

**AN ENSEMBLE SPEAKER AND SPEAKING ENVIRONMENT
MODELING APPROACH TO ROBUST SPEECH RECOGNITION**

A Dissertation
Presented to
The Academic Faculty

by

Yu Tsao

In Partial Fulfillment
of the Requirements for the Degree
Doctor of Philosophy
in the
School of Electrical and Computer Engineering



Georgia Institute of Technology
December, 2008

Copyright 2008 by Yu Tsao

**AN ENSEMBLE SPEAKER AND SPEAKING ENVIRONMENT
MODELING APPROACH TO ROBUST SPEECH RECOGNITION**

Approved by :

Dr. Chin-Hui Lee, Advisor
Professor, School of ECE
Georgia Institute of Technology

Dr. Anthony Joseph Yezzi
Professor, School of ECE
Georgia Institute of Technology

Dr. Mark Clements, Committee Chair
Professor, School of ECE
Georgia Institute of Technology

Dr. Biing-Hwang (Fred) Juang
Professor, School of ECE
Georgia Institute of Technology

Dr. Ming Yuan
Assistant Professor, School of ISYE
Georgia Institute of Technology

Date Approved: 11/04, 2008

Dedicated to

My Beloved Family, Chih-I Tsao, Chia-Chen Hsu, Hui Tsao, and Chih-Yung Huang

ACKNOWLEDGEMENTS

First and foremost, I would like to express my most sincere gratitude to my research advisor, Professor Chin-Hui Lee, for his guidance and support through my Ph.D. study. This work would not have been possible without his patience and encouragement that walked me through difficult times. His insights and suggestions directed my research deeply; his visionary thoughts and enthusiastic attitude about work influenced me greatly. It is my best fortune to have been working with him.

I would also like to thank Professors Biing-Hwang (Fred) Juang and Mark Clements. I have benefited a lot from them in a joint ASAT project. I also greatly appreciate Professors Anthony Joseph Yezzi and Ming Yuan for serving on my committee. I owe great acknowledgements to Dr. Kaisheng Yao, Dr. Vishu Viswanathan, Dr. Yifan Gong, Prof. Jen-Tzung Chien, Prof. Koichi Shinoda, Prof. Roger Drury, and Mrs. Gail Palmer, whose precious advices and instructions helped to shape my research and professional communication skills.

I would like to take this opportunity to express my gratitude to a number of people for their support over this Ph.D. study. I sincerely acknowledge my colleagues—Jinyu Li, Chengyuan Ma, Antonio Moreno, Jeremy Reed, Brett Matthews, Sabato Marco Siniscalchi, and friends in Atlanta—Chia-Hung Hou, Ying Hung, Ang Lee, Chen-Ju Lin, Tsung-Lin Wu, Sheng-Yu Peng, Richard Jen, Shen-Shen Lin, Shun-Chieh Chuang, Rung-Yu Tseng, Jen-Yu Yang, You-Chi Cheng, Shu-How Fan, Yu-Heng Lai, Chung-Ching Lin, Megan Tseng, Yi-Han Lin, Chien-I Lin, Bernie Jord Yang. Moreover, I gratefully thank my most important friends—Ming-Hsueh Tsai, Yen-Ting Chen, Shu-Fen Wei, Chung-Hsun Lin, and my best partners—Shu Du, Ping-Hsuan Hsieh.

Last, but certainly not least, I am deeply indebted to my parents, my sister, and my fiancée, Chih-Yung Huang, whose love and encouragements were unswerving. I owe the successful completion of my Ph.D. study to their continuously affectionate support and sacrifices. Now, I am looking forward to a long and joyful life with my family.

TABLE OF CONTENTS

ACKNOWLEDGEMENTS.....	iv
LIST OF TABLES.....	x
LIST OF FIGURES.....	xiii
SUMMARY.....	xv
<u>CHAPTER</u>	
1 SCIENTIFIC GOALS.....	1
2 BACKGROUND OVERVIEW.....	9
2.1 Background of Automatic Speech Recognition.....	9
2.1.1 Feature Extraction.....	11
2.1.2 Acoustic Model.....	11
2.1.3 Language Model.....	11
2.1.4 Decoder.....	12
2.2 Hidden Markov Model.....	12
2.2.1 Definition of HMM.....	13
2.2.2 Evaluation Problem of HMM.....	15
2.2.3 Decoding Problem of HMM.....	16
2.2.4 Learning Problem of HMM.....	17
2.3 Separation between Acoustic Models.....	19
2.3.1 Estimation of Model Separation.....	19
2.3.2 Characterization of Model Separation.....	20
2.3.3 Applications of Model Separation Measures.....	23
2.4 Mismatch Modeling.....	30

2.5 Noise Robustness Approaches	32
2.6 ML-Based Stochastic Matching Algorithm	37
3 ENSEMBLE SPEAKER AND SPEAKING ENVIRONMENT MODELING (ESSEM)	39
3.1 Direct ESSEM.....	40
3.2 Indirect ESSEM	41
3.3 Environment Structure Simulation	43
3.4 Experiments	45
3.5 Summary	50
4 OFFLINE ENVIRONMENT CONFIGURATION REFINEMENT	52
4.1 Improving Structure of the Environment Spaces.....	52
4.1.1 Environment Clustering (EC)	52
4.1.2 Environment Partitioning (EP)	54
4.2 Increasing Coverage of the Environment Spaces	56
4.2.1 Intra-Environment (IntraEnv) Training	57
4.2.2 Inter-Environment (InterEnv) Training	59
4.3 Experiments	61
4.3.1 Experimental Setup.....	61
4.3.2 EC on Framework-1(GI).....	65
4.3.3 EC on Framework-1(GD)	67
4.3.4 EC+EP on Framework-1(GI).....	68
4.3.5 IntraEnv and InterEnv Training on Framework-2(GI)	71
4.3.6 IntraEnv and InterEnv Training on Framework-2(GD).....	72

4.3.7 MLLR and MAPLR on Framework-2(GI)	73
4.3.8 SME-based IntraEnv Training on Framework-2(GI)	75
4.3.9 ESS Space Analysis: IntraEnv and InterEnv Training	76
4.3.10 Overall Combination: IntraEnv+InterEnv+EC+EP	78
4.4 Summary	80
5 ONLINE ESTIMATION PROCESS ENHANCEMENT	82
5.1 Improving Estimation Precision	82
5.1.1 Mapping Structure Precision.....	82
5.1.2 Multiple Cluster Matching (MCM) Precision.....	85
5.1.3 Weighted <i>N</i> -best Information	87
5.1.4 Cohort Selection Precision.....	88
5.1.5 Online Environment Space Adaptation	88
5.2 Enhancing Estimation Efficiency	89
5.3 Generalized ESSEM	90
5.4 Experiments	93
5.4.1 Online Mapping Structures on Framework-2(GD).....	94
5.4.2 Multiple Cluster Matching on Framework-2(GD).....	96
5.4.3 Weighted <i>N</i> -best Information on Framework-2(GD)	97
5.4.4 Cohort Selection and ESA on Framework-2(GD).....	99
5.4.5 Complex Online Mapping Structure on Framework-2(GD)	101
5.4.6 ESSEM with Best Offline and Online Configuration on Framework-2(GD)	102
5.4.7 EC, PCA, and HDR on Framework-1(GI)	102

5.4.8 Dimensionality Reduction on EC on Framework-1(GI)	104
5.4.9 Integration of EC and PCA (EC-PCA) on Framework-2 (GD)...	106
5.4.10 Generalized ESSEM on Framework-1 (GD)	107
5.5 Summary	110
6 CONCLUSION	112
APPENDIX A ASSOCIATION OF ESSEM AND EXEMPLAR THEORY.....	118
APPENDIX B DATABASES USED IN THIS THESIS	121
B.1 TIMIT.....	121
B.2 NOISEX-92.....	122
B.3 Aurora-2	123
REFERENCES	125
VITA.....	137

LIST OF TABLES

Table 2.1. Errors for two phone models /ay/ and /ix/ .	26
Table 3.1. WER (%) for baseline, MLLR, direct and indirect ESSEM in a supervised adaption mode.	48
Table 3.2. WER (%) for MLLR, direct and indirect ESSEM in an unsupervised adaption mode.	50
Table 4.1. WER (%) for baseline and ESSEM plus EC with different tree structure on Framework-1 (GI).	66
Table 4.2. WER (%) and P-value for two different EC ESS spaces on Framework-1 (GI).	67
Table 4.3. WER (%) for ESSEM using two EC ESS spaces on Framework-1 (GD).	68
Table 4.4. WER (%) and P-value for two different EC ESS spaces on Framework-1 (GD).	68
Table 4.5. WER (%) for ESSEM using EC+EP ESS spaces on Framework-1 (GD).	70
Table 4.6. WER (%) and P-value for two types of EP ESS spaces on Framework-1 (GD).	70
Table 4.7. WER (%) for ESSEM using the ESS spaces refined by intraEnv and interEnv training on Framework-2 (GI).	72
Table 4.8. WER (%) and P-value for intraEnv and interEnv training on Framework-2 (GI).	72
Table 4.9. WER (%) for ESSEM using the ESS spaces refined by intraEnv and interEnv training on Framework-2 (GD).	73
Table 4.10. WER (%) and P-value for intraEnv and interEnv training on Framework-2	

(GD).....	73
Table 4.11. WER (%) for MLLR and MAPLR on Framework-2 (GD).....	74
Table 4.12. WER (%) and P-value for MLLR and MAPLR.	75
Table 4.13. WER (%) for ESSEM using the ESS spaces refined by SME-based intraEnv and MCE-based interEnv training on Framework-2 (GD).	76
Table 4.14. WER (%) and P-value for MCE-based and SME-based intraEnv training on ESS spaces on Framework-2 (GD).	76
Table 4.15. Divergence distance within one set of environment-specific HMMs.....	77
Table 4.16. WER (%) for ESSEM with combined offline techniques on Framework-2 (GD).....	79
Table 4.17. WER (%) and P-value for ESSEM with combined offline techniques on Framework-2 (GD)	79
Table 5.1. WER (%) for ESSEM with different online mapping structures on Framework 2 (GD).....	94
Table 5.2. WER (%) and P-value for ESSEM with different online mapping structures on Framework-2 (GD).	94
Table 5.3. WER (%) for ESSEM with complex online mapping structures on Framework-2 (GD).	95
Table 5.4. WER (%) for ESSEM with MCM on Framework-2 (GD).....	96
Table 5.5. WER (%) and P-value for ESSEM with MCM on Framework-2 (GD).....	96
Table 5.6 WER (%) for ESSEM with weighted N -best information on Framework-2 (GD).	98
Table 5.7. WER (%) for ESSEM plus EC, cohort selection, and ESA with linear	

combination on Framework-2 (GD).	100
Table 5.8. WER (%) for ESSEM plus EC, cohort selection, and ESA with linear combination with bias on Framework-2 (GD).....	102
Table 5.9. ESSEM with the best offline and online configuration (in accuracy %).	102
Table 5.10. WER (%) for GESSEM with two-stage and combination methods.	109

LIST OF FIGURES

Figure 2.1. Architecture of an automatic speech recognition system	10
Figure 2.2. HMM with a left-to-right topology	13
Figure 2.3. An illustration of target and cohort models in the parametric space	21
Figure 2.4. An illustration of GLLR plot for pattern verification.....	23
Figure 2.5. Model separation and acoustic discrimination	25
Figure 2.6. Model separation and acoustic mismatches	27
Figure 2.7. Model separation and training criteria.....	28
Figure 2.8. Model separation and acoustic resolution	29
Figure 2.9. Speaker model separation and MFCC and pitch	30
Figure 2.10. Mismatch modeling and three classes of solutions	31
Figure 2.11. System architecture of the stochastic matching framework.....	38
Figure 3.1. Architecture of the ESSEM framework	39
Figure 3.2. System architecture of the direct ESSEM framework.....	41
Figure 3.3. System architecture of the indirect ESSEM framework.....	42
Figure 3.4. A model for speaking environment distortion.....	43
Figure 3.5. Simulation in the signal domain	44
Figure 4.1. Architecture of environment clustering algorithm.	53
Figure 4.2. Architecture of environment partitioning algorithm	56
Figure 4.3. Discriminative training to increase environment space discrimination	60
Figure 4.4. Separation between environment-space models.....	78
Figure 5.1. ESSEM with different environment spaces.....	103
Figure 5.2. PCA, HDR, H-PCA, cohort selection, on the EC-structured ESS space	105

Figure 5.3. Performance of EC-ESA with different dimensions in EC subspaces	107
Figure A.1. Prototype theory and exemplar theory.....	119

SUMMARY

In this study, an ensemble speaker and speaking environment modeling (ESSEM) approach is proposed to characterize environments in order to enhance performance robustness of automatic speech recognition (ASR) systems under adverse conditions. The ESSEM process comprises two stages, the offline and online phases. In the offline phase, we prepare an ensemble speaker and speaking environment space formed by a collection of super-vectors. Each super-vector consists of the entire set of means from all the Gaussian mixture components of a set of hidden Markov Models that characterizes a particular environment. In the online phase, with the ensemble environment space prepared in the offline phase, we estimate the super-vector for a new testing environment based on a stochastic matching criterion. A series of techniques is proposed to further improve the original ESSEM approach on both offline and online phases. For the offline phase, we focus on methods to enhance the construction and coverage of the environment space. We first demonstrate environment clustering and environment partitioning algorithms to well structure the environment space; then, we propose a discriminative training algorithm to enhance discrimination across environment super-vectors and therefore broaden the coverage of the ensemble environment space. For the online phase, we study methods to increase the efficiency and precision in estimating the target super-vector for the testing condition. To enhance the efficiency, we incorporate dimensionality reduction techniques to reduce the complexity of the original environment space. To improve the precision, we first study different forms of mapping function and propose a weighted N -best information technique; then, we propose cohort selection, environment space adaptation and multiple cluster matching algorithms to facilitate the

environment characterization. We evaluate the proposed ESSEM framework on the Aurora-2 connected digit recognition task. Experimental results verify that the original ESSEM approach already provides clear improvement over a baseline system without environment compensation. Moreover, the performance of ESSEM can be further enhanced by using the proposed offline and online algorithms. A significant improvement of 16.08% word error rate reduction is achieved by ESSEM with optimal offline and online configuration over our best baseline system on the Aurora-2 task.

CHAPTER 1

SCIENTIFIC GOALS

Automatic speech recognition (ASR) systems had been largely improved since hidden Markov model (HMM) was established as a fundamental tool to represent speech signals [1-4]. However, HMMs do not generalize well from the training to testing mismatch conditions. The sources of the mismatch may come from speaker variability and speaking environment distortions. The exact mismatch is usually an unknown combination of these sources. Although some parametric functions have been developed to well characterize particular distortions, the exact form of an unknown combination of multiple speaker and speaking environment distortions can be complex and hard to specify.

Many approaches have been proposed to deal with the mismatch issue. Among them, a category of approaches adjusts parameters of the original hidden Markov model (HMM) set to match the testing conditions. Maximum a posteriori (MAP) [5] and maximum likelihood linear regression (MLLR) [6] are two well-known and widely used approaches. More recently, some approaches prepare prior knowledge to facilitate the characterization of the unknown testing condition. The prior knowledge is usually obtained from multiple sets of hidden Markov models (HMM) prepared in the offline. The HMM sets are trained on the available training data. During testing, another transformation is estimated based on the prior knowledge to generate a new HMM set that matches the testing data. Examples include reference speaker weighting (RSW) [7], cluster adaptive training (CAT) [8], and eigenvoice [9]. Another category of approaches collect speech data from many different noisy conditions and structure these conditions into several noisy clusters [10, 11]. Each noisy cluster presents particular regional information of the entire collected

conditions. Before testing recognition, a noisy cluster is first located [10, 11]. The set of HMMs corresponding to the located cluster can thus be used to test recognition or further adapted to better match the testing utterances [11]. The advantage of noisy clustering is that the speech signals usually show similar characteristics under similar noisy conditions (similar noise types and SNRs). Using the regional information facilitates us to have a better preparation to deal with unknown noisy conditions.

In the mid-90s, a stochastic matching approach [12, 13] has been proposed to improve the ASR performance under mismatched conditions. The effects of individual or an unknown combination of speaker variability and environment distortions are characterized by a mapping structure. The set of parameters of the mapping structure, known as nuisance parameters, is estimated based on the testing utterances. More recently, we extended the original stochastic matching algorithm to including the abovementioned prior knowledge and regional information, and proposed an ensemble speaker and speaking environment modeling (ESSEM) approach [14-17]. In the proposed ESSEM approach, we model each environment of interest with a super-vector. Each super-vector consists of the entire set of mean vectors from all the Gaussian components of a set of HMMs that characterizes a particular environment.

The ESSEM framework comprises two stages, the offline and online phases. In the offline phase, ESSEM prepares an ensemble speaker and speaking environment space, denoted as ESS space for notational simplicity. The ESS space is formed by a collection of super-vectors obtained from a variety of speaker and speaking conditions. With the ESS space, ESSEM estimates a mapping function and uses it to obtain the acoustic models for the testing condition in the online phase.

For the offline phase, we discuss five issues to better prepare the ensemble speaker and speaking environment configuration: 1) coverage, 2) regional knowledge, 3) partition, 4) discrimination of each super-vector, 5) separation among different super-vectors used to characterize a particular environment. First for the coverage issue, we intend to build the ESS space with a good coverage of different acoustic environments. However, it may be too expansive to collect speech data from a wide range of real-world conditions. We handle this issue by artificially simulating the training data needed to model each ESS environment [14]. Therefore, we can obtain super-vectors for various combinations of multiple distortion sources. Next, we propose an environment clustering (EC) algorithm to prepare the regional information of the prior knowledge of the ESS space. We have compared the ESSEM performance using the ESS space with and without EC, and a PCA-built ESS space [16]. From the experimental results, we verified that by incorporating the regional information prepared by EC, ESSEM achieves better performance than both without EC and with PCA. The regional information is used in two aspects: 1) an environment cluster that has similar acoustic properties to the testing environment is selected (this cluster selection resembles subset selection [18-20] and provides a high resolution to model the testing condition); 2) an HMM set that represents the selected cluster of environments is used to collect statistics needed to estimate the mapping function (it is believed that the representative HMM set can provide more accurate statistical estimation than a set of environment independent (EI) HMMs). Next, we propose an environment portioning (EP) approach to prepare several sets of disjoint EP sub-spaces from the original ESS space. The parameters within one EP sub-space share a single mapping structure. We first define the partition criterion; then, we partition

the original super-vector into several sets (e.g., S sets) of sub-vectors; finally, we build S sets of EP sub-spaces from the original ESS space. Two types of EP techniques were implemented in our study, namely, mixture-based and feature-based EP [21-23]. The concept of EP is quite similar to the well-known piecewise-polynomial and spline functions [24] that approximate complicated functions with several local polynomial representations. In addition to EC and EP, we refine the ESS space by increasing the discrimination within each super-vector. We call this procedure intra-environment (intraEnv) training. We can use any discriminative training method for the intraEnv training. In this study, we used the minimum classification error (MCE) training [16, 25] and soft margin estimation (SME) [26, 27] for the intraEnv training [16, 25]. Finally, we increase the distance between each pair of super-vectors by inter-environment (interEnv) training. With interEnv training, the separation between each pair of super-vectors, as well as each pair of EC clusters, can be increased. We adopt MCE for interEnv training since the misclassification measure of MCE represents a probabilistic distance between two classes. In the implementation, all parameters in the ESS spaces are first estimated with the ML criterion. Then, a combination of intraEnv training followed by interEnv training is conducted to refine the ESS space.

We consider two issues for the online estimation process: 1) precision and 2) efficiency. To enhance the estimation precision, we can use more complex mapping functions to better characterize the unknown testing environments. However, too many free parameters to be estimated in a complex function may generate an over-fitting problem. As mentioned earlier, EP enables us to use several local polynomial functions to approximate the complex function associated with the operation conditions; meanwhile,

we can prepare an ESS space covering many different combinations of multiple distortion sources. Therefore, we focus on simple linear mapping function in this study. The most straightforward mapping function is the best first method that directly locates one specific super-vector from the ESS space. If the best-matched super-vector can not be well located, a linear combination may be a better choice. Furthermore, when a testing condition contains distortions that are not well-characterized in the training set, a linear combination with a correction bias can provide a further improvement over the linear combination alone [17]. Next, based on the study of ensemble classification method, or called mixture of experts [28], we propose a multiple cluster matching (MCM) algorithm to enhance the EC technique [17]. Moreover, we incorporate weighted information obtained from an N -best list, called weighted N -best information, into the ESSEM framework to enhance the ASR performance in an unsupervised adaption mode [29]. To characterize the testing conditions, we also introduce cohort selection and environment space adaptation to improve the resolution and coverage of the ESS spaces, respectively [29]. To enhance the efficiency of the online operation, we adopt two dimensionality reduction techniques, PCA and HDR [28], to reduce the complexity of the environment configuration of the ESSEM framework before stochastic matching. In this study, we also extend the original stochastic matching algorithm and propose a generalized stochastic matching (GSM) approach that well utilizes the N -best information to enhance the accuracy of model characterization. For the GSM approach, in addition to using frame likelihoods, we take into account other knowledge sources, such as confidence scores, and incorporate them into the objective function during optimization. We further derive a generalized ensemble speaker and speaking environment modeling (GESSEM)

framework based on the GSM approach. Based on the experimental results, we verified that GESSEM can provide further improvement over the original ESSEM in a supervised adaptation mode.

The main contribution of this thesis can be summarized in the following three points.

1) Instead of using a multi-style training style, ESSEM uses a single style training criterion. Multi-style training (also known as multicondition training) trains a set of acoustic models by using speech data from many different training conditions, while single-style training obtains the acoustic models by using training data from a particular condition. Therefore, multi-style trained acoustic models can cover many acoustic conditions, and single-style trained acoustic models focus on one single condition. In previous studies, multi-style trained speaker independent (SI) or environment independent (EI) acoustic models are usually used for ASR systems and show better performance robustness than that are trained on single-style clean condition training [30]. Moreover, speaker clustering [31, 32] and CAT [8] also use multi-style training to prepare multiple clusters as the prior knowledge to aid the online estimation of the target acoustic models.

For the proposed ESSEM framework, on the other hand, we prepare the ESS space by using multiple sets of acoustic models trained on single-style training. Each set of acoustic models characterizes a particular speaker and speaking environment. During testing, based on the available testing data, a mapping structure is calculated to flexibly model the unknown testing condition. When comparing to multi-style trained models, ESSEM maps models more focused based on speech data from a specific environment. Additionally, the ESS space in the ESSEM framework provides a prior knowledge to

facilitate online model characterization. With this prior knowledge, ESSEM can more efficiently characterize unknown testing conditions with simpler mapping functions when compared to traditional robust methods.

2) To overcome the difficulty of data collection, we artificially simulate the environment structure. An additional advantage of simulating the ESS space is—when the acoustic information for the testing condition is given in advance, we can construct a compact environment structure that provides high resolution to model the testing environment. When an ASR application specifies possible noise types along with SNR range before testing, a proper environment structure can be well constructed. On the other hand, when dealing with channel distortions, we can prepare an environment structure incorporating only channel characteristics. The factor of additive noise distortions does not need to be considered when building the environment structure for that particular application.

3) We propose online and offline algorithms to further improve the original ESSEM framework. Although the original ESSEM approach already provides significant improvement over the baseline, a series of offline and online techniques has been proposed to further enhance the ESSEM performance. With the offline techniques, we can obtain a well-structured ESS space with wide coverage of different speaker and speaking environments. With the well-prepared ESS space, ESSEM can use simple mapping functions, such as best first method, to estimate the super-vector for the testing condition. In this study, we also demonstrated other mapping structures that provide better performance when the best-matched super-vector can not be precisely located from the ESS space. In addition, we proposed online ESS space refinement techniques to cope with a possible mismatch between testing conditions and the prepared ESS space.

This dissertation is organized as follows. In Chapter 2, we first give a brief overview of ASR and review the background knowledge of HMM and model separation. We also introduce mismatch modeling and review existing robustness techniques. Chapter 3 details the main concept and implementation steps of two types of ESSEM approaches (direct and indirect ESSEM). Then, we introduce how to artificially sample the environment structure. Chapter 4 presents the research issues and possible solutions related to the offline process of the ESSEM framework. Chapter 5 discusses the online research issues and provides methods to handle these issues. Moreover, we present a generalized ensemble speaker and speaking environment modeling (GESSEM) framework. Chapter 6 concludes the study of this dissertation.

