Again, the ideal variance form shows that UO decoding has the capability of reducing compression errors by almost half (a 56.2% reduction), an amount which is almost met by the  $C_4$ -External-2State method (40.4%). Using only a single global diagonal covariance matrix still shows a drop in errors (27.0%). An overview comparison of the  $C_4$ -External algorithms presented in this section, as applied to MELP-coded speech, is given in Table 10. Analyzing these results shows that while a single global variance update does not reach the

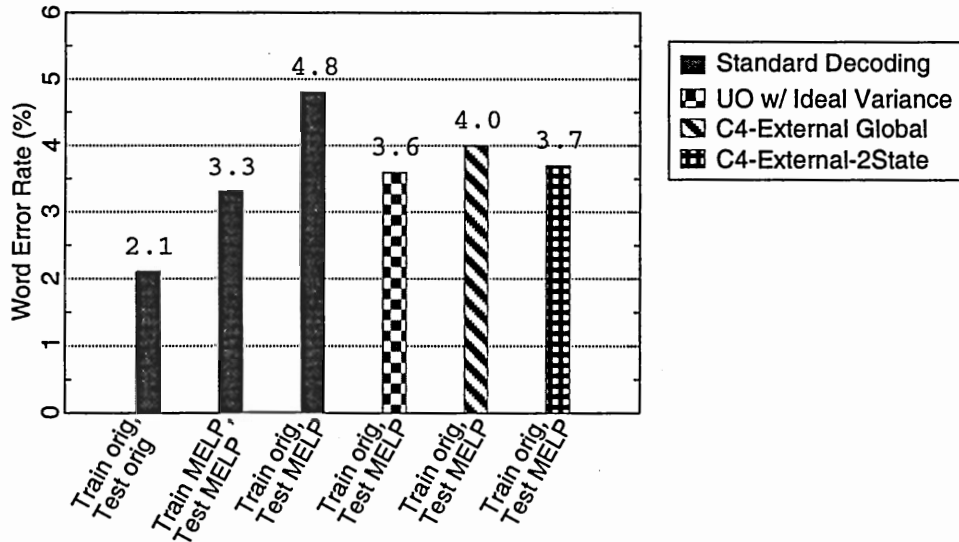


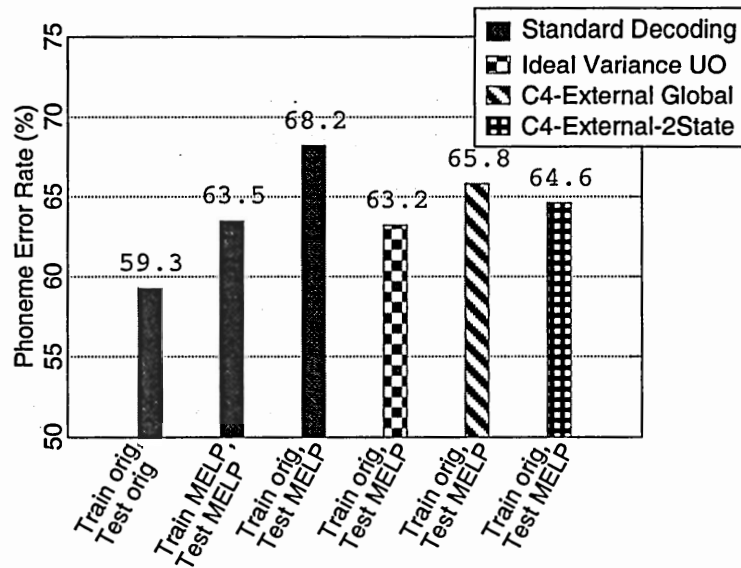
Figure 39: Digit recognition results for UO decoding of static MFCC feature vectors extracted from Aurora2-clean speech compressed with the MELP speech codec.

full capabilities of UO decoding, the two-state form comes relatively close.

Table 11: Comparison of recognition methods for MELP coded speech, with one mixture diagonal Gaussian observation covariances for UO.

| Recognition System              | Computational Overhead | Memory Overhead | Digit Error Reduction | Phoneme Error Reduction |
|---------------------------------|------------------------|-----------------|-----------------------|-------------------------|
| Standard                        | none                   | none            | —                     | —                       |
| C <sub>4</sub> -External-Global | none                   | none            | 29.6%                 | 27.0%                   |
| C <sub>4</sub> -External-2State | none                   | double          | 40.7%                 | 40.4%                   |
| Ideal Variance UO               | —                      | —               | 44.4%                 | 56.2%                   |

Another interesting question not covered in this thesis is the application of the variance information of one coder to that of another, for example training  $\Sigma_g$  using GSM, but performing recognition on G.729-compressed speech. This is of particular interest when the exact compression algorithm is not known at decoding time. For example, suppose a call center is recording incoming calls for audio data mining. Even if it is a simple matter to identify those calls that are from cellular phones, there are many different cellular standards. Since many of these, however, are based upon some type of smoothed spectrum vector



**Figure 40:** Phoneme recognition results for UO decoding of static MFCC feature vectors extracted from Aurora2-clean speech compressed with the MELP speech codec.

quantization, so long as the same feature extraction algorithm is used on the  $C_4$  waveform, it would seem possible that  $\Sigma$ 's trained with one coder might improve the recognition performance of speech that was actually compressed with a different coder. This topic is left for future work.

### 5.3 Handling Packet-Lossy Channels

This section now describes using UO decoding to improve speech recognition during short bursts of missing speech frames, common in distributed recognition.

#### 5.3.1 Problem Description

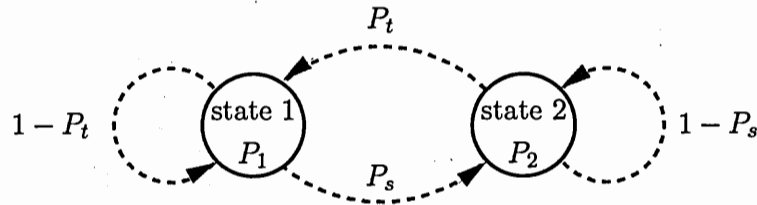
Many forms of digital communication systems currently in use face the problem of dropped packets. The more standard analog slow degradation of a signal due to fading, etc., is not necessarily a problem because of error correcting codes. But if a packet is corrupted, or not received, all of the information is likely lost, leaving a burst of one or more areas with no information interspersed in an otherwise perfect digital signal.

Packet drops can occur due to interference on wireless systems, overall network traffic through a router, or other reasons. Normal internet data traffic handles this situation by

allowing retransmission requests for packets that are not received, or have been corrupted during transmission. For real-time communication systems, however, retransmission requests are not practical. Instead, a maximum allowable latency is chosen, and any packets not received by this time are considered permanently lost, and the communication system must continue without the missing information. Cellular telephone, Voice over IP, and other packet-based communication systems used in DSR must therefore be designed to expect, and be robust to, dropped packets.

The speech coders used in packet based communication systems, for example MELP, G.729, and others, use extrapolation and frame repetition to handle missing frames. This has been an adequate solution for human to human communication, as with proper design, it can sound somewhat natural, and when intelligibility becomes difficult, retransmission requests can easily be handled by the communication system users (i.e., cell phone users waiting for a positive response to “Can you hear me now?”). Attempting speech recognition on data that has missing frames, however, gives extremely poor results, as shown in [62, 63]. While re-prompting is a possibility for interactive dialog systems, it is preferable to avoid this if possible. But interactive solutions are not possible at all for many applications, such as broadcast dictation, call monitoring, and database mining, and recognition must proceed with whatever data is available. For these cases, in packet lossy environments, UO techniques are well suited, allowing the recognizer to focus on the portions of the speech that *are* well known, ignoring what is not known. This is very similar to the special case scenario of completely unknown features discussed in Section 3.5.2.

A model often used for the transmission characteristics of a packet lossy system is the Gilbert model shown in Figure 41 [27]. This two-state Markov model is designed to mimic burst-loss scenarios often seen in real world system. During most operation, the system is in State 1, with a low likelihood of losing a packet. But occasionally, the system will transition to State 2, which has a high probability of losing a packet. Once in State 2, it will stay there for several frames before transitioning back to State 1. This models the common situation where if one packet is lost, more than likely there will be a succession of several lost packets in a burst fashion.



**Figure 41:** Gilbert model for burst packet loss.  $P_1$  and  $P_2$  are, respectively, the probabilities of a lost packet when the system is in State 1 or 2.  $P_s$  and  $P_t$  are, respectively, the probabilities of changing from State 1 to 2, or State 2 to 1.

As an example of handling burst packet loss, the common cellular G.729 coding standard uses a variety of techniques to maintain listener acceptability. Since latency is a critical issue for real-time communication, interpolation from both sides of the burst is not an available option. Instead, G.729 extrapolates smoothed spectral information from previous frames in order to retain causality. Further, as it is known that these frames are becoming further and further from the spectrum of the true speech, the gain is reduced. This has been found empirically to be more pleasing to the listener, even though it is artificially moving the output waveform even further from the true speech than is absolutely necessary. Immediately upon receipt of a valid packet, signaling the end of the burst, the coder returns to full gain with the received spectral and pitch information. Again, while this is good for the listener, it gives faulty information for the derivative components of the next several frames.

This analysis of G.729 shows clearly why speech recognition on this channel would be difficult. Even for short bursts, gain and derivative coefficients are being distorted due to the progressive gain reduction through the burst. For longer bursts, although the underlying speech phonetic content will be changing, the coder will artificially extend the segment and continue whatever phoneme was present when the burst began. Similar schemes are embedded in all modern speech coding algorithms.

### 5.3.2 Application of UO to Lossy Channels

A technique that can improve speech recognition in packet lossy systems is the application of the UO Decoding method as derived in Chapter 3. The assumption for this chapter is made

that the speech signal itself is otherwise clean and undistorted, except for dropped packets. Thus, the majority of speech vectors would be represented as *Dirac* delta PDFs since they contain no distortion. During a burst of lost packets, however, the feature vectors will no longer be completely well-known. As a burst lengthens, the feature vectors extrapolated for missing frames will become more and more unreliable. By representing these frames by PDFs, and using a UO decoder, speech recognition can be performed in a manner that performs a statistically optimal weighting of the language model and reliable feature information with the unreliable feature information. The result is a speech recognition system that will be better suited to correct recognition across bursts of missing feature frames.

To implement this version of occasional UO decoding, the computational cost is proportional to the number of dropped frames. If no frames are ever missing, there is no need for anything other than standard decoding. For missing frames only, feature vector PDFs  $f_t(\mathbf{x})$  must be generated, and the UO decoding equation for state output probability needs to be evaluated. The next section describes methods for determining PDFs.

### 5.3.3 PDF Feature Extraction

A first cut approach for generating PDF features for missing frames is direct use of the special case of Section 3.5.2, that is, assuming that absolutely nothing is known about the missing frames of speech. Enhanced speech feature means are thus irrelevant, and enhanced speech feature variances are set to infinity. This will be referred to as the *Completely Unknown Packet* method.

The Completely Unknown Packet method results in all acoustic information during the burst being completely ignored, and all decoding during the burst would be based on the language model.

A second, and more accurate, method is to use features extrapolated from prior frames, as is done for speech coding systems, and to estimate the variance of these extrapolated features. Using the analysis given above of how G.729 handles lost packets, a reasonable

**Table 12:** The set of Gilbert Model state transition and frame loss probabilities used for experiments in this section.

| $P_1$ | $P_2$ | $P_s$ | $P_t$ | Packet<br>Loss<br>Rate | MBL  | 90% of<br>Bursts |
|-------|-------|-------|-------|------------------------|------|------------------|
| 0.00  | 0.00  | 0.000 | 1.00  | 0%                     | 0.00 | = 0 frames       |
| 0.01  | 0.90  | 0.011 | 0.10  | 10%                    | 3.9  | ≤ 10 frames      |
| 0.01  | 0.90  | 0.016 | 0.06  | 20%                    | 5.5  | ≤ 13 frames      |

clean speech PDF generator for such a scenario would use the smoothed spectral representation during a burst for the Gaussian mean, but ignore the energy, as this is artificially lowered for listener acceptability. Note that this just represents repeating the last known good feature vector throughout the burst.

To generate variances, a large stereo database of artificially created burst losses was created. The assumption was made that vowels are longer than other parts of speech, and thus frame repetition during a vowel segment is likely to lead to smaller expected error than low energy speech, which will change more rapidly. This led to a three state frame-type classifier to treat nonspeech, low energy speech, and vowel frames separately. Thus, the burst losses in the stereo database were segmented into three groups, according to the frame-type of the last received frame.

For each group, then, the variance was found as a function of depth into the burst. The resulting implementation for UO decoding repeats the last received frame throughout the burst, using this as the mean of a Gaussian PDF. The variance is determined by using the frametype classification for the last received frame, and based upon how many frames into the burst, the proper diagonal observation covariance matrix from the burst training data is chosen.

### 5.3.4 Experiments and Results

In order to test the packet lossy PDF feature extraction from Section 5.3.3, a lossy channel was simulated using the channel model shown in Figure 41. In order to give a range of burst lengths and frequencies, several scenarios were tested. The  $P_1$ ,  $P_2$ ,  $P_s$ , and  $P_t$  values chosen are listed in Table 12, along with the overall packet loss rate and mean burst length.

Each of the loss scenarios were tested using phoneme recognition on the TIMIT database. Since the basic unit of this test is the phoneme, it does not take a long burst to obscure an entire phoneme. On the other hand, since the duration of a phoneme is short, the burst has little effect beyond the phonemes directly obscured. In a system using a full language model instead of a phoneme loop grammar, errors would be allowed to propagate, and thus bursts could easily effect wider areas than just the few phonemes they span. But a language model also would help, by allowing misrecognized single phonemes to be correctly reassigned due to context. The end conclusion is that the phoneme recognition results presented may or may not apply to large vocabulary systems, and thus any extrapolation of the results of this section to LVCSR should be avoided until further recognition results are run.

Training for the system used clean, burst-free speech. If all frames are received, the baseline performance of this system is a 38.7% phoneme error rate. The drop in recognition for the various burst cases was not very significant, with error rates of 40.1% and 41.3%, respectively, for 10% and 20% dropped frames. The application of UO decoding did improve the results, but not significantly, giving error rates of 39.8% and 40.9%, as shown in Table 13. For both loss rates, UO decoding was able to handle less than 20% of the errors introduced

**Table 13:** ASR Performance on packet-lossy data

| Packet<br>Loss<br>Rate(%) | Standard<br>Decoding<br>PER (%) | UO<br>Decoding<br>PER (%) |
|---------------------------|---------------------------------|---------------------------|
| 0                         | 38.7                            | 38.7                      |
| 10                        | 40.1                            | 39.8                      |
| 20                        | 41.3                            | 40.9                      |

by the missing frames of speech. The application of UO decoding to the lost packet scenario is thus only a moderate success.

## ***5.4 Summary and Discussion***

This chapter has discussed the application of Uncertain Observation decoding techniques to robust recognition in distributed speech recognition environments. The mismatch scenarios

examined were compression, and transmission across packet-lossy networks.

Two types of compression systems covered in this chapter: those that observe speech parameters prior to the compression algorithm's waveform synthesis state, and those that only have access to the final output waveform. These are referred to as systems performing feature extraction at the  $C_3$  and  $C_4$  locations, respectively. For pre-waveform synthesis systems ( $C_3$  systems), a further breakdown was made into two subtypes: ones that have internal access into the VQ stage, and ones that do not. These were referred to as  $C_3$ -Internal and  $C_3$ -External, respectively.

For  $C_3$ -Internal systems, a solution was proposed that each codebook entry  $i$  have its own cluster mean  $\mathbf{x}_i$ , and covariance matrix  $\Sigma_i$ . The means are obviously calculated at the time of the codebook training, and the variances can be either calculated at the same time, or as was done for the experiments for this chapter, calculated later, by using a database of training speech. The speech is run through the coder, and by noting which codebook index is used for each frame, variance information can be tallied for each  $i$ . This information is given *a priori* to the VQ receiver, thus network bandwidth remains identical to the standard case: only the codebook index need be transmitted. But at the receiver, the feature is represented as a PDF, rather than as a single point in space.

The solution proposed for  $C_3$ -External systems was much simpler. Access inside the VQ step was not allowed, thus while feature vectors could be observed both pre- and post-compression, they could not be associated with individual codebook indices. Thus, a system was design which created only a global covariance matrix, shared by all VQ clusters. The disadvantage of this system is slightly reduced performance, but the advantage is that UO decoding now translates into standard model adaptation. The original speech model set  $\Lambda_X$  can be converted *a priori* to  $\hat{\Lambda}_X$ , and no extra network bandwidth or CPU cycles are required for robust recognition.

Two methods were also proposed for the  $C_4$  scenario. It was assumed that if the recognizer input is at this location, access to compression algorithm internals is not possible. Hence, the first solution was to follow exactly that as done for the  $C_3$ -External scenario. Given that the synthesized waveform has an extra level of distortion not seen at the  $C_3$

location, the performance of this technique was slightly lower. Thus a second method was developed that cut a balance between the Internal and External solutions. A set of training data was again used, but separated according to signal energy. This multi-state algorithm showed an improvement compared to the  $C_4$ -External solution. Thus while the  $C_3$ -Internal algorithm cannot be directly applied, it can be approximated.

The proposed compression-robust methods were tested on two different styles of data. First, testing was performed on the clean portions of the Aurora2 digits database. Second, each method was tested for phoneme recognition on the clean TIMIT database. For  $C_3$  tests, a modified form of the ETSI distributed ASR front end was used, while for  $C_4$  tests the MELP coder was used.

This chapter has also described a novel technique for using the Uncertain Observation Decoding algorithm, coupled with a PDF feature extractor for bursts of lost frames, to improve automatic speech recognition on wireless and other delay-constrained networks. The technique is based upon knowing that further into a burst, the less reliable the frame estimates will be. Using the common G.729 speech coder, which has an algorithm to handle burst losses, it was shown the recognition degrades in packet-lossy environments. By augmenting the basic speech features with variance information during bursts, the UO method described was shown to improve recognition performance. The level of improvement, however, was not significant, and further work is required to generate a practical implementation.

It is worth mentioning that the application of UO decoding to compression methods, lost packets, and additive background could be combined to handle general "DSR of noisy speech". By creating a PDF estimator that is dependent on each of these distortions, their handling could be done in a single UO framework. This is in particular interesting because it has been noted in the literature [46] that while the 4.8kbps ETSI DSR compression has little affect on recognition of clean speech, it lowers significantly the accuracy of noisy speech. Thus, additive noise and DSR compression are not independent, and observation PDF estimation should be done in a joint manner.