CHAPTER 2

BACKGROUND OVERVIEW

In this chapter, we first introduce the architecture of automatic speech recognition (ASR). Next, we detail the hidden Markov model (HMM) that is widely used in most current state-of-the-art ASR systems. After that, we present a probabilistic distance measure, generalized log-likelihood ratio (GLLR), to evaluate the separation between acoustic models. Then, we use a mismatch model to demonstrate the inevitable mismatch between training and testing conditions and give an overview of robustness techniques to reduce the mismatch. Finally, we introduce a stochastic matching algorithm that has been proven to give good performance robustness for ASR applications under adverse conditions.

2.1 Background of Automatic Speech Recognition

During the past decades, the performance of automatic speech recognition (ASR) systems has improved significantly [3, 33-35]. Deployment of ASR in mobile devices—such as personal data assistants and cell phones, as well as in client services, such as online-ticketing systems, and customer care management systems in call centers—has greatly facilitated human-machine interfaces in recent years.

The goal of ASR is to decode the incoming speech data into a sequence of words. The decoding problem is usually formulated as a maximum a posterior problem:

$$W' = \underset{W}{\operatorname{argmax}} P(W | F_Y) = \underset{W}{\operatorname{argmax}} \frac{P_{\Lambda_X}(F_Y | W) P_{\Gamma_X}(W)}{P(F_Y)}, \quad (2.1)$$

where $W = \{W_1, W_2, \dots, W_L\}$ is the recognized word sequence corresponding to incoming observation, F_Y . During optimization, we generally assume the incoming observation, F_Y , to be fixed; therefore, Eq-(2.1) becomes:

$$W' = \underset{W}{\operatorname{argmax}} p_{\Lambda_X}(F_Y | W, \Lambda_X) P_{\Gamma_X}(W). \quad (2.2)$$

We call $p_{\Lambda_X}(F_Y | W)$ and $P_{\Gamma_X}(W)$, respectively, acoustic model likelihood and language model probability; Λ_X and Γ_X are the acoustic model and language model parameters that characterize the probability density functions.

A typical ASR system comprises four major components—feature extraction, acoustic model, language model, and decoder. Figure 2.1 demonstrates an ASR system comprising these four components. The parameters of acoustic model (AM) and language (LM) are estimated in the offline stage based on the available acoustic and textual training data, respectively. During testing, the incoming utterances are first transformed to perceptually salient feature vectors. With AM and LM, a decoding unit generates the word sequence by the maximum a posteriori process based on the input feature vectors. A more detailed introduction of these four components will be stated in the following.

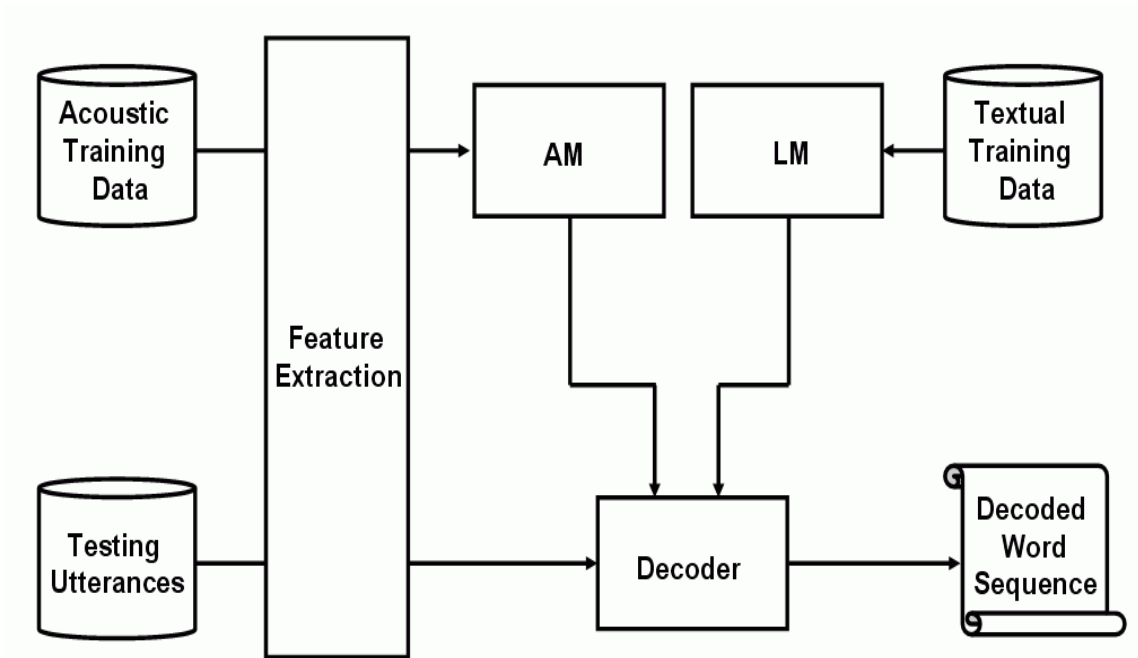


Figure 2.1. Architecture of an automatic speech recognition system.

2.1.1 Feature Extraction

Feature extraction processes signal enhancement on the incoming speech data and transforms the enhanced speech signals into compact and perceptually meaningful feature vectors. Mel-frequency cepstrum coefficient (MFCC) [36] and perceptual linear prediction (PLP) [37] are two well-known acoustic features that have been adopted successfully in many ASR systems. In addition to MFCC and PLP, some acoustic features were developed to specially increase the robustness over environment changes. Examples of these robust feature extraction techniques will be introduced in Section 2.5.

2.1.2 Acoustic Model

Acoustic model characterizes the likelihood of acoustic features with respect to the word sequence. To design acoustic model, we first determine its structure to well characterize speech signals based on the available training data. Since mid-70, HMM has been widely adopted as a basic structure of acoustic model in most ASR systems. The success of HMM lies in its well characterizing the time-variant property of speech signals. For different applications, we can choose discrete HMMs [38], semi-continuous HMMs [39], or continuous HMMs [40]. When a structure is defined, the parameters of Λ_X are calculated from the training data. This set of parameters, Λ_X , determines the likelihood, $p_{\Lambda_X}(F_Y | W)$ in Eq-(2.1), and presents information about acoustics, phoneme, speaker variability and speaking environments.

2.1.3 Language Model

Language model determines the probability of a possible word sequence. In small vocabulary ASR systems, such as command-and-response applications, language model gives limited information in the decoding process. For large vocabulary systems, on the

other hand, language model is heavily required during recognition. Among language model technologies, an N-gram method [33, 34] is used in most current state-of-the-art large vocabulary continuous speech recognition (LVCSR) systems. In such LVCSR systems, a textual corpus with a large amount of text knowledge is prepared in the offline stage and used to train the N-gram language model.

2.1.4 Decoder

Decoder searches a word sequence that gives highest posterior probability corresponding to the acoustic and language models. A good search algorithm considers all possible knowledge sources while maintaining a favorable run-time speed. Viterbi decoding [41] and A* search [42] are two well known and widely used methods. To further increase the speed, some approaches limit the knowledge sources when processing search tasks. Examples include fast match [33], look-ahead strategy [43], fast likelihood computation [44], different layers of beam search [45], and lexical tree optimization [46].

In this study, we use the HMM for the acoustic model and attempt to improve the ASR performance robustness by adjusting parameters in the HMM set to match various adverse environments. In the next section, we review the definition and three fundamental problems of HMM.

2.2 Hidden Markov Model

In this section, we detail the definition and three fundamental problems of HMM. Originated from the Markov chain models, HMM has a finite state process with transition between states specified by the probability function $P(s|s')$. Different from the Markov chain model that each state corresponds to a deterministic observations, HMM uses a non-deterministic process that can output observations at any state.

2.2.1 Definition of HMM

We use one set of HMMs to characterize a basic acoustic unit, e.g., phone, triphone, whole word. The definition of the basic unit depends on the requirement of recognition accuracy and computation efficiency of ASR applications. Generally, a left-to-right state sequence topology is used in HMM as shown in Figure 2.2. In Figure 2.2, we use three active states to model the speech unit. With the given speech signals, we actually can not observe a definite state sequence. In other words, the state sequence corresponding to the speech observations is hidden. Therefore, we place a “hidden” in front of the Markov models. Although the state sequence is not observable, each state usually represents salient information about the speech signals. To have a finer characterization of speech unit, we can use more states in one set of HMMs. However, with too many states in one set of HMMs, the requirement of memory storage will be high and the online computation will be intense during decoding.

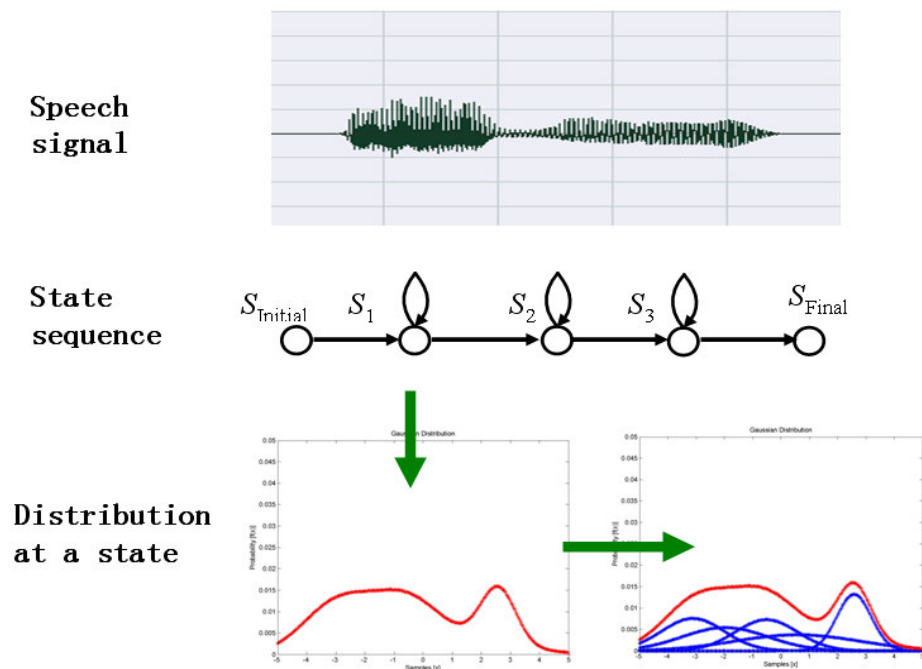


Figure 2.2. HMM with a left-to-right topology.

As shown in Figure 2.2, the characterization of each state is modeled by a probabilistic function, which is modeled by a Gaussian mixture model (GMM). We call the type of HMM illustrated in Figure 2.2 the continuous HMM [40]. As mentioned earlier, we can also use discrete HMM [38] and semi-continuous HMM [39]. In this thesis, we focus on the continuous HMM.

One set of HMM contains the following parameters:

- N —number of state in the HMM. Although the state sequence is hidden, it may represent physical information of speech segments. We denote the individual state by $\{1, 2, \dots, N\}$, and specify the state at time t by s_t .
- M —number of distinct symbol in one state. The symbols correspond to the physical output of the system being modeled. Here, we define the output symbol as $U = \{u_1, u_2, \dots, u_M\}$.
- $A = \{a_{i,j}\}$ —transition probability matrix. $a_{i,j}$ is the probability of a transition from state i to state j , where $1 \leq i, j \leq N$. The matrix A is an N by N square matrix.
- $B = \{b_j(k)\}$ —observation probability matrix, where $b_j(k)$ is the observation probability of symbol u_k in the j -th state. With a set of observed output, $\mathbf{X} = \{X_1, X_2, \dots, X_T\}$, we have :

$$b_j(k) = P(X_t = u_k | s_t = j). \quad (2.3)$$

- $\pi = \{\pi_i\}$ —initial state distribution where:

$$\pi_i = P(s_0 = i), 1 \leq i \leq N. \quad (2.4)$$

A complete set of HMM specification includes the above parameters, and we denote these parameter set by $\Lambda = (A, B, \pi)$. In the following discussions, we review three fundamental problems of HMM, i.e., the evaluation, decoding and learning problems.

2.2.2 Evaluation Problem of HMM

The first problem is to evaluate the probability $P(\mathbf{X}|\Lambda)$ of the observation sequence, $\mathbf{X}=(X_1, X_2, \dots, X_T)$, given a set of HMM, Λ . The most straightforward way is to sum up the probabilities by enumerating every possible state sequence of length T :

$$P(\mathbf{X}|\Lambda) = \sum_{\text{all } S} P(\mathbf{X}|S, \Lambda)P(S|\Lambda), \quad (2.5)$$

where S is a particular state sequence of length T , $S=(s_1, s_2, \dots, s_T)$. $P(\mathbf{X}|S, \Lambda)$ is :

$$P(\mathbf{X}|S, \Lambda) = \prod_{t=1}^T P(X_t|s_t, \Lambda) = b_{s_1}(X_1)b_{s_2}(X_2)\dots b_{s_T}(X_T), \quad (2.6)$$

and $P(S|\Lambda)$ can be represented by:

$$P(S|\Lambda) = P(s_1|\Lambda) \prod_{t=2}^T P(s_t|s_{t-1}, \Lambda) = \pi_{s_1} a_{s_1 s_2} \dots a_{s_{T-1} s_T}. \quad (2.7)$$

From Eqs-(2.6) and (2.7), we can have:

$$P(\mathbf{X}|\Lambda) = \sum_{s_1, s_2, \dots, s_T} \pi_{s_1} b_{s_1}(X_1) a_{s_1 s_2} b_{s_2}(X_2) \dots a_{s_{T-1} s_T} b_{s_T}(X_T). \quad (2.8)$$

A direct evaluation of Eq-(2.8) demands an extremely high computation. Fortunately, an advanced algorithm can be used to efficiently compute Eq-(2.8), which is known as the forward algorithm. For the forward, we first define a forward variable:

$$\alpha_t(i) = P(X_1, X_2, \dots, X_t, s_t=i|\Lambda), \quad (2.9)$$

where $\alpha_t(i)$ is the probability of the partial observation (X_1, X_2, \dots, X_t) at time t in the i -th state. The forward algorithm includes three steps:

- Initialization:

$$\alpha_1(i) = \pi_i b_i(X_1), \quad 1 \leq i \leq N. \quad (2.10)$$

- Induction:

$$\alpha_t(j) = \left[\sum_{i=1}^N \alpha_{t-1}(i) a_{ij} \right] b_j(X_t), \quad 2 \leq t \leq T; \quad 1 \leq i \leq N. \quad (2.11)$$

• Termination:

$$P(\mathbf{X} | \Lambda) = \sum_{i=1}^N \alpha_T(i), \quad (2.12)$$

where

$$\alpha_T(i) = P(X_1, X_2, \dots, X_T, s_T = i | \Lambda). \quad (2.13)$$

In a similar manner, a backward procedure is derived to evaluate the probability of HMM. First, we consider a backward variable $\beta_t(i) = P(X_{t+1}, X_{t+2}, \dots, X_T | s_t = i, \Lambda)$, and we use the following procedure:

• Initialization:

$$\beta_T(i) = 1, \quad 1 \leq i \leq N. \quad (2.14)$$

• Induction:

$$\beta_t(i) = \sum_{j=1}^N a_{ij} b_j(X_{t+1}) \beta_{t+1}(j), \quad t = T-1, T-2, \dots, 1, \quad 1 \leq i \leq N. \quad (2.15)$$

More details about the forward and backward algorithms are referred to [33, 34].

2.2.3 Decoding Problem of HMM

In this study, we use the Viterbi decoding [41] to find the best state sequence, $S = (s_1, s_2, \dots, s_T)$, that maximizes $P(\mathbf{X} | S, \Lambda)$. The Viterbi algorithm can be seen as a modified forward algorithm. Instead of summing up probabilities from different paths coming to the same destination state, the Viterbi algorithm keeps the best path. The Viterbi algorithm first defines a best-path probability:

$$V_t(i) = \max_{s_1, s_2, \dots, s_{t-1}} P(X_1, X_2, \dots, X_t, s_1, s_2, \dots, s_{t-1}, s_t = i | \Lambda). \quad (2.16)$$

$V_t(i)$ is the probability of the most likely state sequence at time t , which has generated the

observation (X_1, X_2, \dots, X_t) and ends in the i -th state. Next, an array $\psi_t(j)$ is used to keep track of the argument for each t and j . Then, the Viterbi algorithm uses the following procedure to find the best state sequence:

- Initialization:

$$V_1(i) = \pi_i b_i(X_1), 1 \leq i \leq N, \quad (2.17)$$

$$\psi_1(i) = 0. \quad (2.18)$$

- Recursion:

$$V_t(j) = \max_{1 \leq i \leq N} [V_{t-1}(i) a_{ij}] b_j(X_t), \quad 2 \leq t \leq T, 1 \leq j \leq N, \quad (2.19)$$

$$\psi_t(j) = \arg \max_{1 \leq i \leq N} [V_{t-1}(i) a_{ij}], \quad 2 \leq t \leq T, 1 \leq j \leq N. \quad (2.20)$$

- Termination:

$$P^* = \max_{1 \leq i \leq N} [V_T(i)], \quad (2.21)$$

$$s_T^* = \arg \max_{1 \leq i \leq N} [V_T(i)]. \quad (2.22)$$

where P^* and s_T^* are the best score and the final state.

- Backtracking:

$$s_t^* = \psi_{t+1}(s_{t+1}^*), t=T-1, T-2, \dots, 1, \quad (2.23)$$

$$S^* = (s_1^*, s_2^*, \dots, s_T^*), \quad (2.24)$$

where S^* is the best state sequence.

2.2.4 Learning Problem of HMM

The learning process is to find the HMM parameter $\Lambda = (A, B, \pi)$ that maximize the probability $P(\mathbf{X}|\Lambda)$ of observation \mathbf{X} . Because no close-form solution is available, an expectation maximization (EM) algorithm is used to estimate HMM parameters. We use Λ and $\hat{\Lambda}$ to denote the original and new HMM parameters, respectively, and define an auxiliary function Q as:

$$Q(\Lambda, \hat{\Lambda}) = Q_\pi(\Lambda, \hat{\pi}) + \sum_{i=1}^N Q_{a_i}(\Lambda, \hat{\mathbf{a}}_i) + \sum_{j=1}^N Q_{b_j}(\Lambda, \hat{\mathbf{b}}_j), \quad (2.25)$$

where:

$$Q_\pi(\Lambda, \hat{\pi}) = \sum_{i=1}^N P(\mathbf{X}, s_0 = i | \Lambda) \log \hat{\pi}_i, \quad (2.26)$$

$$Q_{a_i}(\Lambda, \hat{\mathbf{a}}_i) = \sum_{j=1}^N \sum_{t=1}^T P(\mathbf{X}, s_{t-1} = i, s_t = j | \Lambda) \log \hat{a}_{ij}, \quad (2.27)$$

$$Q_{b_j}(\Lambda, \hat{\mathbf{b}}_j) = \sum_{t=1}^T P(\mathbf{X}, s_t = j | \Lambda) \log \hat{b}_j(k). \quad (2.28)$$

From Eq-(2.25), we can see the Q -function is now separated into three terms, the maximization procedure on $Q(\Lambda, \hat{\Lambda})$ can be done maximizing the individual terms separately. After some derivations, we can obtain the model estimate by the following three equations:

$$\pi_i = \frac{P(\mathbf{X}, s_0 = i | \Lambda)}{P(\mathbf{X} | \Lambda)}, \quad (2.29)$$

$$\hat{a}_{ij} = \frac{\sum_{t=1}^T P(\mathbf{X}, s_{t-1} = i, s_t = j | \Lambda)}{\sum_{t=1}^T P(\mathbf{X}, s_{t-1} = i | \Lambda)} = \frac{\sum_{t=1}^T \xi_t(i, j)}{\sum_{t=1}^T \sum_{k=1}^N \xi_t(i, k)}, \quad (2.30)$$

$$\hat{b}_j(k) = \frac{\sum_{t=1}^T P(\mathbf{X}, s_t = j | \Lambda) \delta(X_t, u_k)}{\sum_{t=1}^T P(\mathbf{X}, s_t = j | \Lambda)} = \frac{\sum_{t=1; s.t. X_t=u_k}^T \xi_t(i, j)}{\sum_{t=1}^T \sum_{k=1}^N \xi_t(i, k)}, \quad (2.31)$$

$$\xi_t(i, j) = P(s_{t-1} = i, s_t = j | \mathbf{X}, \Lambda) = \frac{\alpha_{t-1}(i) a_{ij} b_j(X_t) \beta_t(j)}{\sum_{k=1}^N \alpha_t(k)}, \quad (2.32)$$

where $\beta_t(j)$ is the backward probability as defined in Eq-(2.15).

With the new parameters, set $\Lambda = \hat{\Lambda}$, and repeat from Eq-(2.25) for several iterations until convergence.

2.3 Separation between Acoustic Models

A good measure of separation usually serves as a key indicator of the discrimination power of these speech models because it can often be used to indirectly determine the performance of speech recognition and verification systems. In this study, we introduce a probabilistic distance, called generalized log likelihood ratio (GLLR), to measure the separation between a model of a target speech attribute and models of its competing attributes. We illustrate five applications to compare separations among models obtained over multiple levels of discrimination capabilities, at various degrees of acoustic definitions and resolutions, under mismatched training and testing conditions, and with different training criteria and speech parameters. We demonstrate that the well-known GLLR distance and its corresponding histograms also provide a good utility to qualitatively and quantitatively characterize the properties of trained models without performing large scale speech recognition and verification experiments.

2.3.1 Estimation of Model Separation

In real-world pattern matching problems, such as automatic speech recognition (ASR) [4] and utterance verification (UV) [47], the true distributions of the patterns to be matched are often not precisely known. Thus, the performance of such systems are usually determined by running experiments over a representative collection of evaluation samples intending to cover all possible variations of testing conditions using models created in a separate training phase. In many cases, such an endeavor can be very challenging, if not impossible, in order to collect a large enough testing set that will produce statistically significant results. We are therefore interested in developing techniques that can be used to estimate the performance and behavior of real-world systems without conducting large

scale experiments. Intuitively, the separation between competing models in the same system serves as an important indicator to accomplish such purposes. For example, model-based error estimation algorithms have been shown capable of predicting ASR performance [48].

Learning from MCE [25] and minimum verification error (MVE) [49] training formulations, the misclassification measure provides a quantitative indicator to represent a distance between a target model and its competing models. It can be used to measure the model separation as well. MCE and MVE can then be considered as a way to find model parameters that enhances the overall separation of the collection of models. A closer look at the misclassification measure reveals that it can also be considered as a probabilistic distance, called *generalized log likelihood ratio* (GLLR), commonly used in statistical hypothesis testing [50], if a log likelihood function is used to compute the class discriminant function [51]. GLLR also plays a key role in evaluating speech attribute detectors in a new speech research paradigm we are currently exploring under the ASAT (automatic speech attribute transcription) project [52]. In this study, we illustrate a number of applications of the GLLR measure, and demonstrate that GLLR provides a good utility to characterize the discrimination capabilities of trained models without running large scale ASR and UV experiments.

2.3.2 Characterization of Model Separation

We now discuss issues related to computing GLLR measures and show that the corresponding histograms obtained from the sample GLLR values of target and non-target sets serve as useful tools to visually analyze model separation, and predict system performance for many ASR and UV tasks.

A. Defining Target and Competing Sets

In pattern verification [53] of a signal X , we first define a null hypothesis, H_0 , and an alternative hypothesis, H_1 , with H_0 : $\{X \text{ is generated from } S_0\}$ versus H_1 : $\{X \text{ is generated from any source but } S_0\}$. A statistical test is then designed to divide the signal space S_X into two complimentary regions such that we reject hypothesis H_0 , if $X \notin S_0$, and accept H_0 , if $X \in S_0$. In speech problems, H_1 is usually a composite hypothesis consisting of many signal classes. It has been shown that only the most competitive classes to H_0 need to be considered. This is usually accomplished by finding a speaker or phone “cohort” set [54]. In this study, the cohort set is determined by selecting models that obtained the highest likelihood values when evaluating training data from the target class. Figure 2.3 illustrates target and cohort models in a parametric space.

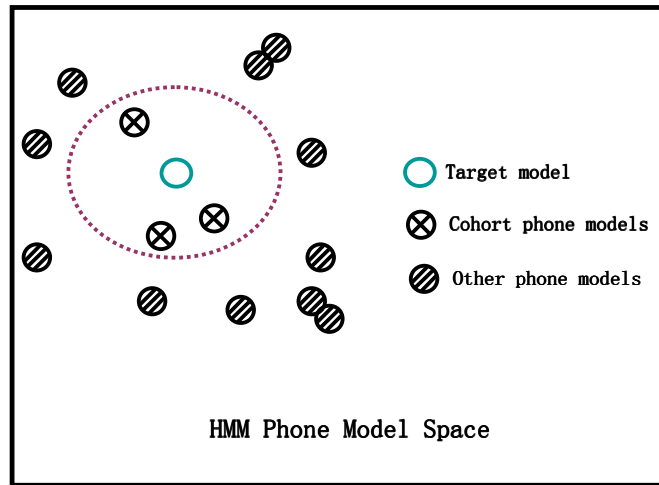


Figure 2.3. An illustration of target and cohort models in the parametric space.

B. Computing Target and Competing Scores

The LLR measure used in verification problems is defined as:

$$T(X | \lambda_0, \lambda_1) = \log[\ell(X | \lambda_0)] - \log[\ell(X | \lambda_1)], \quad (2.33)$$

where λ_0 and λ_1 are the parameters for the target model and non-target model, with $\log[\ell(X|\lambda_0)]$ and $\log[\ell(X|\lambda_1)]$ representing the target and competing scores, respectively. When we use a cohort set for the target to calculate the non-target score generated by multiple competing models, the modified LLR score in Eq-(2.34) is called a generalized log likelihood ratio (GLLR) computed as follows:

$$T(X|\lambda_q, \bar{\Lambda}_q) = \log[\ell(X|\lambda_q)] - \log[f(X|\bar{\Lambda}_q)], \quad (2.34)$$

where λ_q is a model for the target q , and $\bar{\Lambda}_q$ represents the set of competing models. The second term in the right hand side of Eq-(2.34) is an L_η of the scores in the cohort set C_q with size $|C_q|$ of the claimed target q , commonly used in MCE [25]:

$$f(X|\bar{\Lambda}_q) = \{|C_q|^{-1} \sum_r \exp[\eta \log \ell(X|\lambda_r)]\}^{1/\eta} \quad . \quad (2.35)$$

C. Preparing Competing GLLR Histograms

Based on the GLLR scores evaluated on samples of target and non-target segments in a set of speech utterances, a pair of GLLR histograms can be obtained with Eqs-(2.34) and (2.35). Figure 2.4 is an example of a typical GLLR plot with the right distribution (or histogram) curve representing the samples from the target source ($X \in S_0$), and the left curve depicting the sample distribution of the non-target source ($X \notin S_0$). The shaded region to the left of the vertical threshold line under the target curve gives the Type I error which is target samples missed. On the other hand, the shaded region to the right under the non-target curve represents the false alarms in detection. The smaller the regions the less the errors will be. Therefore, the performance of verification or recognition systems with the given models can be predicted. It is clear that the GLLR plot

can be generated for any verification problems we are interested in ASR and UV.

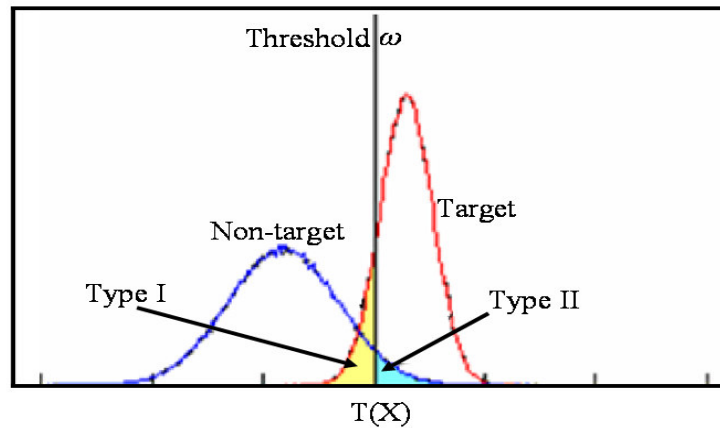


Figure 2.4. An illustration of GLLR plot for pattern verification.

D. GLLR as a Measure of Separation

It is noted that GLLR is also a good measure for estimating the separation between a target and its competing cohort models. Therefore, it is easy to visually analyze the separation between two sets of models by examining the GLLR plots. New training algorithms can be developed to move the target and non-target curves. The effectiveness of different speech parameters, speech attributes, or model resolutions can be evaluated by comparing the overlap regions for each case. By moving the right curve to the right, or the left curve to the left or both, it is clear that it results in more separation between the two sets of competing models. It also indicates reduced Type I and Type II errors. Since minimizing errors and maximizing the model separation are closely related, it is clear to see why MCE and MVE algorithms have been shown very effective in many ASR and UV applications.

2.3.3 Applications of Model Separation Measures

In this following, we illustrate five applications of GLLR to compare separation among

models obtained over multiple levels of discrimination capabilities, at various levels of acoustic definitions and resolutions, under mismatched training and testing conditions, and with different training criteria and speech parameters. We show that the GLLR separation measures and their corresponding histograms are good utilities to quantitatively and qualitatively study the properties of trained models without carrying out an extensive set of ASR and UV experiments.

In all the following experiments, both TIMIT and NTIMIT (Network TIMIT) databases [55] are used. Data in TIMIT were recorded with high-quality desktop microphones in a clean environment at a 16 KHz sampling rate. More details about TIMIT can be found in Appendix B. Excluding the speech materials reserved for speaker adaptation, there are 3696 and 1344 utterances in the standard training and testing sets, respectively. The NTIMIT data were obtained by passing the TIMIT version over dial-up lines, intending to simulate channel and noise distortion over the telephone network.

We used the entire training sets in the TIMIT corpus to train HMMs for phones and speech attributes. All HMMs were either related to a set of 45 English phones or another set of five manners of articulation, namely vowel, fricative, stop, nasal and approximant [56], plus silence. Almost all models have 3 states with each state characterized by eight Gaussian mixture components. In most cases, we used a feature vector of 39 elements, consisted of 13 MFCC parameters plus their first and second time derivatives.

A. Model Separation and Acoustic Discrimination

First, we are interested in any correlation between model separation and acoustic discrimination capabilities. Two vowels, /ix/ (in doing) and /ay/ (in hiking), were chosen for illustration. We used the five most competitive phones for /ay/, namely {/ah/, /aa/, /ae/,

/eh/, /ao/} obtained from recognition results over the training set, to form its corresponding cohort set. Similarly, the five most competitive phones, {/ih/ (in bit), /ax/, /eh/, /uw/, /uh/}, to /ix/ were used to build the cohort set for /ix/. Based on some phonetic knowledge, the diphthong /ay/ is usually considered easier to recognize than /ix/, so the separation of /ay/ from other competing sounds is expected be larger than that of the phone /ix/ from its competing sounds. Figure 2.5 validates our assumption. It is seen that the overlap region in the top panel for /ix/ is clearly larger than that in the bottom panel for /ay/. This utility can be used to compare the degree of difficulty in recognizing and verifying different phones. We can also use the cohort set for each phone to evaluate the confusability of competing words in an ASR vocabulary, and try to avoid confusable pairs as much as possible in vocabulary design.

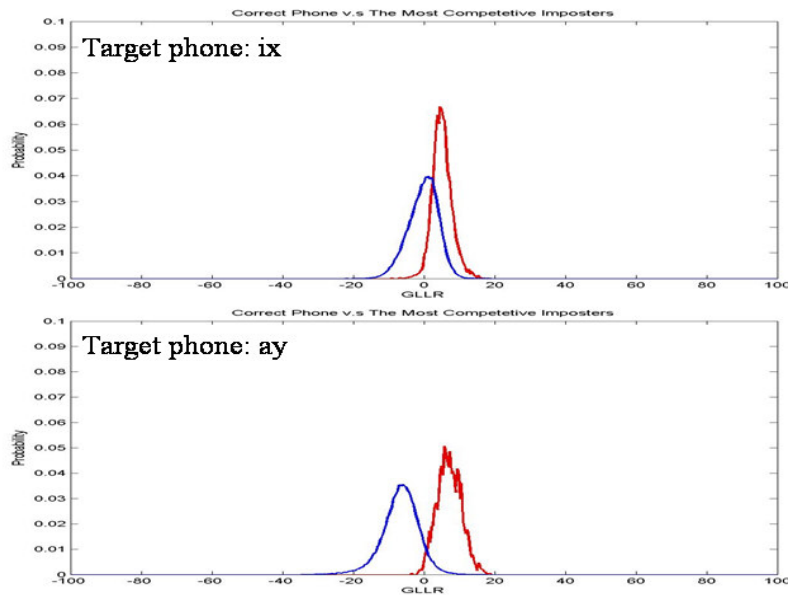


Figure 2.5. Model separation and acoustic discrimination.

Another way to examine the properties of the model separation measure is to list recognition errors as shown in Table 2.1. Since Figure 2.5 indicates that there are much

more Type II errors for phone /ix/ when compared to phone /ay/, we predict that sound /ix/ is easier than sound /ay/ to be substituted by other competitive sounds. The results from Table 2.1 confirm the information displayed in Figure 2.5.

Table 2.1. Errors for two phone models /ay/ and /ix/.

Phone model	/ay/	/ix/
Correct	77.37%	40.11%
Substitution	18.64%	41.96%
Deletion	3.99%	17.93%
Insertion	6.07%	3.84%

B. Model Separation and Acoustic Mismatch

Next, we are interested in comparing model separation in mismatched conditions. Models built from the noise and channel distorted data in the NTIMIT database were used for comparison. Since the spectral content in the higher frequency bands have been removed in the telephone data, it is expected that the discrimination among fricative sounds is likely to be seriously degraded, more than the vowel sounds.

In Figure 2.6, we compare vowel /iy/ (in sheet) with fricative /sh/ (in sheet). All the training data were from the TIMIT database. The two plots in the top panels display results for matched testing conditions. They clearly show that the fricative /sh/ is easier to recognize than the vowel /iy/. When the testing data were from the mismatched NTIMIT databases, it is noted that the overlap regions to discriminate /sh/ is significantly increased in the bottom right panel, while the increase for /iy/ in the bottom left panel was not as serious. This validates our assumptions that for phone /sh/, the separation between the target and its competing models will be significantly reduced in a

mismatched environment, and it is believed that the recognition performance will also be greatly degraded. On the other hand, the separation for the vowel phone /iy/ does not change as much in mismatched conditions.

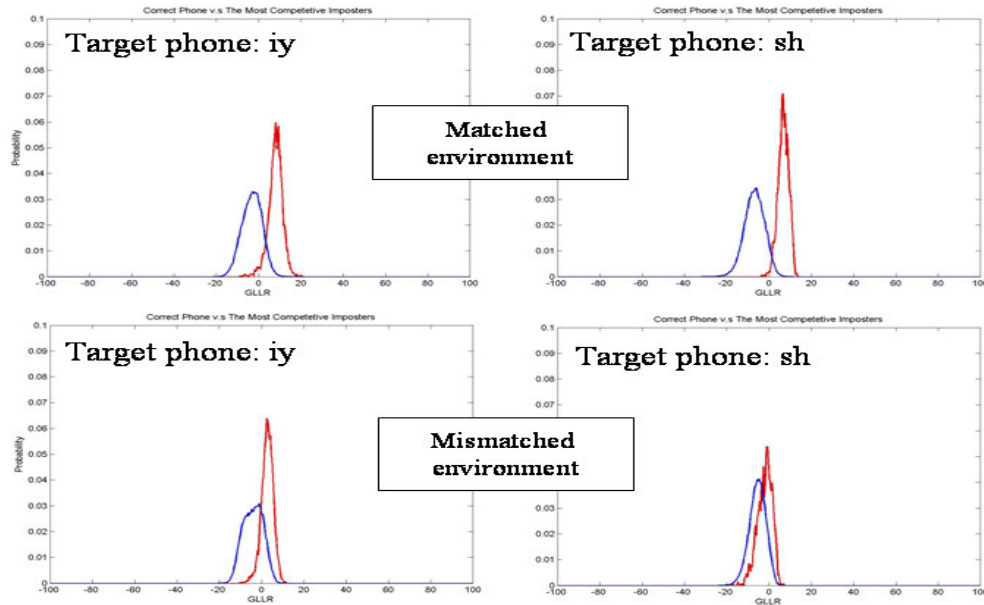


Figure 2.6. Model separation and acoustic mismatches.

Again, we find the GLLR plots of models serve as a good utility to observe model behavior of unseen data by simulating adverse conditions. New compensation algorithms can also be developed to enhance model separation using this utility [47].

C. Model Separation and Training Criteria

It is well-known that a set of good models will usually provide a good performance improvement. This improvement can easily be observed using the GLLR utility with running large scale recognition experiments. For example, when comparing the conventional maximum likelihood (ML) trained with MCE learned models, we always plot the GLLR statistics before and after MCE training to illustrate the concept of separation enhancement. Here, we illustrate this by using a context independent /Vowel/

manner HMMs. In Figure 2.7, it is clearly shown that the MCE-trained model enhances the separation with its competing models. It is recommended that such GLLR plots are used to compare models trained in various conditions with different optimization criteria.

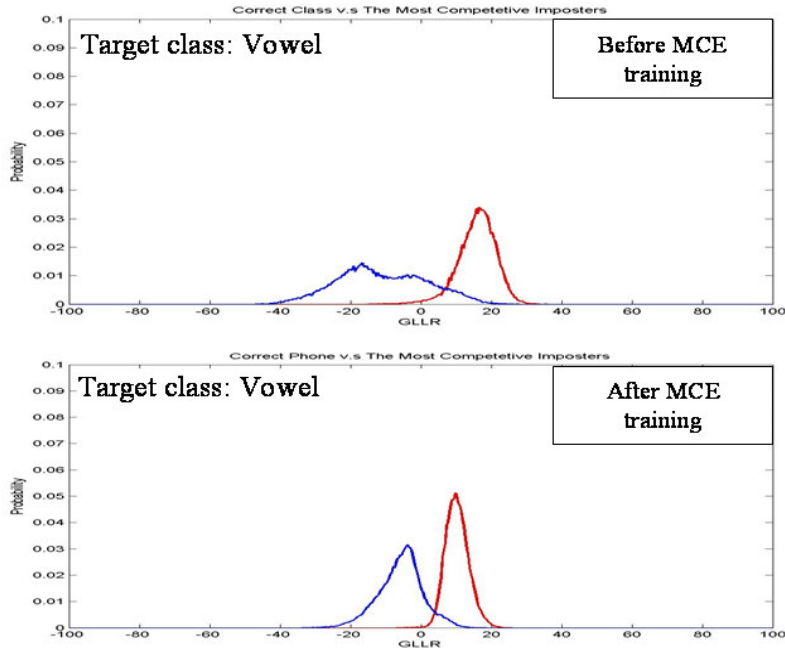


Figure 2.7. Model separation and training criteria.

D. Model Separation and Acoustic Resolution

Intuitively, a model with a better acoustic resolution will give more separation than models with less detailed description. This can be demonstrated using the GLLR utility to compare context independent (CI) and context dependent (CD) models. Here we used manner attribute models. Our recognition results showed that CD class models reduced the overall class error rate by 18.23% (from 28.91% to 23.64%) when compared with CI class models. In Figure 2.8, we compared CI /Vowel/ class model with CD /Fricative-Vowel+Stop/ model. It can be seen that the separation is enhanced with models with a better acoustic resolution, which resulted in a reduction of both Type I and Type II errors.

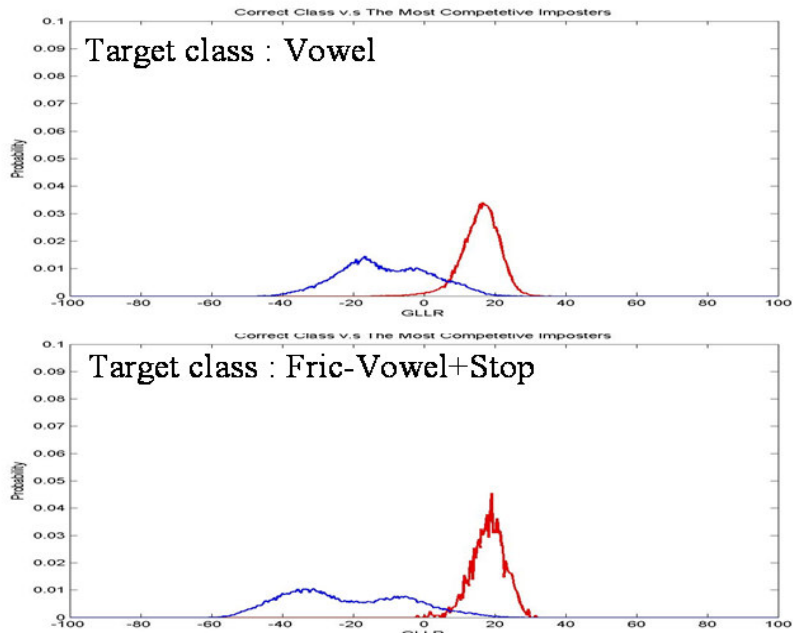


Figure 2.8. Model separation and acoustic resolution.

E. Model Separation and Speech Parameter Selection

The same GLLR utility can also be used to compare detectors using different speech parameters. It is well known that some speech parameters are more discriminative in detecting certain speech attributes. A single voice onset time (VOT) parameter was shown to give better detection results than those produced with 39 MFCC parameters in differentiating voiced against unvoiced stop sounds [56]. This property can be clearly illustrated by plotting the GLLR histograms to compare the model separation induced by the two sets of detectors using different speech parameters. In Figure 2.9 (adopted from [57]), for comparing speaker verification parameters, we plot two sets of GLLR histograms for one speaker to show that a single pitch parameter gives a smaller overlapping region shown in the bottom panel than that obtained with 39 MFCC parameters in the top panel, similar to the above VOT case for ASR.

Although new speech parameters may not give a significant word error reduction in a complex large vocabulary continuous speech recognition task, the GLLR measure is still a useful tool to evaluate these speech parameters in a well-controlled testing environment in order to demonstrate its utility in discriminating special classes of sounds.

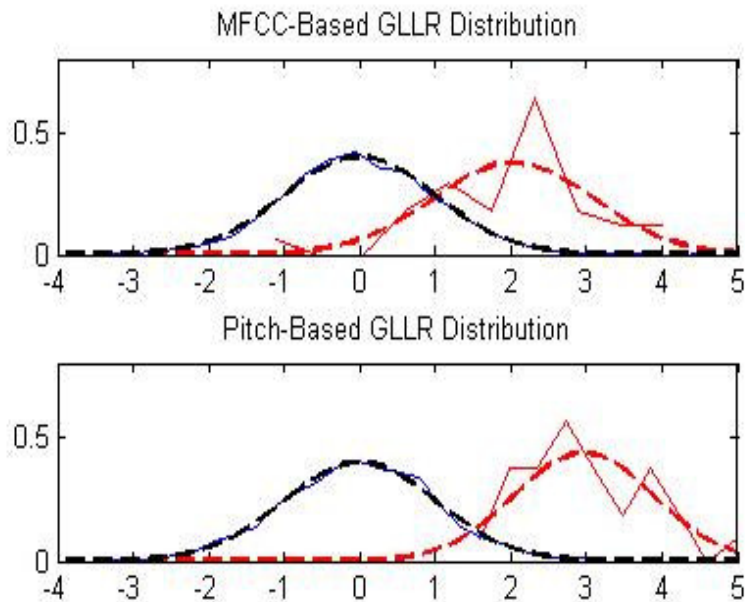


Figure 2.9. Speaker model separation and MFCC and pitch.

2.4 Mismatch Modeling

In the previous sections, we review fundamental knowledge of ASR and study separation between acoustic models. In the following sections, we will state the objective of this study—improving ASR performance robustness under distorted conditions.

The applicability of HMM-based ASR is limited due to one critical issue: data-driven HMM-trained acoustic models do not generalize well from training to testing conditions. Such an inevitable mismatch is generally derived from: 1) speaker effects, e.g., accent, dialect, and speaking rate differences; and 2) speaking environment effects,

e.g., interfering noise, transducers, and channel distortions. Although some functions can model particular distortion sources well, the form of an unknown combination of speaker and environment distortions is often unavailable or cannot be exactly specified.

The mismatch between training and testing conditions can be viewed in the signal, feature or model space, as illustrated in Figure 2.10 [12, 13]. First, in the signal space, S_X and S_Y denote the speech signals in the training and testing conditions, respectively. We represent the distortion observed in the signal space as $D_S(\cdot)$. A following feature extraction procedure converts the speech signals to a few compact and perceptually meaningful features. We represent training and testing speech features as F_X and F_Y in Figure 2.10. From these features, the statistical models Λ_X and Λ_Y can then be trained. We denote the mismatches in feature and model spaces as $D_F(\cdot)$ and $D_M(\cdot)$, respectively.

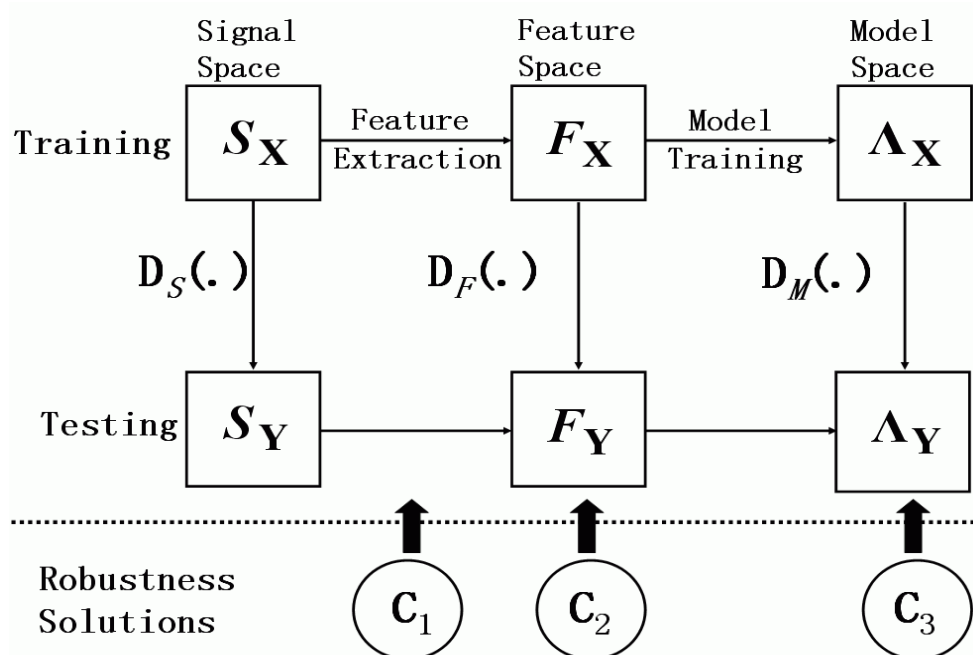


Figure 2.10. Mismatch modeling and three classes of solutions.

2.5 Noise Robustness Approaches

The approaches that tackle the mismatch problems can be roughly classified into three categories, C_1 , C_2 , and C_3 as shown in Figure 2.10. The first category of C_1 approaches is often referred to as speech enhancement methods because the objective is to produce robust features less sensitive to environment changes and reduce mismatch in the feature extraction stage. These approaches usually involve a new feature extraction procedure.

One such C_1 approach, spectral subtraction (SS), and its extensions [58-60], significantly reduce additive noise by subtracting the noise power spectrum from each speech frame. Likewise, the central mean subtraction (CMS) [61] normalizes speech features in the cepstral domain by subtracting the means from speech frames. Techniques using second or higher order cepstral moment normalization adjust the distribution of noisy speech features closer to that of the clean ones and provide further improvement over the first order CMS [62-65]. More recently, the ETSI advanced front-end is proposed to achieve good performance in ASR noise robustness [66]. This ETSI advanced front-end removes mismatch by using several stages of noise reduction schemes, including a two-stage Wiener filter, signal noise ratio (SNR)-dependent waveform processing, cepstrum calculation, and blind equalization.

The second category of approaches removes mismatches in the feature-space; we denote them as C_2 in Figure 2.10. These methods form a parametric function to model the distortion $D_F(.)$ between the training and testing features. The parametric function is estimated based on some optimality criterion and is used to compensate testing features. The codeword dependent cepstral normalization (CDCN) algorithm [67] and the stereo-based piecewise linear compensation environments (SPLICE) technique [68], for

example, perform feature compensation with a correction vector, which is estimated or located with a VQ codeword that indicates the gap between the training and testing environments. Similarly, both feature-space maximum likelihood linear regression (fMLLR) [69] and feature-space eigen-MLLR [70] compute affine transformations to compensate noisy speech features based on a maximum likelihood (ML) criterion and reduce mismatches even in unseen testing environments.