An interesting side effect of this research is the concept of storing not only VQ cluster

means in a VQ receiver, but also the information about the cluster variance. This information is certainly known at the time the VQ system is trained, or as was shown can also be generated later by using example data for an existing system. Thus, upon receiving a codebook index, the receiver actually knows more than just the mean of the corresponding cluster, it knows information about how reliable this mean is, and can output a full PDF describing the cluster instead of just a single point in space. Few applications in the past have had need for this extra information, and hence it was discarded. But the UO decoding framework, which expects feature PDFs as input, can be directly fed codebook PDFs by such a receiver.

The final conclusion of this chapter is that in a system that compresses either speech or speech recognition feature vectors, recognition can be degraded due to aggressive coding. This degradation can be lessened by one third to one half by using feature vector variance information and the UO decoding algorithm. No extra network bandwidth is required for transmitting this variance information, as it can be determined *a priori* and stored in the receiver. Depending on the level of internal coder access (or improvement desired) the computational burden is either that of the standard UO algorithm for Internal solutions, or no extra computations for External solutions. Similarly, bursts of lost packets can be modelled by repeating the last known good frame, but increasing the variance through the burst. This implementation described showed modest recognition improvement for this scenario.

## CHAPTER VI

### UO TECHNIQUES APPLIED TO HMM TRAINING

#### *6.1 Motivation*

A problem often not addressed in speech recognition is the consequence of training on noisy data. Either the assumption is made that enough clean data can be collected to adequately train the speech HMMs, or else it is assumed that the noisy data is representative of the eventual testing environment, and thus multistyle training is performed. There are a few instances, however, where this is not a reasonable assumption. First, data collection is a time consuming and expensive task, as it requires both careful recording procedures and hand transcription. For some scenarios, the data collection process could be simplified if noisy data were allowed, even if testing is expected on clean data. Second, and more relevant, are cases where the amount of data is necessarily limited by the task. For example, in speaker adaptation, the goal is to observe a short segment of speech from a particular person and generate a transform such that the speech models better fit the specific talker [54]. If the few seconds observed contain background noise, then the adaptation will be joint on both the talker *and* the environment. If, however, the adaptation results are to be used again in the future in a new noise environment, it is desirable to separate the information about the speaker from that of the noise.

Thus the goal of the research in this chapter is to improve HMM parameter estimation or adaptation when using noisy training data. A basic method to accomplish this is to use standard feature transformation techniques to attempt to remove the noise from the training data. It is shown in this research that these results can be improved further by augmenting feature transformation with the uncertain observation principles described in Chapter 3. First, the standard Baum-Welch (BW) parameter reestimation method for training HMMs will be reviewed. This method will then be extended to allow uncertain observations. This new training algorithm, referred to as BW-UO, will then be demonstrated by training

HMMs using noisy data, and testing on clean speech. Experimental results of standard training, training with feature transformed data, and training using the new algorithm will be compared, showing that recognition accuracy can be increased by incorporating feature reliability information into the training process.

While this chapter focuses on Baum-Welch, implementation of UO techniques in other training or adaptation methods can be performed in a manner similar to that presented in this chapter.

## 6.2 Review of Baum-Welch Training

HMM parameters can be trained in a variety of ways, with the most common method being Baum-Welch reestimation. This section will briefly review the procedure (following the derivation given in [51]) allowing comparison to the BW-UO derivation presented in the next section.

The overall goal of Baum-Welch reestimation is to find HMM model parameters  $\lambda$  that maximize  $\Pr[\mathcal{O}|\lambda]$ , the probability of observing the given set of training data,  $\mathcal{O}$ . This is accomplished by beginning with an initial estimate for  $\lambda$ , and finding a new set  $\lambda'$  that increases the probability, such that  $\Pr[\mathcal{O}|\lambda'] > \Pr[\mathcal{O}|\lambda]$ . The procedure is then iterated to find new model sets until a maximum is reached and it is no longer possible to increase the likelihood further. For this review section, the training data  $\mathcal{O}$  is given as

$$\mathcal{O} = (\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_T), \quad (81)$$

a sequence of  $T$  observation vectors extracted from the transcribed training audio, i.e., audio with a known corresponding sequence of states, but unknown exact locations of the state transitions.

Each iteration begins with the initial model parameter estimate,  $\lambda$ , and Baum's auxiliary function  $\mathcal{Q}(\lambda', \lambda)$  which is defined as

$$\mathcal{Q}(\lambda', \lambda) = \sum_{\text{all } \mathbf{q}} \Pr[\mathcal{O}, \mathbf{q}|\lambda] \log \Pr[\mathcal{O}, \mathbf{q}|\lambda']. \quad (82)$$

Maximizing this function with respect to  $\lambda$  is guaranteed by the EM algorithm [10] to find the desired  $\lambda'$  with either  $\Pr[\mathcal{O}|\lambda'] > \Pr[\mathcal{O}|\lambda]$ , or  $\lambda' = \lambda$ . The probability terms in

Equation 82 can be written in terms of the HMM parameters,

$$\Pr[\mathcal{O}, \mathbf{q}|\lambda] = \pi_{q_0} \prod_{t=1}^T a_{q_{t-1}q_t} b_{q_t}(\mathbf{x}_t), \text{ and} \quad (83)$$

$$\log \Pr[\mathcal{O}, \mathbf{q}|\lambda] = \log \pi_{q_0} + \sum_{t=1}^T \log a_{q_{t-1}q_t} + \sum_{t=1}^T \log b_{q_t}(\mathbf{x}_t). \quad (84)$$

Separating  $\mathcal{Q}$  into  $2N + 1$  independent parts,

$$\mathcal{Q}(\lambda', \lambda) = \mathcal{Q}_\pi(\lambda', \pi) + \sum_{i=1}^N \mathcal{Q}_{a_i}(\lambda', \mathbf{a}_i) + \sum_{i=1}^N \mathcal{Q}_{b_i}(\lambda', \mathbf{b}_i) \quad (85)$$

where

$$\pi = [\pi_1, \pi_2, \dots, \pi_N], \quad \mathbf{a}_i = [a_{i1}, a_{i2}, \dots, a_{iN}], \quad \mathbf{b}_i = [\mu_i, \Sigma_i], \quad (86)$$

and

$$\mathcal{Q}_\pi(\lambda', \pi) = \sum_{i=1}^N \Pr[\mathbf{O}, q_0 = i|\lambda] \log \pi'_i \quad (87)$$

$$\mathcal{Q}_{a_i}(\lambda', \mathbf{a}_i) = \sum_{j=1}^N \sum_{t=1}^T \Pr[\mathcal{O}, q_{t-1} = i, q_t = j|\lambda] \log a'_{ij} \quad (88)$$

$$\mathcal{Q}_{b_i}(\lambda', \mathbf{b}_i) = \sum_{t=1}^T \Pr[\mathcal{O}, q_t = i|\lambda] \log b'_i(\mathbf{x}_t). \quad (89)$$

Each term of the auxiliary function can thus be maximized separately over  $\lambda$ , subject to the following constraints:

$$\begin{aligned} \sum_{j=1}^N \pi'_j &= 1, \quad \pi'_j \geq 0 \\ \sum_{j=1}^N a'_{ij} &= 1, \quad a'_{ij} \geq 0, \quad \forall i \\ \int_{-\infty}^{\infty} b'_j(\mathbf{x}) d\mathbf{x} &= 1, \quad b'_j(\mathbf{x}) \geq 0, \quad \forall j, \mathbf{x} \end{aligned}$$

The global maximum of each of these three constrained functions can be found using Lagrange multipliers (see [58], for example, for details), giving the following updated parameter

estimates for the case of a single mixture Gaussian  $b_j(\mathbf{x})$ :

$$\pi'_i = \frac{\Pr[\mathcal{O}, q_0=i|\lambda]}{\Pr[\mathcal{O}|\lambda]} \quad (90)$$

$$a'_{ij} = \frac{\sum_{t=1}^T \Pr[\mathcal{O}, q_{t-1}=i, q_t=j|\lambda]}{\sum_{t=1}^T \Pr[\mathcal{O}, q_{t-1}=i|\lambda]} \quad (91)$$

$$\mu'_j = \frac{\sum_{t=1}^T \Pr[\mathcal{O}, q_t=i|\lambda] \mathbf{x}_t}{\sum_{t=1}^T \Pr[\mathcal{O}, q_t=i|\lambda]} \quad (92)$$

$$\Sigma'_j = \frac{\sum_{t=1}^T \Pr[\mathcal{O}, q_t=i|\lambda] (\mathbf{x}_t - \mu_j)(\mathbf{x}_t - \mu_j)^T}{\sum_{t=1}^T \Pr[\mathcal{O}, q_t=i|\lambda]} \quad (93)$$

Together, these terms make up the desired new set of model parameters,  $\lambda'$ .