The third class of approaches, C_3 , reduces mismatches by improving the acoustic models so that they can be more robust over or accurately match various adverse testing conditions. These approaches can be classified into two categories, the offline preparation and online adjustment approaches. Among the offline preparation approaches, a good training procedure is a straightforward way to increase ASR robustness. It is well known that discriminative training methods, such as MCE [25] and soft margin estimation (SME) [26, 27], improve the maximum likelihood (ML) training under either clean or noisy conditions. Likewise, using data from different acoustic environments prepared in the offline stage in training models can improve robustness. For example, multicondition training usually achieves better performance over clean-condition training [30]. Meanwhile, a set of speaker independent (SI) HMMs trained on data from many different speakers is often favorable in speaker adaptation studies. On the other hand, the online adjustment approaches intend to map the original acoustic models Λ_X to a new set of acoustic models Λ_Y that matches the testing features. For these approaches, a set of speech segments from the testing environment is required for the mapping process, and these speech samples are called adaptation data. The model-mapping process can be done in either a direct or an indirect manner [71]. A direct mapping finds the target acoustic

models for the unknown testing environment directly. When sufficient adaptation data is available, such direct mapping achieves good performance. However, the performance improves marginally when only a limited amount of adaptation data is given. Maximum a posteriori (MAP) estimation [5] is a well-known method belonging to this category. On the other hand, indirect adaptation models the difference between training and testing conditions by a mapping function that maps the original models Λ_X to transformed models Λ_Y . The most often used form of the mapping function is the affine transformation. Maximum likelihood linear regression (MLLR) [6] and its Bayesian version, maximum a posteriori linear regression (MAPLR) [71, 72], have been adopted with good success, where the parameters of affine transformations are estimated through ML and MAP learning, respectively. Another successful mapping function is a distortion model, which characterizes the mismatch between Λ_X and Λ_Y . For the approaches using distortion models, a vector Taylor series (VTS) expansion is often used as an approximation technique. Examples include the joint compensation of additive and convolutive distortion (JAC) [73] and VTS-based HMM adaptation [74, 75]. When comparing the direct and indirect adaptation approaches, the later ones are generally more effective when a small set of adaptation data is available. Therefore, extensions have been proposed to the direct mapping approaches to improve performance in testing conditions with a small amount of adaptation data. One good extension is to introduce a hierarchical structure as a flexible parameter tying strategy in estimating HMM parameters. Structural MAP (SMAP) [76] uses such a hierarchical structure and shows performance improvements over the conventional MAP when only limited adaptation data is given. Moreover, a unified framework for a joint MAP adaptation of

transformation (indirect) and HMM (direct) parameters has been proposed [77] to not only achieve rapid model adaptation with limited adaptation data but also to continuously enhance performance when a large set of adaptation data is available.

For the approaches in C_3 , we consider the availability of adaptation data and the corresponding transcription for different adaptation modes. When we have an additional set of adaptation data from the testing environment and when the correct transcription is given, we update the model parameters in a supervised adaptation mode. We usually conduct the offline preparation in such a supervised learning mode. However, the correct transcriptions may not be available for the online adjustment in the real-world ASR systems. In such a case, an assumed transcription obtained through a recognition process, such as decoding, should precede the mapping procedure. We call this second mode an unsupervised adaptation mode. Next, when we do not have an additional set of data, we use a segment of testing speech data to first transform the acoustic models; then, the same testing utterance is decoded with the transformed acoustic models. This process can be repeated until convergence. We call this third mode a self-adaptation or compensation mode. Generally, the unsupervised compensation mode is more user-friendly in ASR applications because it does not require active enrollment by the users. However, this mode may not achieve good performance since the statistics from the testing data might be too limited and the decoded referenced transcription can be incorrect. To overcome this limitation, a stochastic matching algorithm, which effectively estimates the mismatch factor in a maximum likelihood manner, has been proposed [12, 13, 78].

More recently, we extend the stochastic matching algorithm and propose an ensemble speaker and speaking environment modeling (ESSEM) approach for

characterizing environments in the presence of multiple distortion sources [14-17]. In the ESSEM framework, we model an environment of interest with a super-vector, consisting of the entire set of mean vectors from all the Gaussian components of a set of HMMs for that particular environment. Moreover, we prepare an environment configuration based on a collection of speech data from a variety of different speaker and speaking conditions. With such an environment configuration, we estimate a target super-vector for the unknown testing environment online with a mapping function. The target super-vector is then used to construct HMMs for the testing environment.

Based on our previous study [14], we know that two classes of mapping procedures are applicable to find the target super-vector, namely, direct and indirect ESSEM approaches. For direct ESSEM, we estimate the target super-vector through a mapping function along with a large collection of environment-specific super-vectors consisting of parameters in HMM sets trained with speech data from their corresponding environments. For indirect ESSEM, we first use a mapping function to estimate a transformation with another large collection of transformations, each corresponding to the mapping required for a particular known environment over a reference super-vector. Then, we compute the target super-vector with the estimated transformation and the reference super-vector [14]. Similar frameworks to indirect ESSEM show good performance in speaker adaptation [79, 80]. More recently, some approaches resembling the indirect ESSEM approach have been proposed [81, 82] for rapid speaker adaptation. These methods first map every super-vector, which consists of the entire set of mean vectors in a set of HMMs or the entire set of transformation parameters, to a higher dimensional space. The method then calls for the use of a mapping function to find a super-vector for the testing environment

in the corresponding high-dimensional space. Finally, the estimated high-dimensional super-vector is mapped to the original space and forms the HMMs for speech recognition.

The proposed ESSEM framework was derived from the stochastic matching algorithm. Therefore, we first review the stochastic matching algorithm in the next section. Then in Chapter 3, we will further detail the main concept and implementation steps of the ESSEM framework.

2.6 ML-Based Stochastic Matching Algorithm

In this section, we briefly review the ML-based stochastic matching approach [12, 13, 78]. Based on Eq-(2.2), the stochastic matching approach uses a mapping function, \mathbf{G}_φ , with parameters φ to transform the original models, Λ_X , to desired models, Λ_Y , for the testing environment by:

$$\Lambda_Y = \mathbf{G}_\varphi(\Lambda_X). \quad (2.36)$$

The form of the mapping function depends on the available adaptation data and the type of the acoustic mismatch. For example, a set of affine transformations is often used as the mapping function in many applications. The parameters φ are generally called nuisance parameters that are only used in the mapping procedure but not involved in the recognition procedure. From Eq-(2.2) and Eq-(2.36), we formulate a joint maximization problem over the variables φ and \mathbf{W} :

$$(\varphi', \mathbf{W}') = \underset{(\varphi, \mathbf{W})}{\operatorname{argmax}} P(F_Y | \varphi, \mathbf{W}, \Lambda_X) P(\mathbf{W}). \quad (2.37)$$

An iterative procedure solves (φ, \mathbf{W}) by keeping φ fixed and maximizing over \mathbf{W} , and then keeping \mathbf{W} fixed and maximizing over φ and repeats the process for a few iterations. Since our main interest of stochastic matching is to compute the parameters φ ,

we remove the dependence of \mathbf{W} for notational simplicity and rewrite Eq-(2.37) as:

$$\varphi' = \underset{\varphi}{\operatorname{argmax}} P(F_Y | \varphi, \Lambda_X) . \quad (2.38)$$

The nuisance parameters in Eq-(2.38) are estimated based on the expectation-maximization (EM) algorithm [83]. In addition to the affine transformations, a simpler form such as a bias in the stochastic bias [12] and signal bias removal (SBR) [84], and a more complex structure, such as a non-linear transformation [13] have also been used to model the mismatch factors and provide performance improvements. Figure 2.11 demonstrates the architecture of the stochastic matching framework in an ASR system.

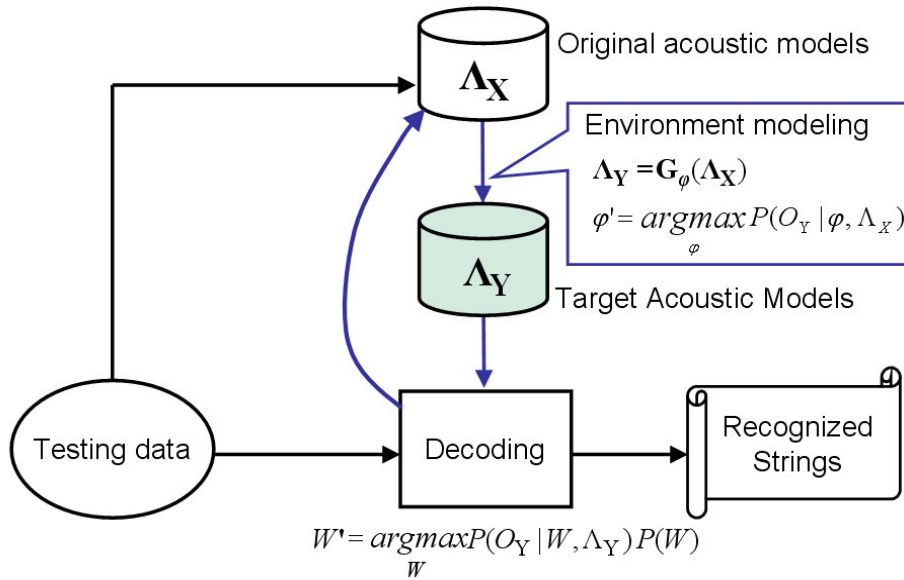


Figure 2.11. System architecture of the stochastic matching framework.

CHAPTER 3

ENSEMBLE SPEAKER AND SPEAKING ENVIRONMENT MODELING (ESSEM)

In this chapter, we detail the main concept and implementation steps of the proposed ESSEM approach. As mentioned earlier, the ESSEM framework is an extension of the stochastic matching framework (Figure 2.11) with incorporating prior knowledge. Figure 3.1 diagrams the ESSEM framework on an ASR system. The prior knowledge is prepared based on multiple sets of training data. Each set of data is collected from one particular acoustic condition. With the prior knowledge, a mapping function is estimated based on the testing utterances to generate a set of acoustic models characterizing the testing condition. In the following sections, we first introduce two types of ESSEM, direct and indirect ESSEM. After that, we will present the experimental results and discussions.

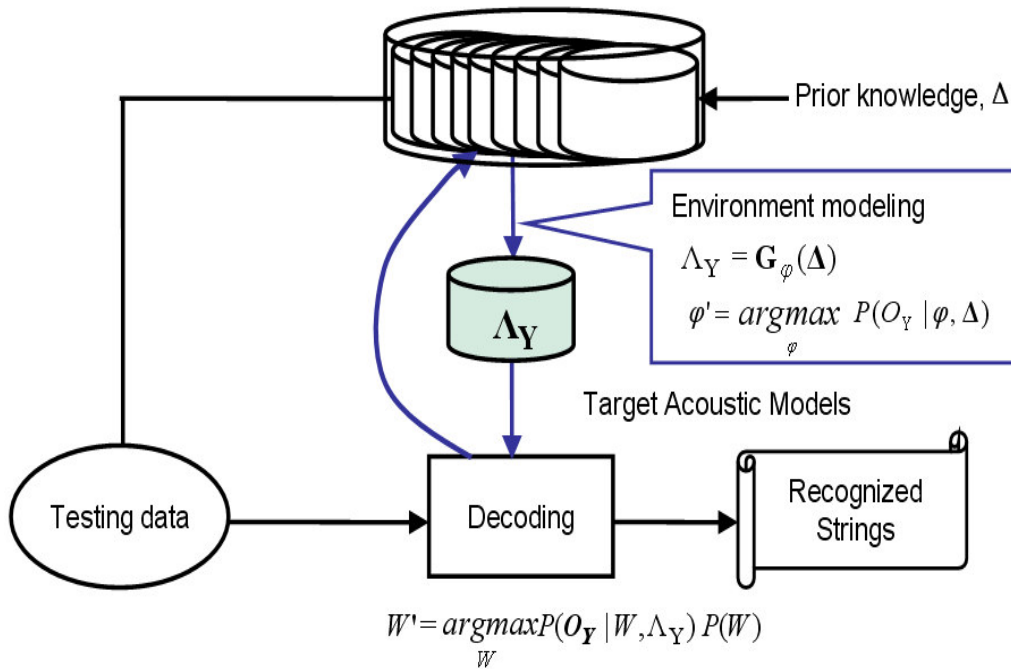


Figure 3.1. Architecture of the ESSEM framework.

3.1 Direct ESSEM

A direct ESSEM framework comprises two integral stages: offline and online phases. In the offline phase, we collect a wide range of speech data from different speaker and speaking environments, e.g., different speakers, noise types, SNR levels, and channel distortions. In the real-world applications, it may be expansive to collect speech data from a wide range of real-world conditions. Therefore, we propose to artificially simulate the training sets [14, 17] (to be discussed in Section 3.3). After the simulation process, we can obtain different conditions with various combinations of multiple distortion sources.

Figure 3.2 demonstrates the system architecture of the proposed direct ESSEM framework. When we collect or simulate P sets of training data, we can train P sets of HMMs, Λ_p , $p=1, \dots, P$, for P different speaker and speaking environments. For ease of modeling each environment, the entire set of the mean parameters for each Gaussian density within a set of HMMs is concatenated into a super-vector, \mathbf{V}_p , $p=1, \dots, P$. If one set of HMMs contains M Gaussian mixture components, and every mean vector has D dimensions, then the super-vector for the p -th environment is an R -dim ($R=D \times M$) vector. These P super-vectors form an ensemble speaker and speaking (ESS) environment space, $\Omega_{\mathbf{V}} = \{ \mathbf{V}_1 \mathbf{V}_2 \dots \mathbf{V}_P \}$, that serves as prior knowledge for estimating the super-vector representing of the target condition. In the online, we estimate the target super-vector, \mathbf{V}_Y , for a testing environment with the ESS space, and Eqs-(2.36) and (2.38) become:

$$\mathbf{V}_Y = \mathbf{G}_{\varphi}(\Omega_{\mathbf{V}}), \quad (3.1)$$

$$\varphi' = \underset{\varphi}{\operatorname{argmax}} P(F_Y | \varphi, \Omega_{\mathbf{V}}) . \quad (3.2)$$

Again, we use the EM algorithm [83], to estimate the nuisance parameters, φ , in Eq-(3.2).

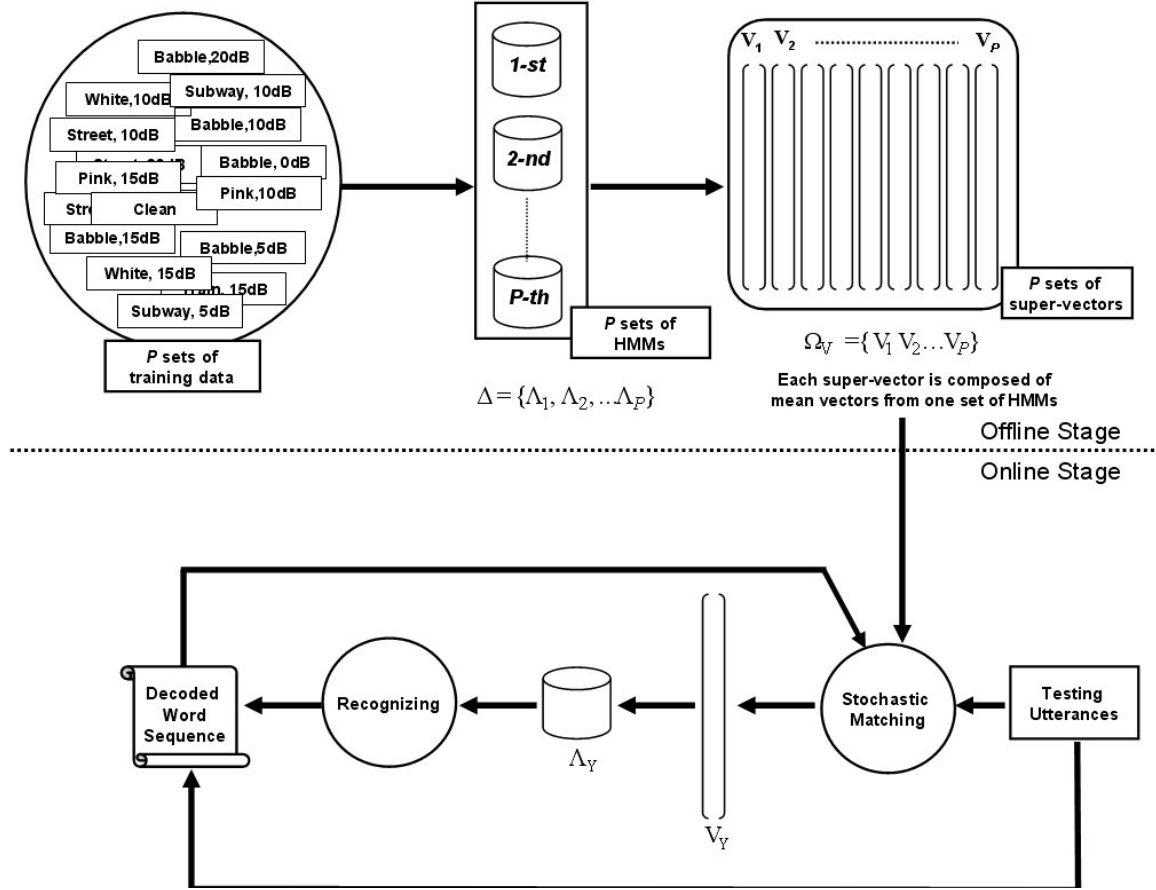


Figure 3.2. System architecture of the direct ESSEM framework.

3.2 Indirect ESSEM

For the indirect ESSEM approach, we describe an environment by a transformation characterizing its correlation with the training environment [14]. The super-vector for the testing environment is computed as:

$$V_Y = T_{test}(V_X), \quad (3.3)$$

$$T_{test} = G_\varphi(\Gamma) \quad (3.4)$$

where V_X is the anchor (reference) super-vector for the training environment, Γ is the environment configuration, and the transformation, T_{test} , is the transformation for the

testing environment. Figure 3.2 illustrates the system architecture of the indirect ESSEM framework. The implementation can also be divided into the offline and online stages. In the offline phase, P sets of transformations, $\Gamma = \{T_1, T_2, \dots, T_P\}$, corresponding to P different environments are calculated as follows:

$$T_p = \arg \max_T P(F_p | T(V_X)), \quad p=1, 2, \dots, P, \quad (3.5)$$

where F_p is the training data from the p -th environment. Parameters in one transformation are then vectorized into a super-vector. With these P super-vectors, a transformation-based ensemble speaker and speaking (TESS) environment space with dimension P is constructed. On the other hand in the online phase, the transformation, G_ϕ , is estimated with speech data from the unknown testing environment.

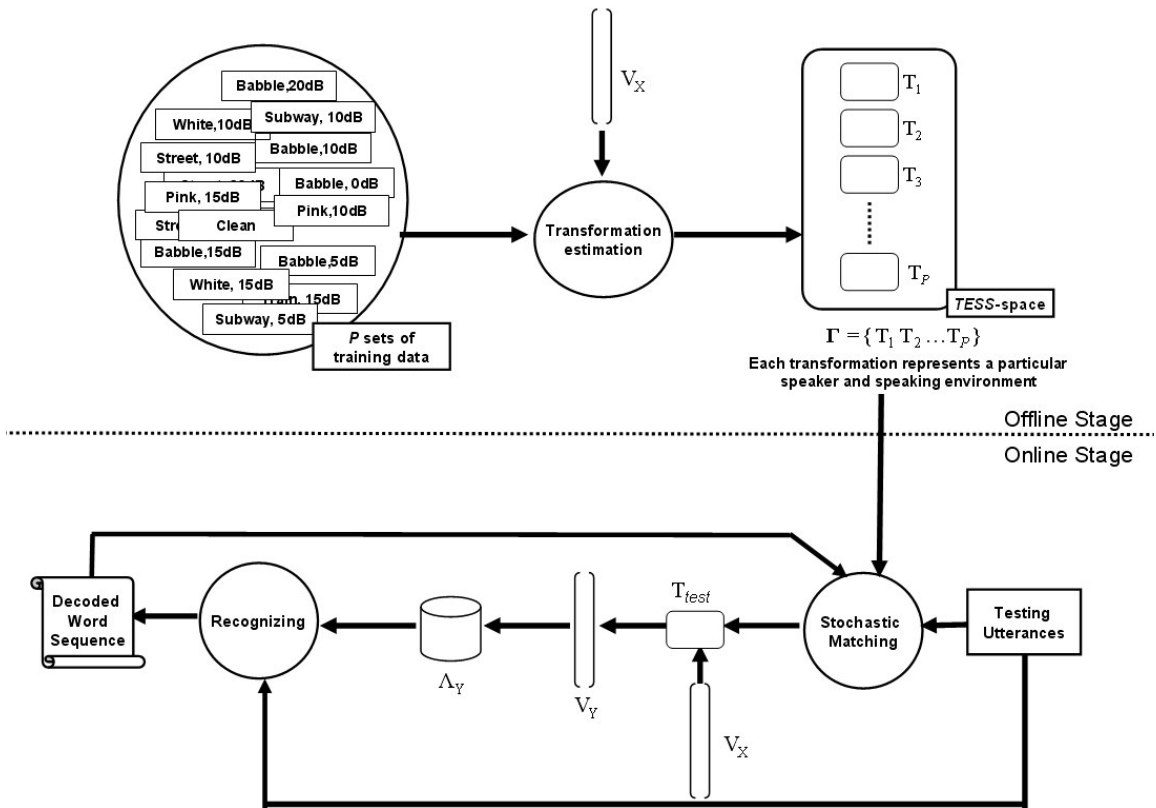


Figure 3.3. System architecture of the indirect ESSEM framework.

3.3 Environment Structure Simulation

As mentioned in Section 3.1, it can be prohibitive to collect speech data from different adverse conditions, and we decide to artificially simulate speech data covering a wide range of acoustic conditions and signal-to-noise (SNR) levels. The simulation process is based on a distortion model illustrated in Figure 3.4.

The distortion model of Figure 3.4 is widely used to represent the correlation between clean speech and degraded speech with both additive noise and convolutive channel distortions. The observed distorted speech, $y[m]$ is generally presented by:

$$y[m] = x[m] * h[m] + n[m] , \quad (3.6)$$

where $x[m]$, $h[m]$, and $n[m]$, respectively, are the clean speech, channel distortion, and noise distortion in the discrete time domain.

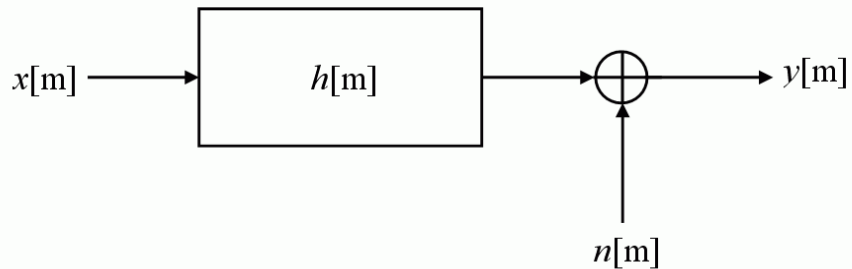


Figure 3.4. A model for speaking environment distortion.

Figure 3.5 demonstrates the process of simulation in a signal domain. To simulate speech signals for a wide range of different acoustic conditions, we use clean speech as the input to the model in Figure 3.5. After passing clean speech signals through various channel characteristics and combining additive noises at different noise types and SNR levels, we can simulate speech signals for many different speaking environments. With this simulation process, we can accordingly build the environment structure. In addition to simulating speech signals, we can also simulate speech features and acoustic models.

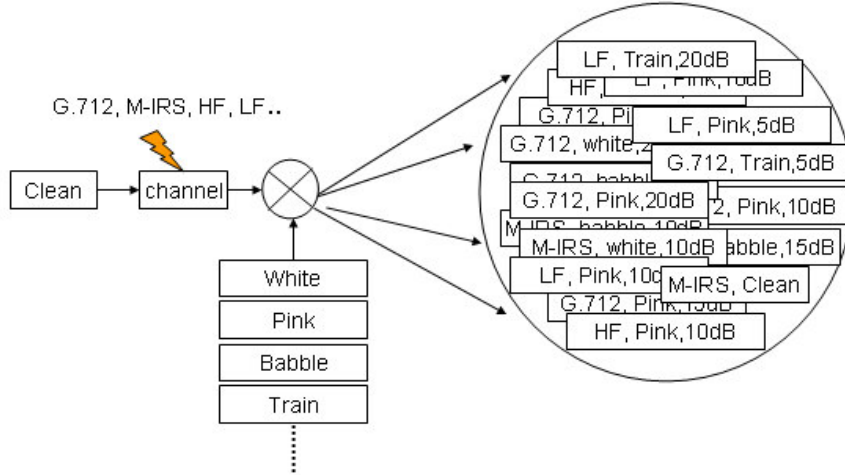


Figure 3.5. Simulation in the signal domain.

When applying discrete Fourier transform (DFT) on both sides of Eq-(3.6), we can represent the distortion model in the spectral domain by:

$$|Y[s]| = |X[s]| |H[s]| + |N[s]|, \quad (3.7)$$

Next, in the log-spectral domain, we can express the distorted speech with ignoring the phase by:

$$\log |Y[s]| = \log [|X[s]| |H[s]| + |N[s]|], \quad (3.8)$$

$$\log |Y[s]| = \log [|X[s]| |H[s]| (1 + |N[s]| / |X[s]| |H[s]|)], \quad (3.9)$$

$$\log |Y[s]| = \log |X[s]| + \log |H[s]| + \log (1 + |N[s]| / |X[s]| |H[s]|) . \quad (3.10)$$

After applying discrete cosine transform (DCT) matrix to both sides of Eq-(3.10), we obtain the following nonlinear distortion model:

$$y = x + h + C \log(1 + \exp(C^{-1}(n - x - h))) , \quad (3.11)$$

where C and C^{-1} are the DCT and (pseudo) inverse DCT matrices, respectively; y , x , h , and n are the distortion speech, clean speech, channel and noise parameters, all in the cepstral domain. Next, we derive the following approximating equation to model these

parameters:

$$\mu_y \approx \mu_x + \mu_h + C \log(1 + \exp(C^{-1}(\mu_n - \mu_x - \mu_h))) , \quad (3.12)$$

where μ_y , μ_x , μ_h , and μ_n are mean vectors of the cepstral parameters y , x , h , and n , respectively. With given channel and noise mean parameters, the distorted mean vectors, μ_y , can be calculated and used for recognition. Based on Eq-(3.11) and Eq-(3.12), we can simulate the distortion environments in feature and model domains, respectively.

If we obtain acoustic information of the testing condition in advance, we can prepare a relevant prior environment structure that well suits the testing condition. However, in most real-world ASR applications, such acoustic information is unavailable before testing. Thus in this study, we intend to prepare speech data for a wide range of different environments and therefore prepare the environment space covering as many different acoustic conditions as possible.

3.4 Experiments

In this section, we present experiments for both direct and indirect ESSEM. To investigate the ESSEM performance under task mismatch testing, we intentionally select three sets of databases, TIMIT [55], NOISEX-92 [85], and Aurora-2 [86] to provide training, noise sources, and testing data, respectively. TIMIT corpus is used as a domain-independent, non-digit training database to obtain phone HMMs in a clean condition. The noise sources needed to perform environment simulation are extracted from the wide selection of noise types in the NOISEX-92 database. The proposed approach is evaluated on the Aurora-2 connected digit recognition task in several diverse conditions. More details about these three databases are described in Appendix B.

Fifteen different types of noise sources were selected from the NOISEX-92 database

[85]. The simulation technique can now be employed to generate the noise data at different SNR levels with various noise types and then be added to the TIMIT data to obtain new artificial training data at a particular point in the noise space. When using S different noise sources with L different SNR levels, P ($P=S \times L$) noisy environments can be constructed to simulate the environment space. Moreover, with different combinations of noise sources and SNR levels, we can qualitatively and quantitatively analyze the characteristics of the environment space.

For compatibility in sampling rates, all the speech and noise data were down-sampled to 8 KHz before performing feature extraction. We used a commonly adopted feature vector of 39 elements, consisted of 13 MFCC parameters plus their first and second order time derivatives. An utterance-level cepstral mean subtraction (CMS) [61] was performed for normalization.

For each environment, the entire training set with 3696 utterances in the TIMIT or simulated TIMIT database was used to train 45 English phone HMMs. All models have 3 states with each state characterized by a mixture Gaussian density with 16 mixture components. The test Set A in the Aurora-2 database was used as the testing set. The testing set includes four different noise types of subway, babble, car, and exhibition over four SNR levels at 5, 10, 15, 20 dB. With the NOISEX-92 database, 285 (15×19) sets of artificially generated noise signals were obtained from combinations of 15 different types of noises, and 19 different SNR levels, ranging from -5dB to 40dB. It should be noted that there is no digit knowledge involved in estimating HMMs, nor in constructing the environment spaces.

From our preliminary experiments, we observed that increasing the number of noise

sources does not improve recognition performance, and some representative noises can be used to effectively construct the environment space. On the other hand, the coverage of SNR levels is more related to the performance in environment modeling. Instead of using all environments, we select 48 (with white, pink and car noise sources at 16 different SNR levels between 0dB to 40dB) representative conditions to construct the environment space in the following experiments.

For the experiments in this section, we adopt the linear combination function as the online mapping function, \mathbf{G}_φ , and therefore Eq-(3.1) becomes:

$$\mathbf{V}_Y = \sum_{p=1}^P \hat{w}_p \mathbf{V}_p, \quad (3.13)$$

where \hat{w}_p is the p -th weighting coefficient in the linear combination function. The set of weighting coefficients is estimated in the online step according to the ML algorithm:

$$\{\hat{w}_p\}_{p=1}^P = \arg \max_{\{w_p\}_{p=1}^P} P(F_Y | \sum_{p=1}^P w_p \mathbf{V}_p). \quad (3.14)$$

Meanwhile, Eq-(3.4) becomes:

$$\mathbf{T}_{test} = \sum_{p=1}^P \hat{w}_p \mathbf{T}_p, \quad (3.15)$$

where the weight coefficients, $\{\hat{w}_p\}_{p=1}^P$, can also be estimated based on the ML criterion:

$$\{\hat{w}_p\}_{p=1}^P = \arg \max_{\{w_p\}_{p=1}^P} P(F_Y | (\sum_{p=1}^P w_p \mathbf{T}_p) \mathbf{V}_X). \quad (3.16)$$

In this study, we use the MLLR [6] transformation matrix to represent the environment- specific transformations, \mathbf{T}_p , $p=1,2,\dots,P$, and the target transformation, \mathbf{T}_{test} . Other linear or nonlinear transformations can also be used for the indirect ESSEM. It can be a future study to investigate different forms of transformation for indirect ESSEM.

We tested the direct and indirect ESSEM approaches in both supervised and unsupervised adaptation modes. For the supervised adaptation mode, 10 adaptation utterances with their corresponding transcriptions were given and used to provide the statistics needed in Eq-(3.13) and Eq-(3.15). The baseline result was obtained by using the clean condition trained HMMs from TIMIT. A set of comparative experiment was conducted by using conventional MLLR [6] to do environment model adaptation, with 6 sets of MLLR matrices corresponding to 6 different manners of articulation, namely, vowel, nasal, fricative, nasal, stop and silence. They were estimated directly with the adaptation utterances, and used to adapt HMM parameters for the testing environment. In Table 3.1, we list the average word error rates (WER) of “Baseline”, “MLLR”, “Direct ESSEM”, and “Indirect ESSEM” in the supervised adaption mode.

Table 3.1. WER (%) for baseline, MLLR, direct and indirect ESSEM in a supervised adaption mode.

SNR	Baseline	MLLR	Direct ESSEM	Indirect ESSEM
20 dB	10.76	9.87	9.62	9.63
15 dB	20.13	16.54	13.93	13.71
10 dB	46.73	26.02	21.79	21.39
5 dB	77.63	42.51	40.03	39.89
Ave.	38.81	23.74	21.34	21.16

From Table 3.1, it is noted that both direct and indirect ESSEM approaches achieve better performance than “Baseline” and “MLLR”, and clearer improvements were observed in lower SNR conditions. For example, when SNR=5dB, the error rate reductions from “Baseline” to “Direct ESSEM” and “Indirect ESSEM” are 48.43% (from 77.63% to 40.03%) and 48.62% (from 77.63% to 39.89%), respectively. When SNR=10dB, the error reduction rates from “MLLR” to “Direct ESSEM” and “Indirect ESSEM” are 16.26% (from 26.02% to 21.79%) and 17.79% (from 26.02% to 21.39%). It

is noted that the “Indirect ESSEM” here also used 6 sets of matrices to characterize the testing environment, and each matrix had the same formats as those used in MLLR (diagonal matrix with a correction bias). However, “Indirect ESSEM” only estimated 6 sets of weighting coefficients to determine the transformation matrix. The number of free parameters for “MLLR” here was $6 \times (39+39)$, while for “Indirect ESSEM” was only 6×48 . The realization of this data reduction was made by the prior information when building the TESS space.

Next, we test direct and indirect ESSEM in an unsupervised adaptation mode. Since N -best information has already been reported to be positively beneficial to accomplish unsupervised speaker adaptation, we incorporated the N -best information to realize unsupervised adaptation with both the direct and indirect ESSEM approaches. For direct ESSEM, Eq-(3.14) becomes:

$$\{\hat{w}_p\}_{p=1}^P = \arg \max_{\{w_p\}_{p=1}^P} \sum_{n \in N} \lambda_n P(F_Y | \sum_{p=1}^P w_p V_p, W_n), \quad (3.17)$$

where W_n and λ_n are the decoded transcription and weight for the n -th hypothesis.

Similarly for indirect ESSEM, Eq-(3.16) can be rewritten as:

$$\{\hat{w}_p\}_{p=1}^P = \arg \max_{\{w_p\}_{p=1}^P} \sum_{n \in N} \lambda_n P(F_Y | (\sum_{p=1}^P w_p T_p) V_X, W_n). \quad (3.18)$$

In this section, we put equal weight to each of the N -best lists by setting $\lambda_n=1/N$. Experiments to finding the optimal weight for each λ_n will be discussed in Chapter 5. For comparison purpose, we used the same N -best information to conduct an MLLR experiments. Table 3.2 lists results of MLLR, direct ESSEM, and indirect ESSEM.

From Table 3.2, it is noted that when compared with the two ESSEM approaches in

the supervised mode with 10 adaptation utterances in Table 3.1, the unsupervised solution achieved mostly similar and sometimes even slightly better performance at 20dB SNR. We see that this unsupervised adaptation ESSEM is very effective and can be used in many adverse conditions to improve performance. When compared with the “Baseline” in Table 3.1, the unsupervised direct and indirect ESSEM approaches give an average word error rate (WER) reductions of 42.57% (from 38.81% to 22.29%) and 42.67% (from 38.81% to 22.25%), respectively, from SNR=5dB to 20dB conditions. Moreover, when comparing to “MLLR” in Table 3.2, the unsupervised direct and indirect ESSEM adaptations can give an average WER reductions of 7.97% (from 24.22% to 22.29%) and 8.13% (from 24.22% to 22.25%), respectively, from SNR=5dB to 20dB conditions.

Table 3.2. WER (%) for MLLR, direct and indirect ESSEM in an unsupervised adaption mode.

SNR	MLLR	Direct ESSEM	Indirect ESSEM
20 dB	11.43	9.63	9.57
15 dB	15.70	14.12	14.24
10 dB	23.69	22.42	22.51
5 dB	46.04	42.98	42.66
Ave.	24.22	22.29	22.25

3.5 Summary

In this chapter, we introduce the direct and indirect ESSEM approaches. We also present a simulation method to artificially construct an environment configuration with a good coverage of many different environments. To demonstrate the performance under task mismatch conditions, we intentionally select three different sets of database to test ESSEM. TIMIT is used as the clean training set, NOISEX-92 provides noise sources for simulation, and Aurora-2 is the testing set. It is noted that some environments in the

Aurora-2 testing set contain noise sources that are not included in the NOISEX-92. From the experimental results in Table 3.1 and Table 3.2, we first verify that both two types of ESSEM approaches significantly reduce WERs in adverse conditions. Improvements are clearer in lower SNR conditions. In supervised and unsupervised adaptation modes, indirect ESSEM algorithm can achieve 48.43% (from 77.63% to 39.89%) and 45.05% (from 77.63% to 42.66%) WER reductions from the baseline result, respectively, at 5dB SNR testing condition. When SNR=20dB, error rate reduction from baseline result to supervised and unsupervised indirect ESSEM are 10.50% (from 10.76% to 9.63%) and 11.06% (from 10.76% to 9.57%), respectively. Because an unsupervised adaptation is usually preferred in robustness techniques, the ESSEM approaches can be used in many adverse conditions for their good performance in unsupervised environment modeling. Finally, from Table 3.1 and Table 3.2, we can see that direct ESSEM and indirect ESSEM achieve similar performance in every testing condition. Therefore in the following paragraph, we focus our discussion on direct ESSEM.

CHAPTER 4

OFFLINE ENVIRONMENT CONFIGURATION REFINEMENT

In the previous chapter, we introduce the implementation steps of the ESSEM framework. In this chapter, we present techniques to enhance ESSEM performance by improving the construction of the environment configuration in the offline phase. We first state the EC and EP algorithms to structure the environment configuration well; then, we introduce discriminative training to enhance the discriminative power of the overall environment configuration. As mentioned earlier, we limit our discussion on direct ESSEM.

4.1 Improving Structure of the Environment Spaces

The basic concept of environment clustering (EC) resembles that of well-known subset selection methods [18-20] that determine a subset of components from the entire set of components to model a signal of interest. On the other hand, environment partitioning (EP) is similar to the family of piecewise-polynomials and splines [24] that approximate complicated functions with local polynomial representations. Therefore, the EC and EP algorithms structure the ESS space by preparing more sets of sub-spaces instead of relying on a global one.

4.1.1 Environment Clustering (EC)

First, we propose EC to cluster the ensemble environments into several groups, with each group consisting of environments having closer acoustic properties. Environments within the same group then form a sub-space. Figure 4.1 demonstrates the architecture of environment clustering in the offline phase. In our study, we adopt a hierarchical clustering procedure to construct a tree structure. The root node of the tree is the entire

set of training environments, and the tree is partitioned into several layers, with each layer of environment clustering performed based on the similarity between each pair of environments. In the offline phase, the super-vectors belonging to the same cluster form an environment clustering (EC) ESS sub-space. If the hierarchical structure has C groups of environments (including the root node, intermediate nodes, and leaf nodes), we categorize the original ESS space into C sub-spaces: $\Omega_{\mathbf{V}} = \{ \Omega_{\mathbf{V}^{(1)}} \cup \Omega_{\mathbf{V}^{(2)}} \dots \cup \Omega_{\mathbf{V}^{(C)}} \}$.

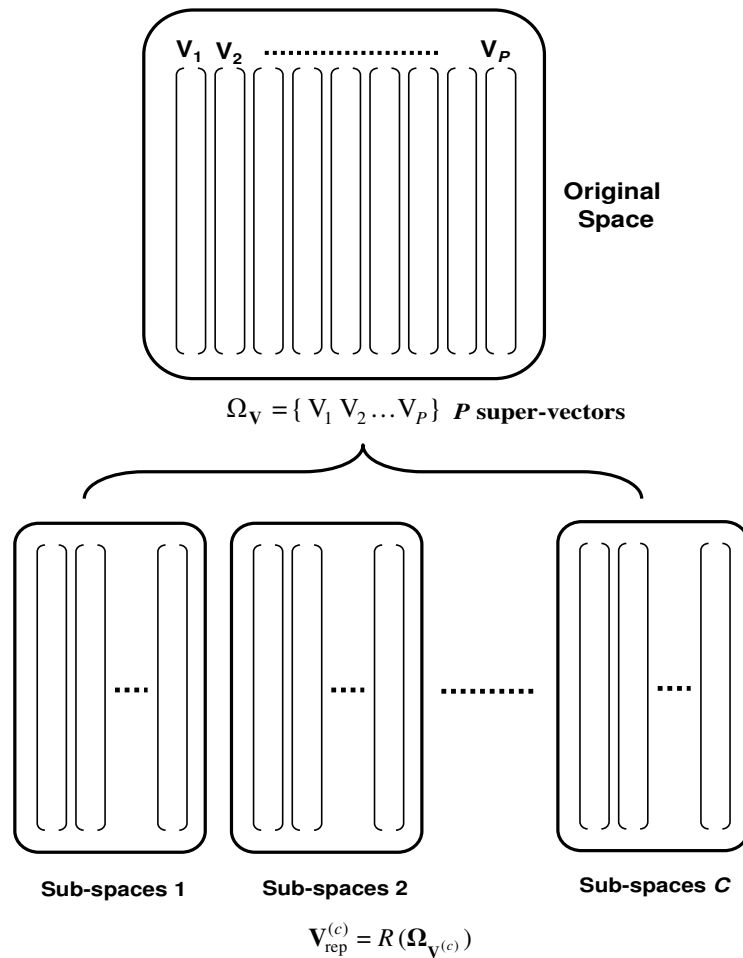


Figure 4.1. Architecture of environment clustering algorithm.

We use a function, $R(\cdot)$, to specify the most representative super-vector for each sub-space; for example, the super-vector, $\mathbf{V}_{\text{rep}}^{(c)}$, represents the c -th cluster, $\Omega_{\mathbf{V}^{(c)}}$, with:

$$V_{\text{rep}}^{(c)} = R(\Omega_{\mathbf{V}^{(c)}}). \quad (4.1)$$

The similarity measure between a pair of environments is defined either by a deterministic distance between two super-vectors or with knowledge about the acoustic difference between them. Using a deterministic distance allows us to construct a hierarchical tree in a data-driven manner, but it is more perceptually meaningful to incorporate acoustic knowledge in measuring environment similarity. For example, it is straightforward to cluster environments based on the different types of distortion components they contain, such as speaker differences, background noises, and channel variations. Therefore, we can form a speaker sub-space, $\Omega_{\mathbf{V}^{(p)}}$, noise sub-space, $\Omega_{\mathbf{V}^{(b)}}$, and channel sub-space, $\Omega_{\mathbf{V}^{(h)}}$, individually, in the offline phase, by:

$$\Omega_{\mathbf{V}} = \{ \Omega_{\mathbf{V}^{(p)}} \cup \Omega_{\mathbf{V}^{(b)}} \cup \Omega_{\mathbf{V}^{(h)}} \}. \quad (4.2)$$

A combination of the deterministic distance and acoustic difference can be another tree construction scheme. For such a case, we first cluster environments into different distortion domains based on the distortion sources they contain. Then, we build a hierarchical tree for each distortion domain based on the deterministic distance. A good example is the hierarchical tree structure for reference speaker weighting adaptation [87].

4.1.2 Environment Partitioning (EP)

Next, we present the EP algorithm to structure ESS spaces. Instead of clustering environments, we partition each super-vector into several sets of sub-vectors. Then, we collect each set of sub-vectors among all the training environments to form a sub-space. From our previous studies [21, 22], two types of super-vector partitioning are successful, namely, the mixture-based and feature-based EP techniques. For mixture-based

partitioning, we establish a tying structure to cluster Gaussian mixture components with close acoustic properties, as in the tree structure in structural maximum a posteriori (SMAP) adaptation [76]. Therefore, the entire set of Gaussian mixture components in a set of HMMs is classified into S clusters. Then, the original super-vector is partitioned into S sets of sub-vectors. For example, in the super-vector for the p -th environment, we have $V_p = [V_{p,1}^T, V_{p,2}^T, \dots, V_{p,S}^T]^T$, with each sub-vector formed by the same cluster of Gaussian mixture components. Then, each set of such sub-vectors from the training environments forms a sub-space individually, $\Omega_{V_s} = \{V_{1,s}, V_{2,s}, \dots, V_{P,s}\}$, $s=1,2,\dots,S$. Now the original space is partitioned into S sub-spaces. Another straightforward tying method is to classify models with whole-word or sub-word units, and therefore their Gaussian components, into different clusters based on the acoustic or linguistic knowledge [22]. For example, phone units are clustered into one group when they belong to the same broad phonetic class, such as consonants or vowels.

The concept of feature-based EP, on the other hand, is different from mixture-based EP. For the feature-based EP technique, we consider parameter tying on different types of vector components, e.g., energy coefficients, MFCC coefficients, first-order and second-order derivative coefficients. If we tie vector components into Z groups of components, the original super-vector is partitioned into Z sub-vectors. For example, we have $V_p = [V_{p,1}^T, V_{p,2}^T, \dots, V_{p,Z}^T]^T$ for the p -th environment. Thus, we construct Z sets of sub-spaces, $\Omega_{V_z} = \{V_{1,z}, V_{2,z}, \dots, V_{P,z}\}$, $z=1,2,\dots,Z$, with each sub-space spanned by a group of coefficient components. Figure 4.2 shows the EP algorithm on ESSEM.

Environment Partitioning

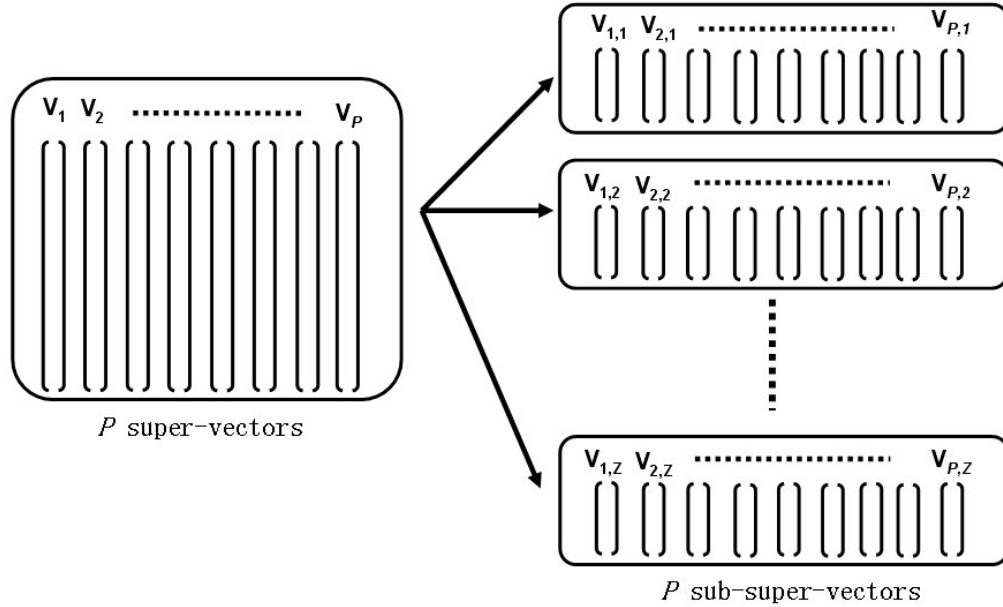


Figure 4.2. Architecture of environment partitioning algorithm.

4.2 Increasing Coverage of the Environment Spaces

Traditionally, discriminative training methods, such as MCE [25], maximum mutual information estimation (MMIE) [88], minimum word/phone error (MWE/MPE) [89], and soft margin estimation (SME) [26, 27], were used to refine accuracy of acoustic modeling. In the ESSEM framework, we use the discriminative training to maximize the separation between super-vectors to spread the coverage of the ESS space. We propose two training modes on the ESS space, intra-environment (intraEnv) and inter-environment (interEnv) training. For both intraEnv and interEnv training, the parameters in the ESS spaces are first estimated with the ML criterion and then refined by the discriminative training. In our implementation, we refine the ESS space iteratively by intraEnv training and

interEnv training. After intraEnv and interEnv training, the ESS space can cover more different speaker and speaking environments that may not be collected in the training set.