Each of the probabilities in the update equations above can be related to the parameters of the HMM by using the *alpha* forward variable and *beta* backward variable. These are defined as

$$\alpha_t(i) = \Pr[\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_T, q_t=i|\lambda] \quad (94)$$

and

$$\beta_t(i) = \Pr[\mathbf{x}_{t+1}, \dots, \mathbf{x}_T, q_t=i|\lambda]. \quad (95)$$

The probabilities necessary for reestimation can be written in terms of  $\alpha$  and  $\beta$ :

$$\Pr[\mathcal{O}, q_t=i|\lambda] = \alpha_t(i)\beta_t(i) \quad (96)$$

$$\Pr[\mathcal{O}|\lambda] = \sum_{i=1}^N \alpha_t(i)\beta_t(i) = \sum_{i=1}^N \alpha_T(i) \quad (97)$$

$$\Pr[\mathcal{O}, q_{t-1}=i, q_t=j|\lambda] = \alpha_{t-1}(i) \sum_{i=1}^N a_{ij} b_j(\mathbf{x}_t) \beta_t(j) \quad (98)$$

Direct calculation of  $\alpha_t(j)$  and  $\beta_t(j)$  is effectively intractable, but a very efficient method exists by using the lattice structure inherent in state transitions. For the  $\alpha_t(j)$ , this is solved using the following procedure:

1. Initialization

$$\alpha_1(j) = \pi_j b_j(\mathbf{x}_1), \quad (99)$$

2. Induction

$$\alpha_{t+1}(j) = \left[ \sum_{i=1}^N \alpha_t(i) a_{ij} \right] b_j(\mathbf{x}_{t+1}). \quad (100)$$

Similarly, for the backward variable  $\beta$ ,

1. Initialization

$$\beta_T(i) = 1, \quad (101)$$

2. Induction

$$\beta_t(i) = \sum_{j=1}^N a_{ij} b_j(\mathbf{x}_{t+1}) \beta_{t+1}(j). \quad (102)$$

The end result is a reestimation method that is iterative in nature, built on top of the forward and backward variables. The procedure described is both efficient, and can be shown to be non-decreasing, producing a  $\Pr[\mathcal{O}|\lambda]$  at the end of every iteration that is at least as good as the previous iteration. Convergence is only guaranteed to be to a local maximum, but stability issues of gradient search methods, which do not provide monotonic improvement of  $\Pr[\mathcal{O}|\lambda]$ , are avoided.

### 6.3 Baum-Welch Reestimation with Observation Uncertainty

For the case of noisy training data, it is desired in this research to incorporate the Uncertain Observation techniques described in Chapter 3 into the HMM parameter reestimation method described above. The resulting novel training method, which will be referred to as the *BW-UO* training algorithm, is derived in the remainder of this section.

The BW-UO algorithm begins by replacing the training data observation sequence  $\mathcal{O}$  of Equation 81 with  $\bar{\mathcal{O}}$ , a new sequence that is specified as PDFs:

$$\bar{\mathcal{O}} = (f_1(\mathbf{x}), f_2(\mathbf{x}), \dots, f_T(\mathbf{x})), \quad (103)$$

just as was given during the initial UO formulation in Equation 14. Without loss of generality, this section will continue assuming single-mixture Gaussian PDFs for observations

$f_t(x)$ , thus each observation will be described by its own mean and variance,  $\mu_t$  and  $\Sigma_t$ , respectively.

The auxiliary function for the PDF observation sequence is

$$Q(\lambda', \lambda) = \sum_{\text{all } \mathbf{q}} \Pr[\bar{\mathcal{O}}, \mathbf{q}|\lambda] \log \Pr[\bar{\mathcal{O}}, \mathbf{q}|\lambda'], \quad (104)$$

with the probability terms being

$$\Pr[\bar{\mathcal{O}}, \mathbf{q}|\lambda] = \pi_{q_0} \prod_{t=1}^T a_{q_{t-1}q_t} \int_{-\infty}^{\infty} f_t(\vartheta) b_j(\vartheta) d\vartheta, \text{ and} \quad (105)$$

$$\log \Pr[\bar{\mathcal{O}}, \mathbf{q}|\lambda] = \log \pi_{q_0} + \sum_{t=1}^T \log a_{q_{t-1}q_t} + \sum_{t=1}^T \log \left[ \int_{-\infty}^{\infty} f_t(\vartheta) b_{q_t}(\vartheta) d\vartheta \right]. \quad (106)$$

Using Equation 53 from the UO decoding chapter, we can rewrite these directly in terms of HMM and observation parameters:

$$\Pr[\bar{\mathcal{O}}, \mathbf{q}|\lambda] = \pi_{q_0} \prod_{t=1}^T a_{q_{t-1}q_t} \frac{1}{\sqrt{2\pi}\sigma_\tau} \exp \left\{ -\frac{1}{2\sigma_\tau^2} (\mu_t - \mu_{q_t})^2 \right\} \quad (107)$$

$$\begin{aligned} \log \Pr[\bar{\mathcal{O}}, \mathbf{q}|\lambda] &= \log \pi_{q_0} + \sum_{t=1}^T \log a_{q_{t-1}q_t} \\ &\quad + \sum_{t=1}^T \left[ \log \left( \frac{1}{\sqrt{2\pi}\sigma_\tau} \right) - \frac{1}{2\sigma_\tau^2} (\mu_t - \mu_{q_t})^2 \right] \end{aligned} \quad (108)$$

where the  $\sigma_\tau = \sigma_t + \sigma_{q_t}$  terms are the variances along the diagonal of  $\Sigma_\tau = \Sigma_t + \Sigma_{q_t}$ .

The auxiliary function can again be split into independent parts, giving the terms

$$Q_\pi(\lambda', \pi) = \sum_{i=1}^N \Pr[\bar{\mathcal{O}}, q_0=i|\lambda] \log \pi'_i \quad (109)$$

$$Q_{a_i}(\lambda', \mathbf{a}_i) = \sum_{j=1}^N \sum_{t=1}^T \Pr[\bar{\mathcal{O}}, q_{t-1}=i, q_t=j|\lambda] \log a'_{ij} \quad (110)$$

$$Q_{b_i}(\lambda', \mathbf{b}_i) = \sum_{t=1}^T \Pr[\bar{\mathcal{O}}, q_t=i|\lambda] \log \left[ \int_{-\infty}^{\infty} f_t(\vartheta) b'_j(\vartheta) d\vartheta \right] \quad (111)$$

that can be maximized independently, subject to the same constraints as for standard BW reestimation. The maximization results in the following parameter estimates for the case

of a HMM states modeled by a single Gaussian mixture  $b_j(\mathbf{x})$ :

$$\pi'_i = \frac{\Pr[\bar{\mathcal{O}}, q_0=i|\lambda]}{\Pr[\bar{\mathcal{O}}|\lambda]} \quad (112)$$

$$a'_{ij} = \frac{\sum_{t=1}^T \Pr[\bar{\mathcal{O}}, q_{t-1}=i, q_t=j|\lambda]}{\sum_{t=1}^T \Pr[\bar{\mathcal{O}}, q_{t-1}=i|\lambda]} \quad (113)$$

$$\mu'_j = \frac{\sum_{t=1}^T \Pr[\bar{\mathcal{O}}, q_t=i|\lambda] \mathbf{x}_t}{\sum_{t=1}^T \Pr[\bar{\mathcal{O}}, q_t=i|\lambda]} \quad (114)$$

$$\Sigma'_j = \frac{\sum_{t=1}^T \Pr[\bar{\mathcal{O}}, q_t=i|\lambda] (\mathbf{x}_t - \mu_j)(\mathbf{x}_t - \mu_j)^T}{\sum_{t=1}^T \Pr[\bar{\mathcal{O}}, q_t=i|\lambda]} \quad (115)$$

Calculating the probabilities for the parameter estimates can be done in a manner similar to that using the *alpha* and *beta* variables used for standard BW training. For BW-UO, we define  $\bar{\alpha}$  and  $\bar{\beta}$  as

$$\bar{\alpha}_t(j) = \Pr[f_1(\mathbf{x}), \dots, f_t(\mathbf{x}), q_t=j|\lambda] \quad (116)$$

$$\bar{\beta}_t(j) = \Pr[f_{t+1}(\mathbf{x}), f_{t+2}(\mathbf{x}), \dots, f_T(\mathbf{x})|q_t=j, \lambda] \quad (117)$$

which are calculated using UO information:

### 1. Initialization

$$\bar{\alpha}_1(j) = \int_{-\infty}^{\infty} f_1(\boldsymbol{\vartheta}) b_1(\boldsymbol{\vartheta}) d\boldsymbol{\vartheta} \quad (118)$$

### 2. Induction

$$\bar{\alpha}_{t+1}(j) = \left[ \sum_{i=1}^N \bar{\alpha}_t(i) a_{ij} \right] \int_{-\infty}^{\infty} f_{t+1}(\boldsymbol{\vartheta}) b_j(\boldsymbol{\vartheta}) d\boldsymbol{\vartheta}, \quad (119)$$

and the backward variable  $\bar{\beta}$ :

### 1. Initialization

$$\bar{\beta}_T(j) = 1 \quad (120)$$

## 2. Induction

$$\bar{\beta}_t(j) = \sum_{k=1}^N a_{jk} \bar{\beta}_{t+1}(k) \int_{-\infty}^{\infty} f_{t+1}(\vartheta) b_j(\vartheta) d\vartheta. \quad (121)$$

All of the probabilities necessary for the parameter reestimates can be written in terms of  $\bar{\alpha}$  and  $\bar{\beta}$ :

$$\Pr[\bar{\mathcal{O}}, q_t = i | \lambda] = \bar{\alpha}_t(i) \bar{\beta}_t(i) \quad (122)$$

$$\Pr[\bar{\mathcal{O}} | \lambda] = \sum_{i=1}^N \bar{\alpha}_t(i) \bar{\beta}_t(i) = \sum_{i=1}^N \bar{\alpha}_T(i) \quad (123)$$

$$\Pr[\bar{\mathcal{O}}, q_{t-1} = i, q_t = j | \lambda] = \bar{\alpha}_{t-1}(i) \sum_{i=1}^N a_{ij} b_j(\mathbf{x}_t) \bar{\beta}_t(j) \quad (124)$$

The BW-UO reestimation scheme for HMM parameters described above has the same properties as standard BW training. Most importantly, for every iteration is it guaranteed that  $\Pr[\bar{\mathcal{O}} | \lambda'] \geq \Pr[\bar{\mathcal{O}} | \lambda]$ , ensuring stable convergence to a local maxima. Further, this training algorithm includes information about feature reliability directly into a strong probabilistic framework, allowing parameter training to noisy features to proceed in an optimal manner.