4.2.1 Intra-Environment (IntraEnv) Training

For the intraEnv training mode, we increase the distance between components within each particular environment. Among the discriminative training methods, we first adopt MCE training [25] for intraEnv training. With the training data F_p of U_p utterances from the p -th environment, we have the following objective function:

$$L(V_p) = \frac{1}{U_p} \sum_{u=1}^{U_p} \frac{1}{1 + \exp(-\gamma d(F_p^u, V_p, \Psi) + \theta)} \quad (4.3)$$

where both γ and θ are control parameters for the sigmoid function, and Ψ stands for the entire set of parameters other than the means in HMMs. Since our goal is to minimize the objective function by adjusting parameters in the ESS space, or equivalently means of environment-specific models, we set Ψ fixed across different environments. Now, we have the misclassification measure $d(\cdot)$ [25] defined as:

$$d(F_p^u, V_p, \Psi) = -\tilde{g}(F_p^u, V_p, \Psi, W_c) + \tilde{G}(F_p^u, V_p, \Psi), \quad (4.4)$$

with

$$\tilde{G}(F_p^u, V_p, \Psi) = \frac{1}{\eta} \log \left\{ \frac{1}{N} \sum_{n=1}^N \exp[\eta \times \tilde{g}(F_p^u, V_p, \Psi, W_n)] \right\}, \quad (4.5)$$

where η is a positive control parameter, W_c is the given correct transcription, and $\{W_1, \dots, W_N\}$ are the N -best decoded competing word sequences. The N -best word sequences are generated by decoding F_p^u using the HMMs for the p -th environment. We used a logarithm of the likelihood for the discrimination function, $\tilde{g}(\cdot)$, in Eq-(4.4) and Eq-(4.5). We then use the generalized probabilistic descent (GPD) algorithm [51] to

update parameters in V_p iteratively:

$$V_p(t+1) = V_p(t) - \kappa \nabla L(V_p) |_{V_p=V_p(t)}, \quad (4.6)$$

where κ is a step size.

More recently, a soft margin estimation (SME) criterion has been proposed and shows better performance than MCE on many ASR tasks [26, 27]. To achieve better ESSEM performance, we also adopt SME for the intraEnv training.

Originated from the statistical learning theory [90], SME considers the test risk to be bounded by two terms, an empirical risk and a generalization term (generalization term is bounded by a decreasing function of margin [90]). During optimization, SME not only minimizes the empirical risk but also maximizes the margin. Therefore, the objective function for SME is defined as:

$$L^{SME}(\rho, \Lambda) = \frac{\lambda}{\rho} + R_{emp}(\Lambda) = \frac{\lambda}{\rho} + \frac{1}{U} \sum_{u=1}^U l(F^u, \Lambda), \quad (4.7)$$

where Λ denotes HMM parameters, $l(F^u, \Lambda)$ is a loss function for the u -th utterance F^u , U is the number of training utterances, ρ is the soft margin, and λ is a coefficient to balance the soft margin maximization and the empirical risk minimization. The loss function is defined by a hinge loss function ($(x)_+ = \max(x, 0)$) as:

$$l(F^u, \Lambda) = \left[\rho - d(F^u, \Lambda) \right]_+ = \begin{cases} \rho - d(F^u, \Lambda), & \text{if } \rho - d(F^u, \Lambda) > 0, \\ 0, & \text{otherwise} \end{cases} \quad (4.8)$$

with the separation measure d defined as:

$$d(F^u, \Lambda) = \frac{1}{n_u} \sum_r \log \left[\frac{P_\Lambda(F^{ur} | S_u)}{P_\Lambda(F^{ur} | \hat{S}_u)} \right] I(F^{ur} \in D_u), \quad (4.9)$$

where D_u is the frame set in which the frames have different labels in the competing strings. n_u is the number of frames in D_u . $I(\cdot)$ is an indicator function, and F^{ur} is the r -th frame of utterance F^u , and $P_\Lambda(F^{ur} | S_u)$ and $P_\Lambda(F^{ur} | \hat{S}_u)$ are the likelihood scores for the target string S_u and the most competing string, \hat{S}_u . Plugging Eq-(4.8) and Eq-(4.9) into Eq-(4.7), the final objective function to minimize for SME is:

$$L^{SME}(\rho, \Lambda) = \frac{\lambda}{\rho} + \frac{1}{U} \sum_{u=1}^U \left[\rho - d(F^u, \Lambda) \right]_+. \quad (4.10)$$

Similar to MCE, when adopting SME for intraEnv training, each environment-specific HMM set is first trained with ML and then refined by SME. Since we only consider the mean parameters in the ESSEM framework, the objective function of SME refinement is:

$$L^{SME}(\rho, \mathbf{V}_p) = \frac{\lambda}{\rho} + \frac{1}{U} \sum_{u=1}^U \left[\rho - d(F_p^u, \mathbf{V}_p) \right]_+, \quad p=1, \dots, P, \quad (4.11)$$

where F_p^u is the u -th training utterances in the p -th environment.

4.2.2 Inter-Environment (InterEnv) Training

For the interEnv training mode, we consider each environment, accordingly its super-vector, as a particular class in the ESS space. Among the discriminative training methods, we select MCE training for intraEnv training because the misclassification measure of MCE can represent a probabilistic distance between two classes. For the MCE-based interEnv training, we collect speech data $F_{train} = \{F_1, \dots, F_P\}$ of a total of U utterances for P different environments and define the objective function as:

$$L(\Omega_{\mathbf{V}}) = \frac{1}{U} \sum_{u=1}^U \frac{1}{1 + \exp(-\gamma d(F_{train}^u, \Omega_{\mathbf{V}}, \Psi) + \theta)}, \quad (4.12)$$

where the misclassification measure $d(\cdot)$ is defined as:

$$d(F_{train}^u, \Omega_{\mathbf{V}}, \Psi) = -\tilde{g}(F_{train}^u, \Omega_{\mathbf{V}}, \Psi, W_c) + \tilde{G}(F_{train}^u, \Omega_{\mathbf{V}}, \Psi), \quad (4.13)$$

with

$$\tilde{G}(F_{train}^u, \Omega_{\mathbf{V}}, \Psi) = \frac{1}{\eta} \log \left\{ \frac{1}{N} \sum_{n=1}^N \exp[\eta \times \tilde{g}(F_{train}^u, \Omega_{\mathbf{V}}, \Psi, W_n)] \right\}, \quad (4.14)$$

where W_c is again the given correct transcription. $\{W_1, \dots, W_N\}$ are the N -best decoded competing word sequences. In the optimization process, we know the target environment to any segment of the training data, and we generate W_n by using the HMMs for the n -th most competitive environment to that target environment. Parameters in the ESS space are then updated iteratively by minimizing the objective function:

$$\Omega_{\mathbf{V}}(t+1) = \Omega_{\mathbf{V}}(t) - \kappa \nabla L(\Omega_{\mathbf{V}}) |_{\Omega_{\mathbf{V}} = \Omega_{\mathbf{V}}(t)}. \quad (4.15)$$

After performing both intraEnv and interEnv training on the ESS space for several iterations, we obtain an MCE-refined ESS space. Figure 4.3 illustrates the ESS spaces with ML, intraEnv, and intraEnv followed by interEnv training procedures, respectively.

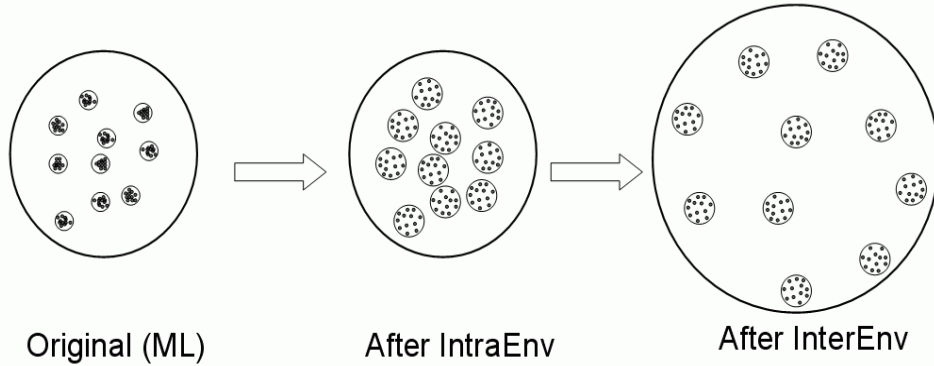


Figure 4.3. Discriminative training to increase environment space discrimination.

4.3 Experiments

In the following sections, we evaluate the performance of ESSEM on the Aurora-2 database [86]. The Aurora-2 database is particularly suited for evaluating the ESSEM approach because it provides speech data from different speakers and speaking environments. Meanwhile, testing sets in Aurora-2 cover a wide range of different environments. Some testing environments (testing environments in test Set B and Set C) contain distortions (noise types and additional channel distortions) that are not included in the training set.

In this section, we first introduce the experimental setup. Two types of frameworks with different front-end and back-end configurations are designed to have a more complete evaluation on the ESSEM approach. Then, we present the testing results of ESSEM with different offline techniques and their combinations.

4.3.1 Experimental Setup

We selected the Aurora-2 database [86] for both training and testing. The multicondition training set was used both to train HMMs and to build the ESS spaces. The training set includes 17 different speaking environments that are originated from the same four types of noise as in test Set A, at four SNR levels: 5dB, 10dB, 15dB, and 20dB, along with the clean condition. We further divided the training set into two gender-specific subsets and thereby obtained 34 (17×2) speaker and speaking environments. We tested recognition on the complete evaluation set of Aurora-2 that consists of 70 different testing environments, with 1001 testing utterances in each environment. We presented the recognition results in word error rate (WER). To compare the performances of the proposed ESSEM approach using different front-end and back-end configurations, we implemented the following two

sets of testing frameworks:

- **Framework-1:** Each speech frame was characterized by 39 coefficients consisting of 13 MFCC with their first and second order derivatives. A cepstral mean subtraction (CMS) [61] was performed for normalization. All digits were modeled by 16-state whole word HMMs with each state characterized by three Gaussian components. The silence and the short pause model were modeled by three states and one state, respectively, with each state characterized by six Gaussian mixture components.
- **Framework-2:** A modified ETSI advanced front-end (AFE) suggested in [66] was used for feature extraction where the log-energy component of each frame was replaced with the C0 coefficient. Every feature vector comprised 13 static plus their first and second order time derivatives. We followed a complex back-end model topology suggested in [66] to train HMMs where there were 20 mixtures per state for the digits and 36 mixtures per state for the silence and short pause.

For both Framework-1 and Framework-2, we implemented gender independent (GI) and gender dependent (GD) systems. For the GI system, a GI HMM set was trained on the multicondition training data, and 34 environment-specific HMM sets were obtained by adapting mean vectors from the GI HMM set to particular environments. Next, we collected the entire set of mean vectors for each of these 34 HMM sets to build an ESS space. For the GD system, two GD HMM sets were first trained. Then 17 environment-specific HMM sets for one GD HMM set were obtained by adapting mean vectors from that GD HMM set to particular environments. Accordingly, two sets of ESS spaces corresponding to the two GD HMM sets were prepared. An additional pair of HMM sets was prepared for automatic gender identification (AGI). In Framework-1, the AGI HMM

set for each gender was modeled with 16 active states with each state characterized by 88 Gaussian mixture components. In Framework-2, we used a simpler configuration for the AGI HMM set. Each gender was modeled with 16 active states with each state characterized by 20 Gaussian mixture components.

In this section, we focus on the ESSEM performance with different ESS configurations and use a linear combination function as the mapping structure throughout all the experiments. We will discuss more about other online methods in the next chapter. As presented in Section 3.4, when using the linear combination function, we estimate the target super-vector, V_Y , by Eq-(3.13).

When conducting the EC technique in ESSEM, we first perform an online cluster selection (CS) process to locate the most relevant cluster, $\Omega_{V^{(t)}}$, whose representative super-vector yields the highest likelihood to the testing data F_Y :

$$\Omega_{V^{(t)}} = \arg \max_{\tilde{t}} P(F_Y | R(\Omega_{V^{(\tilde{t})}})) . \quad (4.16)$$

Then, we use the general formulation of ESSEM in Eq-(3.1) with the selected sub-space $\Omega_{V^{(t)}}$ to estimate the super-vector for the testing environment V_Y :

$$V_Y = \mathbf{G}_{\phi}(\Omega_{V^{(t)}}) . \quad (4.17)$$

Accordingly, the original linear combination in Eq-(3.13) becomes:

$$V_Y = \sum_{p=1}^{P^{(t)}} \hat{w}_p V_p , \quad (4.18)$$

where $P^{(t)}$ is the number of bases in the t -th sub-space. Similarly, the set of weighting coefficients is estimated through:

$$\{\hat{w}_p\}_{p=1}^{P^{(t)}} = \arg \max_{\{w_p\}_{p=1}^{P^{(t)}}} P(F_Y | \sum_{p=1}^{P^{(t)}} w_p \mathbf{V}_p). \quad (4.19)$$

As mentioned earlier, the online process of EC resembles the subset selection methods [18-20]. When comparing to CAT [8] and eigenvoice [9], the major advantage of EC is to use the regional prior knowledge of the ESS space from the EC tree structure. This regional knowledge is critical to dealing with unknown testing conditions. With such regional knowledge, EC only uses the located group of super-vectors (the t -th cluster) to estimate the target super-vector in Eq-(4.19). Moreover, EC locates a representative HMM set through cluster selection. Instead of using the environment-independent (EI) HMM set, we use the located HMM set to collect statistics needed in estimating the weighting coefficients in Eq-(4.19). The representative HMM set provides more accurate statistics estimation than the EI HMM set.

On the other hand, for mixture-based EP, Eq-(3.13) becomes:

$$\mathbf{V}_{Y,s} = \sum_{p=1}^P \hat{w}_p \mathbf{V}_{p,s}, \quad s=1, 2, \dots, S, \quad (4.20)$$

where S is the number of sub-spaces structured by the mixture-based EP technique. Then, the target super-vector is formed by S sets of estimated sub-vectors:

$$\mathbf{V}_Y = [\mathbf{V}_{Y,1}^T, \mathbf{V}_{Y,2}^T, \dots, \mathbf{V}_{Y,S}^T]^T. \quad (4.21)$$

Similarly for feature-based EP, the online estimation becomes:

$$\mathbf{V}_{Y,z} = \sum_{p=1}^P \hat{w}_p \mathbf{V}_{p,z}, \quad z=1, 2, \dots, Z, \quad (4.22)$$

where Z is the number of sub-spaces. We then obtain \mathbf{V}_Y :

$$\mathbf{V}_Y = [\mathbf{V}_{Y,1}^T, \mathbf{V}_{Y,2}^T, \dots, \mathbf{V}_{Y,Z}^T]^T. \quad (4.23)$$

4.3.2 EC on Framework-1 (GI)

In this section, we present ESSEM with EC-structured ESS space. To implement the EC algorithm, we build a two-layered binary tree structure to cluster the 34 speaker and speaking environments into seven groups (one root node, two intermediate nodes, and four leaf nodes). By clustering in a data-driven manner, it is observed that in the first layer, the 34 environments were exactly divided into two groups, each corresponding to one of the two genders. This phenomenon suggests that genders determine significant discriminative power even under unseen noise types and low SNR conditions. In the second layer, another two groups of environments were classified roughly according to high/low SNR levels. To maintain an adequate number of environments in every group, some environments, such as environments at medium SNR levels, were shared across different groups. Finally, each cluster has 12 to 14 different environments. For each cluster in the hierarchy, we use the entire set of training data corresponding to the environments within that cluster to train a set of representative HMMs for the online selection procedure as shown in Eq-(4.16). The same topology is used for training the representative HMMs as that used for training other environment-specific HMMs.

We list the results in Table 4.1. “GI-Baseline” corresponds to the testing results using the multicondition-trained GI HMM set, and we denote “GI-Full” for ESSEM with entire ESS space ($P=34$). “GI-EC(1)” and “GI-EC(2)” are for ESSEM with EC-structured ESS spaces based on a one-layer (cluster number $C=3$) and two-layer (cluster number $C=7$) tree structures, respectively. From Table 4.1, we first observe that the conventional ESSEM approach, “GI-Full”, can already give a significant performance improvement over the “GI-Baseline”. Next, it is clear that “GI-EC(2)” can achieve better performance

over “GI-EC(1)”, and “GI-EC(1)” gives better performance than “GI-Full”, among “Clean”, “0dB-20dB” and “-5dB” conditions. Therefore, we confirm that for the EC algorithm, a better tree structure can produce more significant performance improvements.

Table 4.1. WER (%) for baseline and ESSEM plus EC with different tree structure on Framework-1 (GI).

Test conditions	Clean	0dB-20dB	-5dB
GI-Baseline	1.68	11.01	73.08
GI-Full	1.33	9.25	69.34
GI-EC(1)	1.29	8.91	67.69
GI-EC(2)	1.28	8.81	67.43

For the Aurora-2 evaluation, we are more interested in the results from SNR 0dB to 20dB conditions. Therefore, we listed the average WER for each SNR condition in Table 4.2. In addition to WER, a statistical hypothesis test is usually used to verify whether a method is significantly better than another one. Here, we adopt the dependent t-Test (for matched-pair samples) for the hypothesis test [91, 92]. The dependent t-Test is especially suitable for the Aurora-2 evaluation because: 1) each testing condition has a large amount of testing data (1001 utterances, more than 3000 words), so the measure of the average WER is reliable; 2) two methods have matched-pair sequences of samples, so a pair-wised testing of two methods is reasonable. For the dependent t-Test, we consider H_0 as “method two is not better than method one”, and H_1 as “method two is better than method one”. In Aurora-2, each SNR condition has 10 results (10 pair-wised samples for t-Test). We list the corresponding P-values for all SNR conditions in Table 4.2.

From Table 4.2, we first find that EC(2) achieves lower WER than EC(1) in every SNR condition. Next, we observed that the P-values are 0.013 and 0.051, respectively, for 20dB and 0dB conditions. The small P-values imply consistent improvements of

EC(2) over EC(1). We thus claim EC(2) is significantly better than EC(1) in these two conditions. However, for 15dB, 10dB, and 5dB conditions, although WERs are reduced, the corresponding P-values are relatively large. These observations verify by further using a high/low SNR layer in building the tree structure for EC, ESSEM can better model the very low SNR or very high SNR conditions, while improvements of the medium SNR conditions may not be prominent.

Table 4.2. WER (%) and P-value for two different EC ESS spaces on Framework-1 (GI).

dB	WER		P-value
	GI-EC(1)	GI-EC(2)	EC(2) vs. EC(1)
20	1.61	1.58	0.013
15	2.06	2.04	0.308
10	3.54	3.52	0.341
5	8.74	8.68	0.105
0	28.58	28.24	0.051

4.3.3 EC on Framework-1 (GD)

In the GD system, we used every incoming testing utterance to: 1) determine the speaker’s gender; 2) select a GD HMM set and its corresponding ESS space; 3) perform ESSEM in an unsupervised self-adaptation manner; 4) test recognition with the ESSEM-adapted acoustic models. Similar to the GI system, we compared one-layer and two-layer tree structures and listed their results as “GD-EC(1)” and “GD-EC(2)”, respectively, in Table 4.3. Since the gender identity was determined by the AGI unit beforehand, the EC algorithm did not need an online cluster selection process as shown in Eq-(4.16) for the first layer. To have a fair comparison, we used the AGI process followed by a speaking environment cluster selection to locate a representative HMM set. Then, we directly used the located HMM set to test recognition for the baseline and

denoted the results as “GD-Baseline” in Table 4.3. In Table 4.4, we list the detailed WERs and P-values of “GD-EC(2)” versus “GD-EC(1)” for all SNR conditions.

We observe similar results to the GI system. From Table 4.3, “GD-EC(1)” and “GD-EC(2)” provide 7.88% (8.63% to 7.95%) and 8.57% (8.63% to 7.89%) WER reductions, respectively, over “GD-Baseline” in “0dB-20dB” condition. Next, by comparing “GD-EC(1)” and “GD-EC(2)” in Table 4.4, “GD-EC(2)” provides better performance almost in every SNR condition, and the improvement under very high SNR (20dB) and low SNR (0dB, 5dB) conditions are more significant.

Table 4.3. WER (%) for ESSEM using two EC ESS spaces on Framework-1 (GD).

Test conditions	Clean	0dB-20dB	-5dB
GD-Baseline	1.15	8.63	69.59
GD-EC(1)	1.08	7.95	66.97
GD-EC(2)	1.07	7.89	66.84

Table 4.4. WER (%) and P-value for two different EC ESS spaces on Framework-1 (GD).

dB	WER		P-value
	GD-EC(1)	GD-EC(2)	EC(2) vs. EC(1)
20	1.16	1.14	0.075
15	1.62	1.62	0.422
10	2.86	2.83	0.262
5	7.59	7.49	0.048
0	26.55	26.34	0.037

4.3.4 EC+EP on Framework-1 (GD)

Finally, we present the ESSEM performance with EC followed by EP structuring on the ESS spaces. We used the same two-layer hierarchical tree structure for EC in the previous section followed by mixture-based and feature-based EP techniques. Again, we use a linear combination function for the online super-vector estimation. For mixture-based EP, the online estimation in Eq-(4.18) becomes:

$$\mathbf{V}_{Y,s} = \sum_{p=1}^{P^{(t)}} \hat{w}_p \mathbf{V}_{p,s}, s=1, 2 \dots S, \quad (4.24)$$

and for feature-based EP, the online estimation now becomes:

$$\mathbf{V}_{Y,z} = \sum_{p=1}^{P^{(t)}} \hat{w}_p \mathbf{V}_{p,z}, z=1, 2 \dots Z, \quad (4.25)$$

where $P^{(t)}$ is the total number of bases in the selected t -th sub-space.

The result to be compared with is the first stage EC algorithm alone as “GD-EC(2)” in Table 4.3. The two types of two-stage structured ESS spaces were reported in Table 4.5 as “GD-EC(2)+EP(M)” and “GD-EC(2)+EP(F)” for using mixture-based and feature-based EP in the second stage, respectively. For mixture-based EP, we compared recognition performances using different clustering techniques. Among them, a hierarchical tree structure clustering method as suggested in [76] achieved the best performance. When using a hierarchical tree in mixture-based EP, we first constructed a tree structure based on a set of reference HMMs. In our implementation, we used the representative HMM set in each node from the EC stage to build the EP hierarchical tree. Each node in the EP tree structure, from the root node to leaf nodes, included a group of Gaussian mixture components. We built the EP tree by using a top-down k-means clustering algorithm and the Mahalanobis distance as a distance measure between Gaussians. In the offline phase, the original ESS spaces were partitioned into several sub-spaces by following these EP hierarchical trees. In the online phase, a searching process was conducted beforehand to find a node with a sufficient amount of adaptation statistics from leaf nodes to root node in the EP tree structure. Therefore, the total number of sub-vectors S in Eq-(4.24) was not predefined but determined based on the amount of adaptation data. For feature-based EP, we segmented each super-vector into three

sub-vectors for three different types of coefficient components, namely, 13 static, 13 first and 13 second order time derivatives MFCCs. Then, we built three sub-spaces for three types of components. By comparing “GD-EC(2)” in Table 4.3 with “GD-EC(2)+EP(M)” and “GD-EC(2)+EP(F)” in Table 4.5, we confirm that both the two types of EP techniques can produce better performance than the one-stage EC algorithm alone.

Table 4.6 lists WERs of “GD-EC(2)+EP(M)” and “GD-EC(2)+EP(F)” and P-values of them versus “GD-EC(2)”. From Table 4.4 and Table 4.6, it is clear that the further improvements of EP(M) and EP(F) mainly come from SNR 0dB condition; both improvements at SNR=0dB are significant (P-values=0.046 and 0.022, respectively). We also noted that in some conditions, “GD-EC(2)” achieves lower WERs than “GD-EC(2)+EP(M)” and “GD-EC(2)+EP(F)”. For such cases, we estimate the P-values by hypothesizing “GD-EC(2)” is better than “GD-EC(2)+EP(M)” or “GD-EC(2)+EP(F)”. For example at SNR=15dB, the P-value of “GD-EC(2)+EP(M)” versus “GD-EC(2)” is 0.287. With such a large P-value, we can not claim that “GD-EC(2)+EP(M)” is worse than “GD-EC(2)” even though “GD-EC(2)” provides lower WER.

Table 4.5. WER (%) for ESSEM using EC+EP ESS spaces on Framework-1 (GD).

Test conditions	Clean	0dB-20dB	-5dB
GD-EC(2)+EP(M)	1.05	7.87	66.62
GD-EC(2)+EP(F)	1.05	7.84	66.82

Table 4.6. WER (%) and P-value for two types of EP ESS spaces on Framework-1 (GD).

dB	WER	P-value	WER	P-value
	GD-EC(2)+EP(M)	vs. GD-EC(2)	GD-EC(2)+EP(F)	vs. GD-EC(2)
20	1.14	0.453	1.14	0.339
15	1.63	0.287	1.64	0.247
10	2.83	0.411	2.83	0.470
5	7.50	0.323	7.49	0.432
0	26.22	0.046	26.10	0.022

4.3.5 IntraEnv and InterEnv Training on Framework-2 (GI)

For the following experiments in this section, we present ESSEM using ESS spaces with and without the intraEnv and interEnv training. We follow the procedure of Figure 4.3 and conduct two sets of experiments: 1) intraEnv, and 2) intraEnv followed by interEnv training. After the intraEnv or intraEnv+interEnv training, we apply the EC algorithm with a two-layer tree-structure, as presented in Section 4.3.2, to structure the ESS spaces. A same online mapping function as indicated in Eq-(4.18) is used for the different ESS spaces. Table 4.7 lists the GI system results. “GI-Baseline” is baseline using the GI HMMs only [93]. “GI-ML”, “GI-intraEnv”, and “GI-intraEnv+interEnv” represent ESSEM using the ESS spaces trained by ML, intraEnv, and intraEnv followed by interEnv, respectively. These results correspond to the left, middle, and right panels in Figure 4.3. Table 4.8 lists average WERs of “GI-intraEnv” and “GI-intraEnv+ interEnv” and P-values of “GI-intraEnv+interEnv” vs. “GI-intraEnv” in every SNR condition.

From Table 4.7, it is clear that ESSEM with the original ML-trained ESS space already achieved better performance of 12.85% (6.46% to 5.63%) relative WER reduction over “GI-Baseline”. By comparing “GI-intraEnv”, and “GI-ML”, we observed that “GI-intraEnv” produces better performance than “GI-ML”. Therefore, we verify that intraEnv training can refine ESS spaces and enhance overall ESSEM performance. Finally, by comparing “GI-intraEnv” and “GI-intraEnv+interEnv”, we confirm that intraEnv followed by interEnv training provides further improvements over intraEnv training alone. From Table 4.8, we see that after interEnv training, ESSEM achieves better performance for SNR 20dB to 5 dB conditions, while ESSEM gives no improvement for SNR 0dB condition.

Table 4.7. WER (%) for ESSEM using the ESS spaces refined by intraEnv and interEnv training on Framework-2 (GI).

Test conditions	Set A	Set B	Set C	Overall
GI-Baseline	5.92	6.69	7.11	6.46
GI-ML	5.12	6.07	5.78	5.63
GI-intraEnv	4.94	5.61	5.83	5.39
GI-intraEnv+interEnv	4.93	5.58	5.83	5.37

Table 4.8. WER (%) and P-value for intraEnv and interEnv training on Framework-2 (GI).

dB	WER		P-value
	intraEnv	intraEnv+interEnv	intraEnv+interEnv vs. intraEnv
20	0.54	0.51	0.079
15	0.87	0.85	0.082
10	2.09	2.04	0.093
5	5.53	5.47	0.094
0	17.90	17.95	0.373

4.3.6 IntraEnv and InterEnv Training on Framework-2 (GD)

Next, we demonstrated the ESSEM results on the GD system. Table 4.9 lists ESSEM results with ML-trained and MCE-trained ESS spaces as “GD-ML” and “GD-intraEnv+interEnv”, respectively. We followed the procedure in Section 4.3.3 to obtain the baseline and denoted it as “GD-Baseline”. We also list the average WERs and P-values in Table 4.10. Similar to the GI case, ESSEM with an MCE-trained ESS space is better than that with an ML-trained ESS space. From Table 4.10, we observed that after interEnv and intraEnv training, ESSEM is significantly improved among SNR 20dB to 5dB, while an insignificant degradation is shown in SNR 0dB (P-value= 0.484). From Table 4.7 and Table 4.9, we can see the major improvements come from Set B, which consists of conditions that are not involved in the training set. This observation supports our claim that by using MCE, the coverage of the ESS space is broadened, and performance can be improved, especially for conditions under unseen noise types. However, no improvement

is achieved in testing Set C, where an additional channel distortion is involved. Moreover, from Table 4.8 and Table 4.10, the MCE training gives no significant improvement for SNR 0dB condition. This should be a limitation of MCE training that aims at increasing distance among modeling units only according to the available training data. We can overcome this limitation by including more different environments in the training set or using a more complex online mapping function [17].

Table 4.9. WER (%) for ESSEM using the ESS spaces refined by intraEnv and interEnv training on Framework-2 (GD).

Test conditions	Set A	Set B	Set C	Overall
GD-Baseline	5.11	5.38	6.56	5.51
GD-ML	4.72	5.21	5.60	5.09
GD-intraEnv+interEnv	4.64	4.99	5.64	4.98

Table 4.10. WER (%) and P-value for intraEnv and interEnv training on Framework-2 (GD).

dB	WER		P-value
	ML	MCE	MCE vs. ML
20	0.53	0.47	0.002
15	0.81	0.76	0.008
10	1.95	1.79	0.003
5	5.26	4.98	0.009
0	16.91	16.92	0.484

4.3.7 MLLR and MAPLR on Framework-2 (GD)

We also compared ESSEM performance with other robust approaches on the Aurora-2 task. We reported the MLLR [6] and MAPLR [72] results. Since we tested performance in a per-utterance unsupervised mode, the adaptation data was quite limited. Therefore, simple diagonal affine transformations were adopted for both MLLR and MAPLR. During testing, we first used the AGI process followed by a speaking environment cluster

selection as shown in Eq-(4.16) to locate one set of HMMs. Then, we further adapted the parameters of the selected HMMs to match the testing condition. To have a fair comparison with ESSEM, we did not adapt variance parameters of the HMMs. For MAPLR, we chose a matrix variate normal prior density [72]. The hyperparameters of the prior density were estimated from the multicondition training set. To achieve better performance, each node in the two-layer tree structure had its own set of hyperparameters. The cluster selection procedure not only located an HMM set but also chose a set of hyperparameters that are more relevant to the testing condition. Table 4.11 lists results for both approaches under three different testing sets. The detailed average WERs of MLLR and MAPLR and P-values of “GD-intraEnv+interEnv” versus them in Table 4.9 are listed in Table 4.12.

From Table 4.9 and Table 4.11, we found “GD-intraEnv+interEnv” achieves better performance than MLLR and MAPLR in Set A, Set B, and Overall conditions. However, MLLR and MAPLR give better performance than ESSEM in Set C. As mentioned earlier, this should be a limitation of MCE training and can be enhanced by using a better online mapping function [17]. From Table 4.10 and Table 4.12, we found that ESSEM achieves better performance under noisier conditions (0dB and 5dB). We also used the dependent t-Test to estimate the P-values for the overall 50 testing conditions. The P-values are 0.008 and 0.010 for “GD-intraEnv+interEnv” versus MLLR and MAPLR, respectively. The small P-values confirm that “GD-intraEnv+interEnv” is better than both MLLR and MAPLR for this testing task.

Table 4.11. WER (%) for MLLR and MAPLR on Framework-2 (GD).

	Set A	Set B	Set C	Overall
MLLR	4.88	5.22	5.53	5.14
MAPLR	4.87	5.13	5.54	5.11

Table 4.12. WER (%) and P-value for MLLR and MAPLR.

dB	WER	P-value	WER	P-value
	MLLR	ESSEM vs. MLLR	MAPLR	ESSEM vs. MAPLR
20	0.48	0.415	0.48	0.393
15	0.76	0.426	0.76	0.484
10	1.86	0.039	1.81	0.246
5	5.21	0.004	5.11	0.036
0	17.42	0.046	17.40	0.035

4.3.8 SME-based IntraEnv Training on Framework-2 (GD)

In this set of experiments, we present ESSEM performance with the ESS space refined by SME-based intraEnv training. Table 4.13 lists results of SME-based intraEnv training. We followed the same procedure in Section 4.3.6 to test baseline and ESSEM and list their results as “GD-Baseline(SME)” and “GD-intraEnv(SME)+interEnv”, in Table 4.13, respectively. To have a clear comparison, we list the average WERs and P-values of MCE-based and SME-based intraEnv training for each SNR condition in Table 4.14.

After comparing Table 4.9 and Table 4.13, we first observe that the baseline system is improved with SME-based intraEnv training. Moreover, ESSEM with the SME-based intraEnv training achieves clear improvement over the MCE-based intraEnv training. We can see that the major improvement comes from test Set C. The results suggest that SME-based intraEnv overcomes the limitation of MCE-based intraEnv that archives limited improvement for the conditions containing distortions not included in the training set. From Table 4.14, we can see that SME-based intraEnv training gives better performance than MCE-based intraEnv training almost in every SNR condition. Especially, the improvements are clearer under lower SNR conditions (SNR=0dB and 5dB), and the corresponding P-values at these low SNR conditions are very small (P-value=0.003 and 0.005).

Table 4.13. WER (%) for ESSEM using the ESS spaces refined by SME-based intraEnv and MCE-based interEnv training on Framework-2 (GD).

Test conditions	Set A	Set B	Set C	Overall
GD-Baseline(SME)	5.05	5.31	6.31	5.41
GD-intraEnv(SME)+interEnv	4.58	4.93	5.48	4.90

Table 4.14. WER (%) and P-value for MCE-based and SME-based intraEnv training on ESS spaces on Framework-2 (GD).

dB	WER		P-value
	MCE	SME	SME vs. MCE
20	0.47	0.46	0.122
15	0.76	0.76	0.397
10	1.79	1.76	0.159
5	4.98	4.88	0.005
0	16.92	16.64	0.003

4.3.9 ESS Space Analysis: IntraEnv and InterEnv Training

In addition to recognition results as presented in the previous sections, we used two measurements—1) separation of parameters in one HMM set for a particular environment; 2) difference between two HMM sets for two different environments—to further investigate the discrimination properties of the ESS spaces. We used the first and second measurements to examine the intraEnv and interEnv training, respectively. The first measurement adopts the divergence distance to estimate each pair of Gaussian components and calculate an accumulated measurement for one particular environment. Details about this measurement can be found in [27]. To measure the second distance, we adopted the GLLR plots [94] as presented in Section 2.3.

We performed the two measurements by using environment-specific HMMs before converting them into super-vector forms. In preliminary experiments, we observed similar results for all the environments collected in the ESS spaces for both GI and GD systems. Therefore, we only select and present one particular environment: “subway

noise, 10dB SNR, male speakers” from the GI system as the target environment here in this thesis. Table 4.15 illustrates accumulated divergence distances within HMM sets before intraEnv training (ESS space is trained by ML only), MCE-based intraEnv training, and SME-based intraEnv training. The comparison between without and with intraEnv training corresponds to the left versus the middle panels in Figure 4.3. From Table 4.15, it is clear that after MCE-based intraEnv training, the separation between parameters in the HMM set is increased over without intraEnv training. Additionally, it is clear that SME-based intraEnv training can further increase the separation in the HMM set over MCE-based intraEnv training.

Next, we present a GLLR in Figure 4.4 to indicate the distance between the target environment and its competing cohort environments before and after interEnv training (the middle versus the rightmost panels in Figure 4.3). To have a clear comparison, we used a Gaussian distribution to approximate the histograms and indicate the means for all panels in Figure 4.4. From Figure 4.4, we verify that the distance between the target and its competing environments is enhanced after interEnv training. Along with the improvements presented in Table 4.7 and Table 4.9, we conclude that with discriminative training, we can enhance the discrimination power of the ESS space and thereby enable ESSEM to achieve a better performance. Finally, by comparing Table 4.9 and Table 4.13, we verify by using a better discriminative training criterion (SME) for intraEnv training, the discriminative power of the ESS space can be enhanced. The overall ESSEM performance can accordingly be further improved.

Table 4.15. Divergence distance within one set of environment-specific HMMs.

Test HMMs	Before intraEnv	MCE-based intraEnv	SME-based intraEnv
Distance	72.80	74.44	75.34

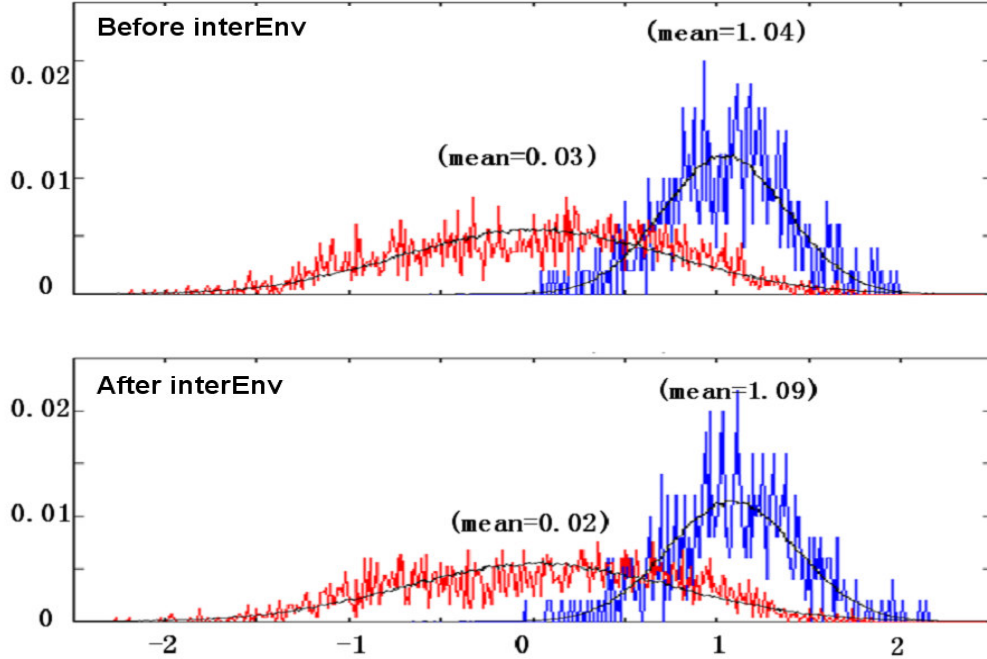


Figure 4.4. Separation between environment-specific models.

4.3.10 Overall Combination: IntraEnv+InterEnv+EC+EP

Finally, we integrated EC, EP, and intraEnv and interEnv training techniques to refine the ESS space. We first used both intraEnv and interEnv training to increase the discrimination and then applied EC followed by EP to well structure the ESS space. In this set of experiments, we adopt MCE for both intraEnv and interEnv training. After the discriminative training, we used the same two-layer hierarchical tree structure for EC followed by mixture-based and feature-based EP. The result of using the first stage of EC alone is “GD-intraEnv+interEnv” in Table 4.9. The two types of two-stage ESS spaces were listed in Table 4.16 as “MCE+EC+EP(M)” and “MCE+EC+EP(F)” for mixture-based and feature-based EP, respectively. Again for mixture-based EP, we used a hierarchical tree structure to clustering Gaussian mixture components. For feature-based EP, we partitioned each super-vector according to different types of coefficient components, namely, 13 static, 13 first and 13 second order time derivatives of AFE

features [94]. We also list the average WERs and P-values for the two overall combination techniques in Table 4.17. The P-values are estimated based on the two combination methods versus “GD-intraEnv+interEnv” in Table 4.9.

From the results in Table 4.9 and 4.16, we find both the two combination techniques provide better performance than “GD-intraEnv+interEnv”. From Table 4.17, similar observations from Table 4.6 are observed. The two combination methods provide significant improvements under low SNR conditions. We further used dependent t-Test to estimate P-values for the overall 50 testing conditions. The corresponding P-values are 0.066 and 0.019, respectively, for “MCE+EC+EP(M)” and “MCE+EC+ EP(F)” versus “GD-intraEnv+interEnv”. Therefore, we can claim that both the two combination methods are better than “GD-intraEnv+interEnv”. Since the concepts of mixture-based and feature-based EP techniques are different, we tested recognition by using an integration of the two EP techniques. However, the integration did not give further improvement over “MCE+EC+EP(F)” alone. We believe that it is due to the limited adaptation statistics needed for the per-utterance compensation mode. Using too many free parameters may have degraded the achievable performance.

Table 4.16. WER for ESSEM with combined offline techniques Framework-2(GD).

	Set A	Set B	Set C	Overall
MCE+EC+EP(M)	4.62	4.99	5.60	4.96
MCE+EC+EP(F)	4.62	4.94	5.56	4.94

Table 4.17. WER (%) and P-value for the combined offline techniques.

dB	WER	P-value	WER	P-value
	MCE+EC+EP(M)	vs. MCE+EC	MCE+EC+EP(F)	vs. MCE+EC
20	0.44	0.209	0.47	0.303
15	0.76	0.463	0.75	0.332
10	1.76	0.273	1.79	0.402
5	4.99	0.383	4.93	0.084
0	16.83	0.021	16.73	0.019

4.4 Summary

In this chapter, we propose techniques to refine the ESS spaces for ESSEM and thereby enhance its overall performance. We propose EC and EP to structure the ESS space well and adopt `intraEnv` and `interEnv` training to improve environment discriminative power. ESSEM with different offline techniques was evaluated on the Aurora-2 task in an unsupervised compensation (self-learning) mode. We prepared two sets of testing framework. Framework-1 uses simpler front-end and back-end, and Framework-2 adopts more complex front-end and back-end. The experimental results from these two frameworks exactly direct to the same conclusions. First, with EC, ESSEM achieves better performance, especially under higher SNR (20dB) and lower SNR (0dB and 5dB) conditions. When integrated with EP, ESSEM gives further improvement, again under higher SNR (20dB) and lower SNR (0dB and 5dB) conditions. Next, for `intraEnv` and `interEnv` training algorithms, recognition results indicate that ESSEM achieves better performance with an MCE-trained ESS space than an ML-trained ESS space on both GI and GD systems. More improvements were observed from 5dB to 20dB conditions, where distortions embedded in these testing conditions are included in the training set. Next, we confirm that by using SME-based `intraEnv` training, we can further enhance the discrimination within each super-vector and therefore improve the ESSEM performance. To directly investigate the effects of `intraEnv` and `interEnv` training, we adopt two measurements. We confirm that `intraEnv` training enhances the separation between parameters within an HMM set for a particular environment, while `interEnv` training increases the difference across environments. Finally, we integrate all the offline techniques, namely, MCE-based `intraEnv` and `interEnv` training with EC and EP, to

obtain our best environment configuration. When applying the integration method on the ESS space, ESSEM can provide an average of 10.34% relative WER reduction (5.51% to 4.94% WER) over the baseline result. When using SME-based intraEnv training to refine the ESS space, we can get even better performance of 4.90% WER. This refined ESS space stands for our current best environment configuration and will be further improved with online methods in the next chapter.

CHAPTER 5

ONLINE ESTIMATION PROCESS ENHANCEMENT

In the previous chapter, we propose techniques to refine the ESS space in the offline stage. We propose EC and EP algorithms to structure the ESS space well and use the discriminative training to enhance the discriminative power of the ESS structure. In this chapter, we focus on the issues of the online super-vector estimation. First, we study different online methods to improve the precision of environment characterization. Second, we study techniques to reduce the complexity of the environment configuration and accordingly enhance the efficiency of the online operation.

5.1 Improving Estimation Precision

In this section, we study five directions to enhance the precision of online super-vector estimation—1) different forms of online mapping structures, 2) multiple cluster matching algorithm, 3) weighted N -best information, 4) cohort selection, 5) online environment space adaptation. The testing results of ESSEM with these online methods along with their combinations will be presented in Section 5.4.

5.1.1 Mapping Structure Precision

Intuitively, by using a more complex mapping function in Eq-(3.1), ESSEM can better characterize the unknown testing environments. However, too many free parameters to estimate in a complex function may induce an over-fitting problem. To determine the best form of mapping structure for a particular task, we do not only take into account the type of distortion that we are dealing with but also need to consider the amount of available adaptation (compensation) data.

A. Best First

The simplest form of mapping function is the best first method [17]. The best first method determines V_Y by locating the most matched super-vector in the ESS space:

$$V_Y = \arg \max_p P(F_Y | V_p), p=1, 2, \dots, P, \quad (5.1)$$

When the ESS space provides a good coverage of different environments, this simple best first method should already give satisfactory performance. However, two problems may occur when we use this method. First, an exhaustive search to locating one most matched condition is both time and computation consuming; such consumptions are undesirable in real world ASR applications. Second, the closest super-vector, V_p , may still be far from the real target super-vector, V_Y , especially when the testing environment is very different from any vector in the available collection of super-vectors in the ESS space. Experimental results verified that the best first method cannot achieve as good performance as a linear combination method [17]. Extensions of the best first method can overcome the above two problems: 1) we can simplify the exhaustive search by a hierarchical search with a tree structure environment clustering; 2) we can implement a two-stage online process, i.e., a best first process followed by a second-stage stochastic matching, to enhance performance. A good example is tree-structured piecewise-linear transformation (PLT) [11]. The PLT approach first locates the best matched environment with a tree structure and then adjusts parameters of HMM set for the located environment through a set of affine transformations.

B. Linear Combination

An interpolation method can be a better method than the best first method. We have presented to use a linear combination function as an online mapping function in Eq-(3.13).

The corresponding solution to Eq-(3.13) is provided in Eq-(3.14).

C. Linear Combination with a Correction Bias

Next, we incorporate a correction bias, \hat{b} , into the interpolation method in Eq-(3.13).

Now we have:

$$\mathbf{V}_Y = \sum_{p=1}^P \hat{w}_p \mathbf{V}_p + \hat{b}. \quad (5.2)$$

The weighting coefficients and correction bias can be estimated base on the ML algorithm:

$$\{\{\hat{w}_p\}_{p=1}^P; \hat{b}\} = \arg \max_{\{\{w_p\}_{p=1}^P; b\}} P(F_Y | \sum_{p=1}^P w_p \mathbf{V}_p + b). \quad (5.3)$$

D. Complex Forms of Mapping Structure

We also tested other mapping functions with more complex structures than the three functions discussed above. First, we use a linear combination of affine transformations as the mapping structure and estimate the target super-vector, \mathbf{V}_Y :

$$\mathbf{V}_Y = \sum_{p=1}^P (\hat{A}_p \mathbf{V}_p + \hat{b}_p). \quad (5.4)$$

We calculate the affine transformation set, $\{\hat{A}_p, \hat{b}_p\}_{p=1}^P$, based on the ML algorithm:

$$\{\hat{A}_p, \hat{b}_p\}_{p=1}^P = \arg \max_{\{A_p, b_p\}_{p=1}^P} P(F_Y | \sum_{p=1}^P (A_p \mathbf{V}_p + b_p)), \quad (5.5)$$

With some simplification assumptions, we estimate the m -th mean vector in \mathbf{V}_Y by:

$$\mu_{Y,m} = \sum_{p=1}^P (\hat{A}_{p,m} \mu_{p,m} + \hat{b}_{p,m}), \quad (5.6)$$

where $\mu_{p,m}$ is the m -th mean vector in the p -th super-vector, and $\{\hat{A}_{p,m}, \hat{b}_{p,m}\}$ is the p -th set affine transformation. Based on the structure of Eq-(5.6), we tested recognition using

different forms of transformation matrix, $\hat{A}_{p,m}$. When the form of $\hat{A}_{p,m}$ becomes complex, the computation cost grows large. Furthermore, an over-fitting problem may occur if the adaptation data is insufficient.

Second, we use a second-order polynomial function as the mapping function. In such a case, the m -th mean vector is estimated by:

$$\mu_{Y,m} = \sum_{p=1}^P \hat{w}_p \mu_{p,m}^2 + \sum_{p=P+1}^{2P} \hat{w}_p \mu_{p,m} + \hat{b}. \quad (5.7)$$

Again, we might encounter the over-fitting problems because the amount of free parameters to be estimated in Eq-(5.7) is large.

5.1.2 Multiple Cluster Matching (MCM) Precision

In this section, we present a multiple cluster matching (MCM) algorithm to reduce the performance degradations caused by a possible poor cluster selection process, and thereby, enhance the precision of the super-vector estimation. The basic concept of MCM is similar to that of the ensemble estimator (EE) algorithm [95, 96]. The EE algorithm is developed in the research for sparse representations of signals and usually compared with the subset selection methods [18-20]. Instead of finding a single best subset, the EE algorithm models the target signal with a combination of estimates obtained from multiple subsets. In particular tasks, the subset selection methods generate unstable results [96], and the EE algorithm can provide better performance in those conditions.