## 6.4 Analysis

The simplest method of visualizing the effect of this new UO-based training algorithm is to consider training with the Viterbi algorithm. Viterbi training is a simplification of Baum-Welch training where HMM forced alignment is first run, giving an optimal state sequence  $\mathbf{q}$ :

$$\mathbf{q} = (q_1, q_2, \dots, q_T). \quad (125)$$

Using this sequence,  $\Pr[q_t = j | \mathcal{O}]$  is now either 0 or 1, depending on whether or not the optimal state sequence goes through state  $j$  at time  $t$ , i.e.,  $q_t = j$ . The standard BW

reestimation equations given in Equations 90-93 can be rewritten as simple counting:

$$\pi'_j = \frac{\delta(q_0 - j)}{T} = \frac{M_0}{T} \quad (126)$$

$$a'_{ij} = \frac{\sum_{t=1}^T \delta(q_{t-1} - i) \cdot \delta(q_t - j)}{\sum_{t=1}^T \delta(q_{t-1} - i)} = \frac{M_1}{M_2} \quad (127)$$

$$\mu'_j = \frac{\sum_{t=1}^T \delta(q_t - j) \mathbf{x}_t}{\sum_{t=1}^T \delta(q_t - j)} = \frac{1}{M_3} \sum_{m=1}^{M_3} \mathbf{x}_{k_m} \quad (128)$$

$$\Sigma'_j = \frac{\sum_{t=1}^T \delta(q_t - j) (\mathbf{x}_t - \mu_j)(\mathbf{x}_t - \mu_j)'}{\sum_{t=1}^T \delta(q_t - j)} = \frac{1}{M_3} \sum_{m=1}^{M_3} (\mathbf{x}_{k_m} - \mu_j)(\mathbf{x}_{k_m} - \mu_j)^T \quad (129)$$

where

- $M_0$  is the number of times  $q_0$  equals  $j$ ,
- $M_1$  is the number of times the state transition from  $i$  to  $j$  is observed in  $\mathbf{q}$ ,
- $M_2$  is the number of times state  $i$  occurs in  $\mathbf{q}$ , and
- $(k_1, \dots, k_{M_3})$  are the  $M_3$  time indices that state  $j$  is in the optimal state sequence  $\mathbf{q}$ .

When converting the BW-UO algorithm to Viterbi style, as was done above for standard BW training, the most obvious change is that the optimal state sequence  $\bar{\mathbf{q}}$  is calculated using a UO decoder. By accounting for background noise, this UO state sequence is improved over  $\mathbf{q}$  that would have been found with either a standard decoder, or with a decoder using

noise-removed features. This new state sequence is then used in the Viterbi reestimation equations

$$\pi'_j = \frac{\bar{M}_0}{T} \quad (130)$$

$$a'_{ij} = \frac{\bar{M}_1}{\bar{M}_2} \quad (131)$$

$$\mu'_j = \frac{1}{\bar{M}_3} \sum_{m=1}^{\bar{M}_3} \mu_{k_m} \quad (132)$$

$$\Sigma'_j = \frac{1}{\bar{M}_3} \sum_{m=1}^{\bar{M}_3} (\mu_{k_m} - \mu_j)(\mu_{k_m} - \mu_j)^T \quad (133)$$

where

- $\bar{M}_0$ ,  $\bar{M}_1$ ,  $\bar{M}_2$ , and  $\bar{M}_3$  have the same meaning as for the standard Viterbi case, except the UO alignment  $\bar{q}$  is used, and
- $\mathbf{x}_t$  has been replaced by  $\mu_t$ , the mean of time  $t$ 's feature PDF.

Thus the effect of using observation uncertainty in the case of Viterbi training is to:

- first, better identify the underlying state sequence during the initial forced alignment pass, and then
- generate parameter reestimates using  $\mu_t$ 's, which effectively are noise-transformed features.

As an example, consider the Viterbi alignments shown in Figure 42. Shown are the alignments at the end of the final stage of training of one of the training phrases (“two three seven”) that is used in the experiments described in the next section. The top alignment is shown for reference only, as it is the output the result of training with clean speech. The second alignment, shown in 42(b), is for the same segment of speech, but is from a training run that uses only noisy speech. It is obvious that the noise has significantly affected the ability of training to correctly align states to the words.

The third alignment in 42(c) is the result of training on noise-removed features. Here, training is obviously able to align much better to the underlying speech. Finally, 42(d)

shows the alignment resulting from a UO training run. There is little difference visual between these two methods, but as will be shown in the experiments section below, training with UO does give a small further improvement in recognition performance.

An obvious special case is training with clean speech. For this case, observation variance is zero, and  $\mu_t = \mathbf{x}_t, \forall t$ . As was shown in Chapter 3, for this scenario, recognition (and thus forced state alignment) will be identical to standard recognition. Thus, the update equations 130-133 are identical to standard Viterbi update equations 126-129, as expected.

## 6.5 *Evaluation on the Aurora2 Database*

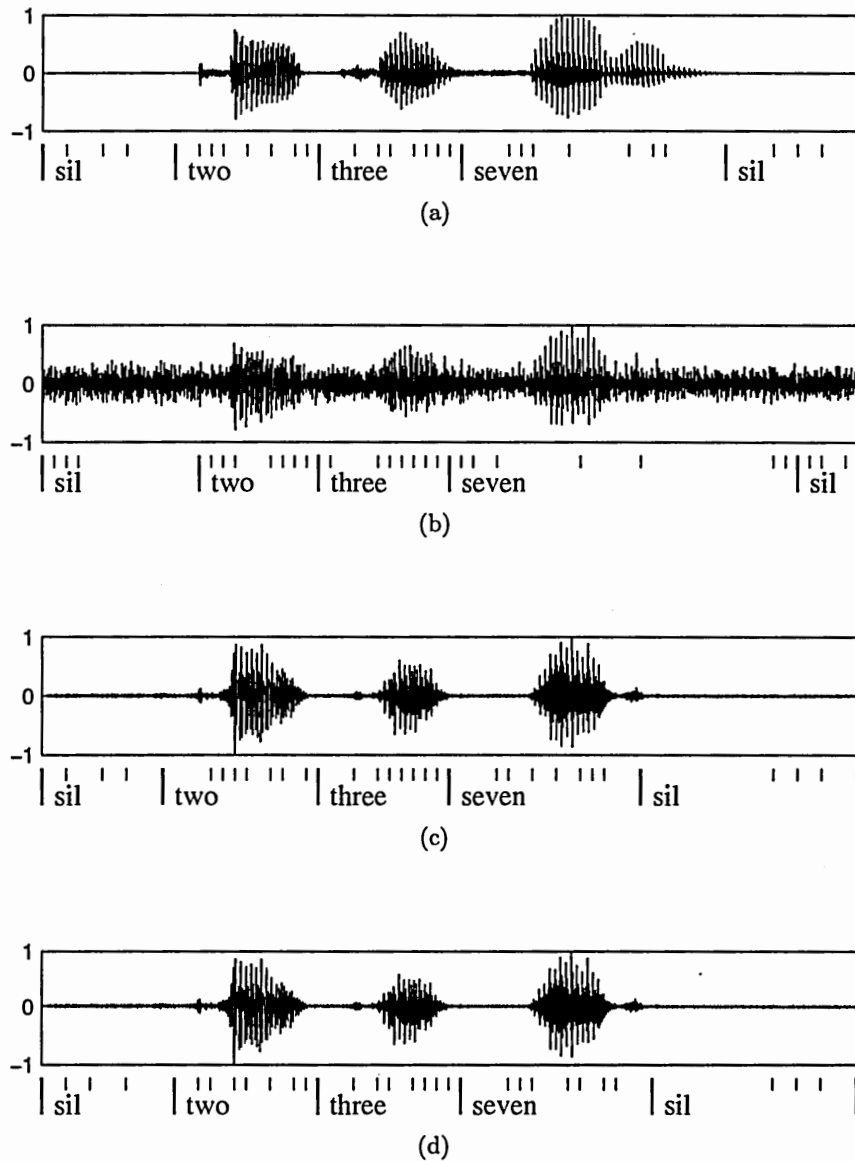
The BW-UO HMM parameter reestimation algorithm derived in this chapter has been implemented by altering the ISIP speech recognition system. Training was extended to use feature PDFs as input, and implement reestimation equations 112-115.

The BW-UO training method was tested by running experiments that train using noisy data, but perform recognition on clean data. The datasets used for the experiments were extracted from the Aurora2 dataset. The training set comprised 4,004 digit strings that had an SNR of 5 dB from Aurora2 “testa” subset. Because the Aurora2 experiment uses artificial additive noise, stereo versions of all of these files without noise were also available. The test set chosen was the 1,688 clean files normally used for training in the Aurora2 experiment setup.

For all experiments, the standard Aurora2 feature extraction method (thirteen MFCCs, plus first and second derivatives) was followed. Digits were modelled using the same left-to-right whole word models used for Aurora2.

The training methodologies that were tested were:

- standard Baum-Welch training using noisy data,
- standard BW that used noise-removed feature data,
- a BW-UO system estimating variances from the noisy data, and
- a BW-UO system that uses “ideal” variances, similar to the oracle experiments of Chapter 3, extracting variances from the stereo version of the training data.



**Figure 42:** Comparison of Viterbi alignments used in UO training. Part (a) shows the ideal Viterbi state alignment at the end of the final iteration of standard training, using clean speech. Part (b) shows the alignment, when standard training uses noisy speech. Part (c) is similar, but training has used noise-removed features. Finally, part (d) uses full PDF features for UO training.

The two BW-UO experiments required full PDFs for features. Single mixture Gaussians with diagonal covariance matrices  $f_t(\mathbf{x}) \sim \mathcal{N}(\mathbf{x}; \boldsymbol{\mu}_t, \boldsymbol{\Sigma}_t)$  were chosen for modeling these observation PDFs. For the realistic system, which estimated feature variances only using the noisy data, this was accomplished using the MCMC-AR method described in Chapter 4.3.1. The “oracle” system used the means of the MCMC-AR system, but used the true variance, calculated as  $\tilde{\boldsymbol{\Sigma}}_t = \text{diag}(y_t - \mathbf{x}_t)^2$ , giving  $f_t(\mathbf{x}; \boldsymbol{\mu}_t, \tilde{\boldsymbol{\Sigma}}_t)$ . Obviously, this requires both the clean and noisy versions of the training data.

For the standard BW system that used noise-removed feature data, the mean  $\boldsymbol{\mu}_t$  of  $f_t(\mathbf{x}) \sim \mathcal{N}(\mathbf{x}; \boldsymbol{\mu}_t, \boldsymbol{\Sigma}_t)$  was used for  $\mathbf{x}_t$ . The variance information in  $\boldsymbol{\Sigma}_t$  was discarded.

The first two experiments show the matched performance cases of first training and testing on clean speech, and second, training and testing on noisy speech. The clean-on-clean scenario shows an error rate of 2.1%, and the baseline performance for matched noisy conditions gives a 22.8% word error rate.

Two baseline experiments were run to show the effect of training/testing mismatch. First, training with clean data and testing noisy speech showed a drop in performance, with a resulting word error rate of 75.8%. More severe, however, were the effects of training on noisy speech and trying to recognize clean speech. Here the word error rate goes to 80.9%, showing that this type of mismatch has consequences as severe as the clean-train, noisy-test scenario.

When the MCMC-AR method is used to generate noisy-removed features, and variance information is disregarded, a significant improvement in performance is seen over the baseline mismatched condition, as the word error rate drops to 30.4%.

When the full BW-UO training algorithm is implemented using both the mean and variance information obtained from the MCMC-AR feature extraction method, a further improvement is seen, with the error rate dropping to 29.1%. This shows that for this feature extraction method, by far the majority of the improvement is gained by handling only the feature transformation.

The oracle experiment showed that more significant jump in performance, dropping the error rate to 28.3%. This again shows the need for better algorithms for estimating feature

variance, although another possibility is that the diagonal variance model is insufficient in the MFCC domain, and full covariance observation models are required. Work on this area is left for future research.

The results of the training experiments are summarized in the chart in Figure 43.

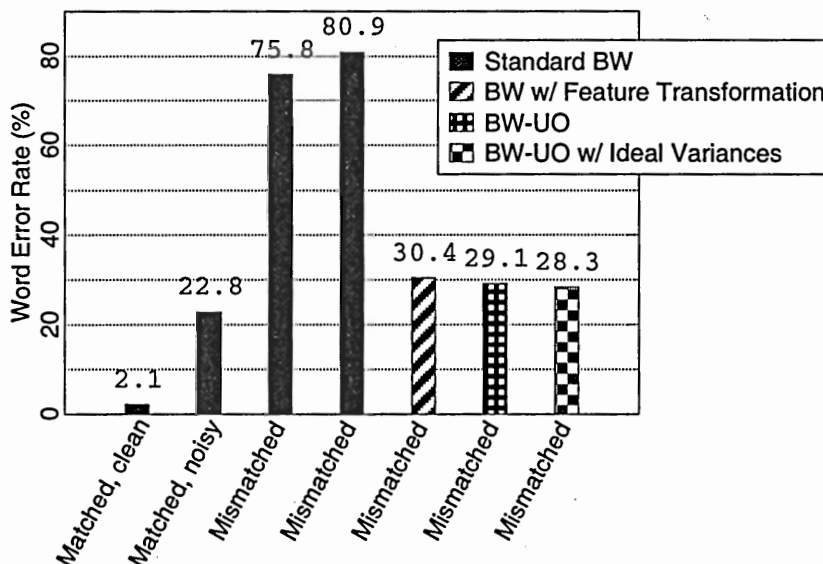


Figure 43: Digit experiments showing the results of training. “Mismatched” refers to training with noisy speech, and testing with clean speech.

## 6.6 Summary and Discussion

This chapter has presented a technique for incorporating information about observation uncertainty into the training of speech parameters. The specific implementation chosen was to extend Baum-Welch HMM parameter reestimation into the new BW-UO training algorithm. This new algorithm has the desirable properties of efficient implementation, monotonically non-decreasing probability of observing the training data, and a solid framework for incorporating observation uncertainty. While BW was chosen for this chapter, the same principles can of course be applied to other training methods.

Experiments using this new training algorithm were tested by training on noisy data and testing on clean speech. The new BW-UO algorithm was compared against training with just standard Baum-Welch, and also Baum-Welch that used feature vectors that have

had noise removal performed. This last form is effectively BW-UO with all variances set to zero, and training with just the PDF means. While feature transformed data significantly improved results, the full UO form showed even further improvement.

The usefulness of this technique is most applicable to training scenarios where available data is limited, and potentially noisy. In particular, a likely scenario that would benefit from UO training is when performing speaker adaptation. The data collected could certainly be noisy, and if the adapted models are to be used in the future when the noise is not present, it is desired to have the adaptation focus only on the speaker dependent information, and not the environment dependent information. Application of UO techniques to adaptive training are left for future work.

## CHAPTER VII

### CONCLUSIONS

In this work, a novel framework for adapting an HMM speech recognition system during mismatched conditions using *observation uncertainty* is presented. The technique leverages the knowledge that speech enhancement can be performed with varying degrees of accuracy at varying points in time and frequency, due to the nonstationary characteristics of both speech, and the type of distortion. To this end, speech features during noise or other distortion are represented as  $d$ -dimensional PDFs instead of simple points in  $d$ -dimensional space. A new HMM decoding algorithm, *Uncertain Observation Decoding*, is presented, which extends the standard technique to handle PDF features as input.

This method is not restricted to any particular feature representation, requiring only the ability to accurately model speech enhancement output by a PDF. This method was shown to have similar performance to model compensation techniques, yet retains the frame-to-frame adaptability of feature enhancement to time varying distortions.

Successful application of this technique to speech containing severe background noise was shown, as was successful application to speech coding and compression artifacts. Application to lossy channel transmission systems did show statistically significant, though overall moderate, improvement. Further, application of observation uncertainty during HMM training was shown to aid situations where the training data is distorted.

#### 7.1 Contributions

##### Uncertain Observation Theory:

- A new speech recognition algorithm, *Uncertain Observation Decoding*, was derived. It uses feature PDFs instead of single points in feature space as input.
- The new algorithm was shown to allow adaptation to time-varying mismatches between training and testing environments, even when the mismatch varies for every

frame of speech.

- It was shown that the general form of the UO decoder solves for the expected value of the state output probability given the current model of the distortion.
- Using Gaussian PDFs for speech features, a simplified UO decoder was derived that could be efficiently implemented, requiring only a 33% processing overhead.
- A new technique for extending any filterbank domain based speech feature enhancement algorithm to give single mixture, diagonal covariance Gaussian feature PDFs was developed using time-frequency local expected-SNR.
- A second technique for computing feature PDFs was developed by extending the Markov Chain Monte Carlo speech enhancement technique to yield arbitrary observation PDFs.
- The Uncertain Observation Decoding algorithm was used to adaptively compensate for time varying, additive noise in experiments on Aurora2 digit data and TIMIT data. It was determined that recognition could in all cases be improved using UO decoding.
- An extension to vector quantization was proposed that stores full cluster PDF information in the VQ decoder, which can then be used in a UO decoder.
- The UO decoding algorithm was used to improve recognition performance of packet-lossy data in conjunction with the G.729 coder, although improvement was modest.
- HMM training was extended to also use observation PDFs. This technique allows training robust speech models given noisy or distorted training data.

## *7.2 Observations*

During the course of this research, several interesting observations were made regarding uncertain observation techniques. Foremost among these was that a high degree of improvement was available when good modeling of feature PDFs is possible, as shown by

the Oracle and Ideal Variance analysis techniques. The task of robust speech recognition was shown to be almost equivalent to the task of modeling feature variance. Throughout this thesis, several examples of observation PDF estimation were shown for several types of distortions, but especially for additive noise, the recognition improvement seen was only a fraction of that available if better PDFs were available. Thus, observation PDF estimation is an excellent field for future research. While this is still a difficult problem in and of itself, it provides future researchers with a new direction towards improving speech recognition accuracy.

A second observation was that during noisy environments, speech features could be modeled by a single Gaussian in the filterbank domain, and with only a small loss in accuracy, have the variance converted to a single mixture, diagonal covariance matrix in the cepstral domain. Better performance was shown using either full covariance feature PDFs, but the computation tradeoff does not justify their use. This is an important finding if UO decoding is to be useful in future real-world systems.

Another observation is that any feature transformation technique designed for robust recognition can be extended to the UO framework. Letting the transformed features represent the means of a multidimensional Gaussian, and estimating variances of each, will give simple Gaussian observation PDFs. So long as the variance estimation step is reasonably accurate, this system will show better performance than feature transformation alone.

A final observation is that the three robustness applications evaluated in this thesis (additive noise, packet-lossy channels, and compression artifacts) could be considered in a single framework, with observation PDFs modelling all three effects in a seamless framework.

### *7.3 Future Work*

There are many directions in which this research can be continued. One is to explore efficient implementations for real time systems. There are several possible processing shortcuts discussed, but a more thorough analysis of these is recommended. In particular, precalculation or estimation of the combined covariance matrix (rather than explicit calculation) would be helpful, as would a method for more quickly estimating the combined covariance

matrix determinant.

Another direction for further work is to explore non Gaussian or truncated Gaussian feature PDFs. Diagonal covariance Gaussians are not necessarily the best model for all types of distortion, and improvements could be made if this area is explored.

Finally, the most promising direction for future work is to find better methods for estimating Gaussian feature variances. It was shown by Ideal Variance and Oracle analysis techniques that knowing the correct feature variance is second only to knowing the clean features themselves. Improvement upon the PDF estimation techniques presented in this thesis is certainly possible especially by recursive recognition/estimation. Further, other domains of distortion could also be considered, such as the effect of convolutional distortions. A difficult problem not addressed in this thesis was how to estimate variances of first and second feature derivatives. If derivative feature variances can be accurately calculated, including this in recognition will only further improve recognition performance.

# APPENDIX A

## PHONEME SET

Table A.1: Phoneme set used for the phoneme loop grammar recognition experiments performed throughout this thesis.

| Phoneme Name | Example Word    | Phoneme Name | Example Word    |
|--------------|-----------------|--------------|-----------------|
| aa           | b <u>o</u> b    | m            | <u>m</u> et     |
| ae           | b <u>a</u> t    | n            | <u>n</u> et     |
| ah           | b <u>u</u> t    | ng           | <u>ng</u> ing   |
| ao           | b <u>o</u> ught | b            | <u>b</u> et     |
| eh           | b <u>e</u> t    | d            | <u>d</u> ebt    |
| er           | b <u>i</u> rd   | g            | <u>g</u> et     |
| ih           | b <u>i</u> t    | dh           | <u>dh</u> at    |
| iy           | b <u>e</u> at   | v            | <u>v</u> at     |
| ow           | b <u>o</u> at   | z            | <u>z</u> oo     |
| uw           | b <u>o</u> ot   | zh           | <u>zh</u> azure |
| uh           | b <u>o</u> ok   | k            | <u>k</u> at     |
| aw           | d <u>o</u> wn   | p            | <u>p</u> et     |
| ay           | b <u>u</u> y    | t            | <u>t</u> en     |
| ey           | b <u>a</u> it   | f            | <u>f</u> at     |
| oy           | b <u>o</u> y    | s            | <u>s</u> ing    |
| l            | <u>l</u> et     | sh           | <u>sh</u> ut    |
| w            | <u>w</u> it     | th           | <u>th</u> ing   |
| r            | <u>r</u> ent    | hh           | <u>hh</u> at    |
| y            | <u>y</u> ou     | ch           | <u>ch</u> urch  |
|              |                 | jh           | <u>jh</u> udge  |

## REFERENCES

- [1] ACERO, A., *Acoustical and Environmental Robustness in Automatic Speech Recognition*. Norwell: Kluwer Academic Publishers, 1993.
- [2] ACERO, A., DENG, L., KRISTJANSSON, T., and ZHANG, J., "HMM adaptation using vector Taylor series for noisy speech recognition," in *Proc. ICSLP '00*, (Beijing, China), 2000.
- [3] AHMED, S. and TRESP, V., "Some solutions to the missing feature problem in vision," *Advances in Neural Information Processing*, pp. 393–400, 1993.
- [4] ARROWOOD, J. A. and CLEMENTS, M. A., "HMM decoding in noisy environments using uncertain observations," in *Proc. SIP*, (Kauai, HA), pp. 107–111, 2002.
- [5] ARROWOOD, J. A. and CLEMENTS, M. A., "Using observation uncertainty in HMM decoding," in *Proc. ICSLP '02*, (Denver, CO), pp. 1561–1564, 2002.
- [6] BARKER, J., JOSIFOVSKI, L., COOKE, M. P., and GREEN, P. D., "Soft decisions in missing data techniques for robust automatic speech recognition," in *Proc. ICSLP '00*, (Beijing, China), pp. 373–376, 2000.
- [7] BOLL, S. F., "Suppression of acoustic noise in speech using spectral subtraction," *IEEE Trans. on Acoustics, Speech, and Signal Processing*, vol. ASSP-27, no. 2, pp. 113–120, 1979.
- [8] COOKE, M., GREEN, P., JOSIFOVSKI, L., and VIZINHO, A., "Robust automatic speech recognition with missing and unreliable data," *Speech Communication*, vol. 34, pp. 267–285, jun 2001.
- [9] DELLER, J. R., PROAKIS, J. G., and HANSEN, J. H. L., *Discrete-Time Processing of Speech Signals*. Macmillan, 1993.
- [10] DEMPSTER, A. P., LAIRD, N. M., and RUBIN, D. B., "Maximum likelihood from incomplete data via the EM algorithm," *Journal of the Royal Statistic Society*, vol. 39, no. 1, pp. 1–38, 1977.
- [11] DENG, L., DROPPA, J., and ACERO, A., "Log-domain speech feature enhancement using sequential MAP noise estimation and a phase-sensitive model of the acoustic environment," in *Proc. ICSLP '02*, (Denver, CO), pp. 1813–1816, 2002.
- [12] DENG, L., DROPPA, J., and ACERO, A., "Dynamic compensation of HMM variances using the feature enhancement uncertainty computed from a parametric model of speech distortion," *to be published in IEEE Trans. on Speech and Audio Processing*, 2004.
- [13] DROPPA, J., ACERO, A., and DENG, L., "Uncertainty decoding with SPLICE for noise robust speech recognition," in *Proc. ICASSP '02*, vol. 1, (Orlando, FL), pp. 57–60, 2002.

- [14] DROPPA, J., DENG, L., and ACERO, A., "Evaluation of SPLICE on Aurora 2 and 3 tasks," in *Proc. ICASSP '01*, vol. 1, (Salt Lake City, Utah), pp. 209–212, 2001.
- [15] EALEY, D., KELLEHER, H., and PEARCE, D., "Harmonic tunnelling: Tracking non-stationary noises during speech," in *Proc. Eurospeech '01*, (Aalborg, Denmark), pp. 437–440, 2001.
- [16] EPHRAIM, Y. and MALAH, D., "Speech enhancement using a minimum mean-square error log-spectral amplitude estimator," *IEEE Trans. on Acoustics, Speech, and Signal Processing*, vol. ASSP-33, no. 2, pp. 443–445, 1985.
- [17] ERELL, A. and WEINTRAUB, M., "Filterbank energy estimation using mixture and Markov models for recognition of noisy speech," *IEEE Trans. on Speech and Audio Processing*, vol. 1, pp. 68–76, 1993.
- [18] ETSI ES 201 108, *Speech Processing, Transmission, and Quality Aspects (STQ); Distributed Speech Recognition; Advanced Front-End Extraction Algorithm; Compression Algorithm*. European Telecommunications Standards Institute, October 2002.
- [19] EVANS, N. W. D. and MASON, J. S., "Noise estimation without explicit speech, non-speech detection: a comparison of mean, median, and modal based approaches," in *Proc. Eurospeech '01*, (Aalborg, Denmark), pp. 893–896, 2001.
- [20] FONG, W., GODSILL, S. J., DOUCET, A., and WEST, M., "Monte-Carlo smoothing with application to audio signal enhancement," *IEEE Trans. on Signal Processing*, vol. 50, pp. 438–449, 2002.
- [21] FREY, B. J., KRISTJANSSON, T., DENG, L., and ACERO, A., "Learning dynamic noise models from noisy speech for robust speech recognition," *Advances in Neural Information Processing Systems*, 2001.
- [22] GALES, M. J. F., *Model-Based Techniques for Noise Robust Speech Recognition*. Ph.D. dissertation, University of Cambridge, Cambridge, UK, 1995.
- [23] GALES, M. J. F., "Predictive model-based compensation schemes for robust speech recognition," *Speech Communication*, vol. 25, pp. 49–74, 1998.
- [24] GALES, M. J. F. and YOUNG, S. J., "HMM recognition in noise using parallel model combination," in *Proc. Eurospeech '93*, (Berlin, Germany), pp. 837–840, 1993.
- [25] GALES, M. J. F. and YOUNG, S. J., "A fast and flexible implementation of PMC," in *Proc. ICASSP '95*, (Detroit, MI), pp. 133–136, 1995.
- [26] GAROFOLO, J. S., LAMEL, L. F., FISHER, W. M., FISCUS, J. G., PALLETT, D. S., and DAHLGREN, N. L., "The darpa timit acoustic-phonetic continuous speech corpus cdrom," tech. rep., Linguistic Data Consortium, Philadelphia, PA, 1993.
- [27] GILBERT, E. N., "Capacity of a burst noise channel," in *Bell System Technical Journal*, vol. 39, pp. 1253–1265, sep 1960.
- [28] GREEN, P., BARKER, J., COOKE, M., and JOSIFOVSKI, L., "Handling missing and unreliable information in speech recognition," in *Proceedings of AISTATS*, 2001.

- [29] GREEN, P., BARKER, J. P., and COOKE, M., "Robust ASR based on clean speech models: An evaluation of missing data techniques for connected digit recognition in noise," in *Proc. Eurospeech '01*, (Aalborg, Denmark), pp. 213–216, 2001.
- [30] HANSEN, J. H. L. and CLEMENTS, M. A., "Constrained iterative speech enhancement with application to speech recognition," *IEEE Trans. on Acoustics, Speech, and Signal Processing*, vol. 39, pp. 795–805, Apr. 1991.
- [31] HUO, Q. and LEE, C. H., "A Bayesian predictive classification approach to robust speech recognition," *IEEE Trans. on Speech and Audio Processing*, vol. 8, no. 2, pp. 200–204, 2000.
- [32] JIM, H. K. and COX, R., "Bitstream-based feature extraction for wireless speech recognition," in *Proc. ICASSP '00*, vol. 3, (Istanbul, Turkey), pp. 1607–1610, 2000.
- [33] JOSIFOVSKI, L., "Missing data techniques for robust speech recognition," tech. rep., Speech and Hearing Research Group, University of Sheffield, Sheffield, UK, 2001.
- [34] JUANG, B.-H. and RABINER, L. R., "Hidden Markov models for speech recognition," *Technometrics*, vol. 33, pp. 251–272, Aug. 1991.
- [35] KIM, D. Y., UN, C. K., and KIM, N. S., "Speech recognition in noisy environments using first-order vector Taylor series," *Speech Communication*, vol. 24, pp. 39–49, 1998.
- [36] KOMORI, Y., KOSAKA, T., YAMAMOTO, H., and YAMADA, M., "Fast parallel model combination noise adaptation processing," in *Proc. Eurospeech '97*, (Rhodos, Greece), pp. 1527–1530, 1997.
- [37] KOSAKA, T., YAMAMOTO, H., YAMADA, M., and KOMORI, Y., "Instantaneous environment adaptation techniques based on fast PMC and MAP-CMS methods," in *Proc. ICASSP '98*, (Seattle, WA), pp. 789–792, 1998.
- [38] KRISTJANSSON, T. and FREY, B., "Accounting for uncertainty in observations: A new paradigm for robust automatic speech recognition," in *Proc. ICASSP '02*, vol. 1, (Orlando, FL), pp. 61–64, 2002.
- [39] LIPPMAN, R. P., "Accurate consonant perception without mid-frequency speech energy," *IEEE Trans. on Speech and Audio Processing*, vol. 4, no. 1, pp. 66–69, 1996.
- [40] MARTIN, R., "Noise power spectral density estimation based on optimal smoothing and minimum statistics," *IEEE Trans. on Speech and Audio Processing*, vol. 9, pp. 504–512, 2001.
- [41] MORENO, P. J., *Speech Recognition in Noisy Environments*. Ph.D. dissertation, Carnegie Melon University, Pittsburgh, PA, 1996.
- [42] MORRIS, A., BARKER, J., and BOURLARD, H., "From missing data to maybe useful data: Soft data modelling for noise robust ASR," in *Proc. Workshop on Innovation in Speech Processing*, (Stratford-upon-Avon, England), 2001.
- [43] MORRIS, R. W., ARWOOD, J. A., and CLEMENTS, M. A., "Markov Chain Monte-Carlo methods for noise robust feature extraction using the autoregressive model," in *Proc. Eurospeech '03*, (Geneva, Switzerland), 2003.

- [44] NADAS, A., "Optimal solution of a training problem in speech recognition," *IEEE Trans. on Acoustics, Speech, and Signal Processing*, vol. ASSP-33, no. 1, pp. 326–329, 1985.
- [45] PAPOULIS, A., *Probability, Random Variables, and Stochastic Processes*. McGraw Hill, third ed., 1991.
- [46] PARIHAR, N., PICONE, J., PEARCE, D., and HIRSCH, H., "Performance analysis of the Aurora large vocabulary baseline system," in *Proc. Eurospeech '03*, (Geneva, Switzerland), 2003.
- [47] PEARCE, D. and HIRSCH, H. G., "The Aurora experimental framework for the performance evaluation of speech recognition systems under noisy conditions," in *Proc. ICSLP '00*, (Beijing, China), pp. 29–32, 2000.
- [48] PICONE, J., "Signal modeling techniques in speech recognition," *Proc. of the IEEE*, vol. 81, pp. 1215–1247, Sept. 1993.
- [49] PICONE, J., "The ISIP public domain decoder for large vocabulary conversational speech recognition," tech. rep., Institute for Signal and Information Processing, <http://www.isip.msstate.edu/projects/speech/software>, 15 Feb. 1999.
- [50] PICONE, J., "Session 10 - LVCSR systems," in *SRSTW'00*, (Starkville, MS), 26 May 2000.
- [51] RABINER, L. R. and JUANG, B.-H., *Fundamentals of Speech Recognition*. Prentice Hall, 1993.
- [52] REVENY, P. and DRYGAJLO, A., "Statistical estimation of unreliable features for robust speech recognition," in *Proc. ICASSP '00*, (Istanbul, Turkey), pp. 1731–1734, 2000.
- [53] RIPLEY, B. D. and HJORT, N. L., *Pattern Recognition and Neural Networks*. Cambridge University Press, 1996.
- [54] SANKAR, A. and LEE, C. H., "A maximum-likelihood approach to stochastic matching for robust speech recognition," *IEEE Trans. on Speech and Audio Processing*, vol. 2, pp. 789–792, 1998.
- [55] SARIKAYA, R. and HANSEN, J. H. L., "PCA-PMC: A novel use of a priori knowledge for fast parallel model combination," in *Proc. ICASSP '00*, (Istanbul, Turkey), pp. 1113–1116, 2000.
- [56] SOHN, J. and SUNG, W., "A voice activity detector employing soft decision based noise spectrum adaptation," in *Proc. ICASSP '98*, (Seattle, WA), pp. 365–368, 1998.
- [57] VARGA, A. P., STEENEKEN, H. J. M., TOMLINSON, M., and JONES, D., "The noisex-92 study on the effect of additive noise on automatic speech recognition," (Malvern, U.K.), 1992.
- [58] VENKATARAMAN, A., "Research topics in maximum likelihood," tech. rep., Stanford Research Institute, <http://www.speech.sri.com/people/anand/771/html>, 16 Sept. 1999.

- [59] WARREN, R. M., RIENER, K. R., BASHFORD, J. A., and BRUBAKER, B. S., "Spectral redundancy: Inteligibility of sentences heard through narrow spectral slits," *Perception & Psychoacoustics*, vol. 57, no. 2, pp. 175-182, 1995.
- [60] YANG, R. and HAAVISTO, P., "An improved noise compensation algorithm for speech recognition," in *Proc. ICASSP '96*, (Atlanta, GA), pp. 45-52, 1996.
- [61] YOMA, N. B., "Speaker verification in noise using a stochastic version of the weighted Viterbi algorithm," *IEEE Trans. on Speech and Audio Processing*, vol. 3, pp. 158-166, 2002.
- [62] ZHONG, X. and ARROWOOD, J. A., "Multiple description coding for recognizing voice over IP," in *Proc. 10<sup>th</sup> DSP Workshop*, (Callaway Gardens, GA), 2002.
- [63] ZHONG, X., ARROWOOD, J. A., and CLEMENTS, M. A., "Speech coding and transmission for improved automatic recognition," in *Proc. ICSLP '02*, (Denver, CO), pp. 1845-1848, 2002.
- [64] ZHU, Q., ISELI, M., CUI, X., and ALWAN, A., "Noise robust feature extraction for ASR using the Aurora2 database," in *Proc. Eurospeech '01*, (Aalborg, Denmark), pp. 1607-1610, 2001.

## VITA

Jon Augustus Arrowood was born in 1970 in Oak Ridge, Tennessee. Through elementary, junior high, and high school, he had *many* excellent teachers, and it was because of them that he received a full scholarship for his undergraduate studies at Georgia Tech.

He remained at Georgia Tech for his Master's degree, spending one academic year at Georgia Tech's campus in Metz, France. While there, he learned a small amount about signal processing, and unfortunately too much about wine for his budget. It is also in France where he met Professor Mark Clements.

Returning to Atlanta for a Ph.D. degree advised by Prof. Clements, he had the opportunity to teach several undergraduate classes, guest lecture for graduate classes, perform research on a wide variety of topics, and also do some outside consulting on signal processing problems for various industry applications. The faculty and students of Georgia Tech's Center for Signal and Image Processing (CSIP) made for both an excellent environment to learn signal processing, and also a lively and entertaining place to work.

Jon's graduate research was on a novel method for incorporating information about the local time-frequency feature extraction reliability into a Hidden Markov Model (HMM) decoding algorithm. The method derived is shown to be the Bayes Predictive Classifier for uncertain feature data, and thus gives the optimal decoder given unreliable feature data. The algorithm is comparable in accuracy to HMM model compensation methods, but unlike those, can rapidly adapt to nonstationary noise environments. Although designed for speech recognition systems, this technique is applicable to a variety of other pattern recognition problems.

Outside of his professional activities, he enjoys running, reading, and football season, in reverse order.