We apply the MCM algorithm into the ESSEM framework in the online phase. When we have prepared an EC-structured ESS space in the offline phase, instead of performing a CS procedure to determine the most relevant cluster of environments, we estimate a super-vector for each cluster:

$$\mathbf{V}_{Y^{(c)}} = \mathbf{G}_{\varphi}(\Omega_{\mathbf{V}^{(c)}}), c=1,2,\dots,C. \quad (5.8)$$

Then, the collection of all the estimated C super-vectors forms a new ensemble environment space, $\Omega_{\mathbf{V}_E}$:

$$\Omega_{\mathbf{V}_E} = \{ \mathbf{V}_{Y^{(1)}} \mathbf{V}_{Y^{(2)}} \dots \mathbf{V}_{Y^{(C)}} \}. \quad (5.9)$$

Finally, a stochastic matching process is carried out to estimate the super-vector for the testing condition through a multiple cluster matching (MCM) function, \mathbf{G}_{φ_E} :

$$\mathbf{V}_Y = \mathbf{G}_{\varphi_E}(\Omega_{\mathbf{V}_E}), \quad (5.10)$$

with

$$\varphi_E' = \underset{\varphi_E}{\operatorname{argmax}} P(F_Y | \varphi_E, \Omega_{\mathbf{V}_E}), \quad (5.11)$$

where φ_E stands for the set of nuisance parameters of the MCM function. Similarly, the mapping structure of the MCM function, \mathbf{G}_{φ_E} , can be either the best first method in Eq-(5.1), linear combination function in Eq-(3.13), or linear combination with a correction bias function in Eq-(5.2).

When the number of the training environments grows large, or when the tree structure built by the EC algorithm is complex, we need special strategies to enhance the efficiency of the MCM algorithm. One possible method is to only take account of a subset of clusters in the tree structure. Environments in those clusters have closer acoustic properties to the testing condition. The subset of clusters can be collected by finding those clusters with their representative super-vectors in Eq-(4.16) giving higher likelihood scores to the testing data. Then, we have a new environment space, $\Omega_{\mathbf{V}_E}'$:

$$\Omega_{\mathbf{V}_E}' = \{ \mathbf{V}_{Y^{(1)}} \mathbf{V}_{Y^{(2)}} \dots \mathbf{V}_{Y^{(C)}} \}, \quad (5.12)$$

where C' is smaller than C , and $\Omega_{\mathbf{V}_E'}$ is a sub-space of $\Omega_{\mathbf{V}_E}$. Finally, we can obtain the target super-vector, \mathbf{V}_Y , through:

$$\mathbf{V}_Y = \mathbf{G}_{\varphi_E}(\Omega_{\mathbf{V}_E'}). \quad (5.13)$$

5.1.3 Weighted N -best Information

In an unsupervised adaptation style, we can use the best decoded transcription for stochastic matching. However, the decoded best transcription may not be the ground truth, especially in the severed noisy conditions. Using N -best transcriptions from the decoder is a good way to address this issue, since the ground truth may be embedded in other candidates. By introducing the N -best transcriptions, we can rewrite Eq-(3.2) as:

$$\varphi' = \underset{\varphi}{\operatorname{argmax}} \sum_{n \in N} \lambda_n P(F_Y | \Omega_{\mathbf{V}}, \varphi, W_n), \quad (5.14)$$

where W_n and λ_n are the decoded transcription and weight for the n -th hypothesis, respectively. Based on a study about unsupervised speaker adaptation [97], we adopt the following equation to dynamically determine λ_n :

$$\lambda_n = \frac{\exp[(L_n - L_1) / \eta]}{\sum_{m \in N} \exp[(L_m - L_1) / \eta]}, \quad (5.15)$$

where L_n is the log-likelihood of the n -th hypothesis, and η is a parameter that determines the confidence of the hypotheses. Eq-(5.15) is a general formulation for using N -best transcription. When setting $\eta \rightarrow \infty$, an equal weighting is applied to the N -best hypotheses; when setting $\eta = 0$, only the 1-best hypothesis is used to estimate the nuisance parameters in Eq-(5.14). If we use a small value of η , the best candidate's likelihood will dominate Eq-(5.15). Therefore, we prefer using a value around 12-15 used in discriminative training [26, 27] to scale down the dominating likelihood.

In the following two sections, we present techniques to construct the ESS space that provides better resolution and coverage to model the test conditions—cohort selection and environment space adaptation.

5.1.4 Cohort Selection Precision

In Chapter 4, we demonstrated that using a succinct environment space with higher resolution can aid ESSEM to better characterize unknown testing environments, especially when only limited adaptation data is available [16]. In this section, we study a cohort selection technique [87, 98] to construct a space with good resolution in the online phase. The concept of cohort selection resembles that of the family of subset selection methods [18-20] that find a subset of components from the entire set of components to model a signal of interest. In the implementation aspect, cohort selection can be seen as an extension of the best first function as shown is Eq-(5.1) [17]. However, instead of locating one most matched environment, cohort selection finds N training environments (cohorts) that are closest to the testing environment. In this study, we use the likelihood to measure the closeness. With the selected cohort environments, we build a cohort ESS space, $\Omega_{V_{CH}}$. Finally, we use the stochastic matching algorithm to estimate the target super-vector for the testing condition:

$$\mathbf{V}_Y = \mathbf{G}_\varphi(\Omega_{V_{CH}}). \quad (5.16)$$

The parameters, φ , can be estimated with the EM algorithm.

5.1.5 Online Environment Space Adaptation

As mentioned earlier, ESSEM prepares the ESS space using the available training data in the offline. Thus, the ESS space may have a poor coverage for the testing conditions that contain distortions not included in the training set. This poor coverage limits the ESSEM

performance. Here, we present an environment space adaptation (ESA) algorithm to online build a new ESS space providing better coverage for the testing environments.

Based on the testing utterances, ESA generates a new ESS online by compensating the parameters of the original ESS space. The stochastic matching criterion is used for the compensation process with a mapping function, $\mathbf{G}_\theta(\bullet)$:

$$\Omega_{\mathbf{V}_{ESA}} = \mathbf{G}_\theta(\Omega_{\mathbf{V}}), \quad (5.17)$$

where $\Omega_{\mathbf{V}_{ESA}}$ is the compensated ESS space, θ denotes the nuisance parameters of the mapping function. This new space should provide a better coverage for the testing condition. Finally, we estimate the target super-vector, \mathbf{V}_Y , through stochastic matching:

$$\mathbf{V}_Y = \mathbf{G}_\varphi(\Omega_{\mathbf{V}_{ESA}}), \quad (5.18)$$

Similarly, the set of parameters, φ , of the mapping function can be estimated by the EM algorithm as presented in Eq-(3.2).

5.2 Enhancing Estimation Efficiency

In this section, we introduce the dimensionality reduction techniques to remove redundant components from the ESS spaces and therefore enhance the online estimation efficiency. We study two dimensionality reduction techniques, principal component analysis (PCA) [99] and factor analysis. Both techniques target at reducing dimensionality by forming linear combinations of the entire set of super-vectors. PCA accounts for the variance of the super-vectors, and factor analysis, on the other hand, accounts for the correlations among the super-vectors. PCA has been adopted and studied extensively. The readers are referred to [99] for an in-depth discussion. For factor analysis, we consider the problem as one of combining (or grouping) highly correlated

super-vectors, and it is clear that the clustering methods are applicable to this problem. In this study, we use a bottom-up hierarchical dimensionality reduction (HDR) method [28]. We first determine a similarity function between a pair of super-vectors. Two super-vectors having a high similarity are clearly good candidates to merge to reduce the dimensionality by one. Then, the process repeats and leads to a hierarchical structure. Finally, we collect a smaller number of nodes and clusters from the hierarchy to construct a lower dimensional ESS space.

After applying dimensionality reduction on the original ESS space, we can have a lower dimensionality space with K ($K \leq P$) bases, $\Omega_{\mathbf{V}^{(e)}} = \{V_1^{(e)} V_2^{(e)} \dots V_K^{(e)}\}$. Stochastic matching is then carried out to estimate the super-vector for the testing environment:

$$\mathbf{V}_Y = \mathbf{G}_\varphi(\Omega_{\mathbf{V}^{(e)}}) . \quad (5.19)$$

5.3 Generalized ESSEM

In this chapter, we extend the original ESSEM framework and propose a generalized ensemble speaker and speaking environment modeling (GESSEM). As introduced in Section 2.6, the conventional stochastic matching algorithm first estimates a mapping function. The estimated mapping function then transforms HMMs to match the testing environment. The acoustic likelihood, or equivalently its logarithm form, is used as the objective function as shown in Eq-(2.38). In this study, we extend the stochastic matching algorithm by incorporating more knowledge sources into the objective function and name this extension the generalized stochastic matching (GSM) algorithm.

In this study, we only implement GSM in a supervised adaptation mode. The generalization of GSM to an unsupervised mode can be further studied in the future. In a supervised adaptation mode, we can write the log-likelihood as:

$$L(F_Y, \varphi, \Lambda_X) = \log [P(F_Y | \Lambda_X, \varphi, W_c)], \quad (5.20)$$

where W_c is the given correct transcription corresponding to the adaptation speech data, F_Y . From Eq-(2.38), we now estimate the nuisance parameters by:

$$\varphi' = \underset{\varphi}{\operatorname{argmax}} \{ L(F_Y, \varphi, \Lambda_X) \}. \quad (5.21)$$

The EM algorithm can be used to compute the nuisance parameters φ in Eq-(5.21).

For the GSM algorithm, we generalize the objective function in Eq-(5.21) by using other knowledge sources in addition to the acoustic likelihood. They include likelihood ratio, duration, language model probabilities [100]. In our study, we select the likelihood ratio. In the discriminative training methods, such as MCE [25] and large margin estimation (LME) [101], the likelihood ratio, $D(F_Y, \varphi, \Lambda_X)$, is used as a separation measure representing the distance between the correct and competing hypotheses:

$$D(F_Y, \varphi, \Lambda_X) = \log [P(F_Y | \varphi, \Lambda_X, W_c)] - \log \left[\frac{1}{N} \sum_{n=1}^N P(F_Y | \varphi, \Lambda_X, W_n) \right], \quad (5.22)$$

where N is the number of competing strings, and W_n is the n -th best string. Such N -best strings can be generated by a decoding procedure. When we use likelihood ratio, $D(F_Y, \varphi, \Lambda_X)$, as the objective function, Eq-(5.21) becomes:

$$\varphi' = \underset{\varphi}{\operatorname{argmax}} \{ D(F_Y, \varphi, \Lambda_X) \}. \quad (5.23)$$

Minimum likelihood classification error linear regression (MCELR) uses a similar objective function to Eq-(5.23) and adopts a set of affine transformations as the mapping function. Two types of MCELR were proposed for using two different methods to solve Eq-(5.23). The first type of MCELR estimates the transformations in the ML criterion in the first stage. Then in the second stage, the parameters of transformations are

re-estimated based on a GPD method [102]. The second type of MCELR, on the other hand, estimates affine transformations directly based on the EM algorithm [103]. Both two types of MCELR show good performance in model adaptation.

In the conventional stochastic matching algorithm, the optimization in Eq-(5.21) enhances the discrimination of the target model. On the other hand, the likelihood-ratio-based GSM algorithm increases the confidence interval of the target model through the optimization in Eq-(5.23). Since the two optimizations carry different physical meanings, a combination of them shall achieve additional performance improvement over individual one of them. In this thesis, we provide two methods to combine the two optimizations. First, we perform a two-stage optimization, namely, an optimization process of Eq-(5.21) followed by an optimization process of Eq-(5.23), and repeat the process for several iterations. Second, we combine the two objective functions in Eq-(5.20) and Eq-(5.22) and form a new objective function of $K(F_Y, \varphi, \Lambda_X)$:

$$K(F_Y, \varphi, \Lambda_X) = \alpha L(F_Y, \varphi, \Lambda_X) + (1 - \alpha) D(F_Y, \varphi, \Lambda_X), \quad (5.24)$$

where α is a weighting coefficient. With the new objective function, $K(F_Y, \varphi, \Lambda_X)$, the optimization in Eq-(5.21) becomes:

$$\varphi' = \underset{\varphi}{\operatorname{argmax}} \{ K(F_Y, \varphi, \Lambda_X) \}. \quad (5.25)$$

For the second method, the nuisance parameters are estimated through a joint maximization over a combination score of acoustic log-likelihood and generalized log-likelihood ratio. We call this second method a combination method in the following.

In the previous studies, a confidence score of a combination of likelihood and likelihood ratio has been used to extend the conventional decoder to a hybrid decoder [100]. The hybrid decoder gives better recognition results than the conventional decoder

that uses either likelihood or likelihood ratio score alone. Therefore, it is verified that an integration score of multiple knowledge sources can be used to improve recognition performance. Analogically, by using more knowledge sources in the objective function in Eq-(5.25), GSM should have a better model adaptation result than using a single knowledge source as shown in Eq-(5.21).

Similar to the extension from stochastic matching to ESSEM, we can easily extend the GSM to generalized ESSEM (GESSEM). For GESSEM, we rewrite Eqs-(5.21), (5.22), and (5.24), respectively, to Eqs-(5.26), (5.27), and (5.28):

$$L(F_Y, \Omega_V, \varphi) = \log [P(F_Y | \Omega_V, \varphi, W_c)], \quad (5.26)$$

$$D(F_Y, \Omega_V, \varphi) = \log [P(F_Y | \Omega_V, \varphi, W_c)] - \log \left[\frac{1}{N} \sum_{n=1}^N P(F_Y | \Omega_V, \varphi, W_n) \right], \quad (5.27)$$

$$K(F_Y, \Omega_V, \varphi) = \alpha L(F_Y, \Omega_V, \varphi) + (1 - \alpha) D(F_Y, \Omega_V, \varphi), \quad (5.28)$$

Then, Eq-(3.2) becomes:

$$\varphi' = \underset{\varphi}{\operatorname{argmax}} \{ K(F_Y, \Omega_V, \varphi) \}. \quad (5.29)$$

Similar to the original ESSEM approach, the set of nuisance parameters, φ , in Eq-(5.29) can be estimated by the EM algorithm. Likewise, we can either use a two-stage method or a combination method to optimize the two different types of objective functions.

5.4 Experiments

In this section, we first compare ESSEM performance with different online mapping structures. Then, we test the MCM algorithm on ESSEM. After that, we demonstrate the results of weighted N -best information and the two techniques for online ESS space refinement. We also report our best offline and online configuration that gives highest accuracy on the Aurora-2 task in Section 5.4.6. After that, we present the ESSEM results

using the ESS space with reduced complexity by PCA and HDR. Finally, the GESSEM performance is presented in Section 5.4.10.

5.4.1 Online Mapping Structures on Framework-2 (GD)

In this section, we compare ESSEM performances by using different online mapping structures. We list the recognition results in Table 5.1. In this set of experiments, we use the same offline configuration to that used in Section 4.3.6—MCE-based intraEnv and interEnv training, plus the EC technique. The baseline result is “GD-Baseline” listed in Table 4.9 in Section 4.3.6. The result of “GD-intraEnv+interEnv” in Table 4.9 is also listed in Table 5.1 as “LC”. Table 5.1 lists best first, as indicated in Eq-(5.1) and linear combination with a correction bias, as indicated in Eq-(5.2), as the mapping structures to test recognition. We listed the results in the rows of “Best First” and “LC+bias” in Table 5.1, respectively. We also listed the average WER for each SNR condition of “LC”, “Best First”, and “LC+bias” in Table 5.2. The P-values of “LC” vs. “LC+bias” for all the SNR conditions are further presented in Table 5.2.

Table 5.1. WER (%) for ESSEM with different online mapping structures on Framework-2 (GD).

	Set A	Set B	Set C	Overall
LC	4.64	4.99	5.64	4.98
Best First	4.98	5.22	6.38	5.35
LC+bias	4.62	4.95	5.13	4.85

Table 5.2. WER (%) and P-value for ESSEM with different online mapping structures on Framework-2 (GD).

dB	WER	WER	WER	P-value
	LC	Best First	LC+bias	LC vs. LC+bias
20	0.47	0.49	0.47	0.406
15	0.76	0.75	0.75	0.383
10	1.79	1.85	1.79	0.382
5	4.98	5.57	4.92	0.079
0	16.92	18.11	16.34	0.057

By comparing “LC”, “Best First”, and “LC+bias” in Table 5.1, we can observe two phenomena. First, we note that the best first method gives worse performance than the other two mapping structures. We believe that it is due to the natural limitation of the best first method—the closest super-vector, V_p , may still be far from the real, V_Y , especially when the testing condition is very different from any vector in the collection of super-vectors. However, this limitation can be overcome when the ESS-space is well prepared by incorporating a good coverage of different acoustic conditions. Second, we observe that “LC+bias” achieves the best performance among the three mapping structures. From Table 5.2, we can see the better performance of “LC+bias” comes from the testing under low SNR conditions (0dB and 5dB). Based on the two observations, we confirm that when using a properly specified online mapping function, ESSEM can produce better performance. Moreover, we can also find a major improvement from testing Set C, where a channel distortion is added as another acoustic difference. Since such channel distortion is not included in the training set, the performance improvement suggests that the correction bias can successfully compensate for the mismatch caused by unseen distortion sources.

Next, we list results using two complex mapping structures presented in Section 5.1.1—linear combination of affine transformations as indicated in Eq-(5.4), and second-order polynomial as indicated in Eq-(5.7). We listed the results in the rows of “LC-AF”, and “Second-Poly” in Table 5.3, respectively.

Table 5.3. WER (%) for ESSEM with complex online mapping structures on Framework-2 (GD).

	Set A	Set B	Set C	Overall
LC-AF	4.63	4.95	5.18	4.87
Second-Poly	4.61	4.99	5.16	4.87

By comparing the results of “LC+bias” in Table 5.1 and “LC-AF”, and “Second-Poly” in Table 5.3, we observe that “LC+bias” can provide similar performance to the two complex mapping functions. This result suggests that although complex mapping functions can better characterize testing conditions, the possible over-fitting issue may confine their characterization performance when very few statistics are available. In the following, we limit our discussion on the three linear mapping functions, i.e., best first, linear combination and linear combination with a correction bias.

5.4.2 Multiple Cluster Matching on Framework-2 (GD)

In this section, we present the results of the MCM algorithm. We used the same offline ESS configuration as in Table 5.1 and Table 5.2 and fixed the linear combination with a correction bias as the online mapping structure. Since the total number of nodes in the two-layer tree is not too large ($C=7$), we use all the clusters to perform the MCM algorithm. We test the MCM algorithm with two MCM functions—best first and linear combination with a correction bias. The corresponding results are “LC+bias–BF” and “LC+bias–LC+bias” in Table 5.4. Moreover, we list WERs for different SNR conditions and P-values of “LC+bias–LC+bias” vs. “LC+bias” in Table 5.5.

Table 5.4. WER (%) for ESSEM with MCM on Framework-2 (GD).

	Set A	Set B	Set C	Overall
LC+bias–BF	4.50	4.95	5.02	4.78
LC+bias–LC+bias	4.48	4.95	5.00	4.77

Table 5.5. WER (%) and P-value for ESSEM with MCM on Framework-2 (GD).

dB	WER	WER	P-value
	LC+bias–BF	LC+bias–LC+bias	vs. LC+bias
20	0.48	0.45	0.018
15	0.70	0.69	0.062
10	1.81	1.80	0.441
5	4.82	4.81	0.119
0	16.11	16.10	0.032

From Table 5.4, we found that “LC+bias–LC+bias” provides slightly better performance than “LC+bias–BF”. Next, when comparing “LC+bias” in Table 5.1 with “LC+bias–BF” and “LC+bias–LC+bias” in Table 5.4, we confirm that the MCM algorithm further enhance the ESSEM performance from 4.85% WER to 4.78% WER and 4.77% WER. From Table 5.5, we observe MCM provides significant improvement over “LC+bias” under almost any SNR conditions, except SNR=5dB and SNR=10dB conditions. Therefore, it is verified that MCM can produce better overall ESSEM performance than the EC algorithm.

5.4.3 Weighted N -best Information on Framework-2 (GD)

Next, we present the results of weighted N -best information on the ESSEM framework. In the following sections, we used an ESS space refined by SME-based intraEnv and MCE-based interEnv training. As stated in Section 4.3.8, this offline configuration enables ESSEM achieves the best performance on Aurora-2. To have a clear comparison, we report two baseline results in Table 5.6. For “Baseline(AGI)” in Table 5.6, we directly used the AGI unit to identify speaker’s gender for each testing utterance. Then, the HMM set for the identified gender is used to decode the same testing utterance. For “Baseline(EC)” in Table 5.6, we adopted an environment clustering (EC) tree as presented in Section 4.3.8 to obtain this set of testing results. First, we built a two-layer hierarchical EC tree to structure the 34 environments. In the first layer, the 34 environments were exactly divided into two groups, each corresponding to one gender. In the second layer, another two groups were classified roughly according to high/low SNR levels. We prepared a representative HMM set for each of these seven nodes. Each set of representative HMMs was trained in a multi-style training manner using the speech data

belonging to its corresponding node. Then, SME [26, 27] further improved these representative HMM sets. During testing, we located one cluster from this EC tree and used its corresponding representative HMM set to recognize the testing utterance. Here, the same AGI unit was used for the first layer of the EC tree to identify speaker’s gender. At the second layer, an online cluster selection was conducted to determine one most suitable cluster of speaking environments.

Table 5.6 WER (%) for ESSEM with weighted N -best information on Framework-2 (GD).

Test Condition	SetA	SetB	SetC	Overall
Baseline(AGI)	5.09	5.32	6.69	5.50
Baseline(EC)	5.05	5.31	6.31	5.41
ESSEM+EC($\eta \rightarrow \infty$)	4.58	4.88	5.51	4.89
ESSEM+EC($\eta=15$)	4.53	4.78	5.46	4.82
ESSEM+EC($\eta=0$)	4.53	4.89	5.54	4.88

By comparing “Baseline(AGI)” and “Baseline(EC)” in Table 5.6, we observe that the EC-structured baseline provides better performance than the AGI-only baseline. This result confirms that in addition to two genders, a speaking environment clustering process gives us a better baseline system to improve from. Moreover, by comparing “GD-Baseline” in Table 4.9 and “Baseline(EC)” in Table 5.6, we observe that after SME training, we can get better baseline results.

To test weighted N -best information, we conduct experiments of ESSEM with EC [16]. We adopted the same hierarchical EC tree as stated in the “Baseline(EC)” testing to prepare seven clusters. Environments belonging to a same cluster then formed an EC sub-space, $\Omega_{\mathbf{v}^{(c)}}$, $c=1,2,\dots,C$ (here $C=7$). In the online stage, ESSEM selected a cluster (for example, the t -th cluster) and located its corresponding sub-space ($\Omega_{\mathbf{v}^{(t)}}$). With the

selected sub-space, we estimate the target super-vector, V_Y , by:

$$V_Y = \mathbf{G}_\phi(\Omega_{V^{(t)}}), \quad (5.30)$$

with

$$\{\hat{w}_p\}_{p=1}^{P^{(t)}} = \arg \max_{\{w_p\}_{p=1}^{P^{(t)}}} \sum_{n \in N} \lambda_n P(F_Y | \sum_{p=1}^{P^{(t)}} w_p V_p, W_n). \quad (5.31)$$

We used an 8-best list ($N=8$). Table 5.6 lists results of ESSEM plus EC with setting $\eta \rightarrow \infty$ and $\eta=0$. By testing many different values, we found $\eta=15$ gave the best performance, and we listed the results in Table 5.6. In the following ESSEM experiments, we will integrate the weighted N -best information technique with $\eta=15$.

5.4.4 Cohort Selection and ESA on Framework-2 (GD)

Next, we tested the two online methods, cohort selection and ESA. For cohort selection, we located 15 environments closest to the testing condition in the original ESS space. For the ESA technique, we integrated it with EC (named EC-ESA). Similar to the original EC algorithm, a cluster was first selected based on the testing utterances. Then, ESA compensated the parameters of the selected EC sub-space, $\Omega_{V^{(t)}}$, and generated a new EC-ESA space, $\Omega_{V_{ESA}^{(t)}}$, through stochastic matching:

$$\Omega_{V_{ESA}^{(t)}} = \mathbf{G}_\theta(\Omega_{V^{(t)}}). \quad (5.32)$$

Here, we adopted a simple mapping process for $\mathbf{G}_\theta(\bullet)$. We compensated each super-vector to match the testing condition individually:

$$V'_p = \mathbf{G}_{\theta_p}(V_p), p=1 \dots P^{(t)}, \quad (5.33)$$

where V'_p and V_p , are the compensated and original super-vectors for the p -th

environment, and $\mathbf{G}_{\theta_p}(\bullet)$ is the mapping structure for the p -th super-vector. Finally, we obtain the EC-ESA space by:

$$\Omega_{\mathbf{V}_{ESA}^{(t)}} = \{ \mathbf{V}'_1 \mathbf{V}'_2 \dots \mathbf{V}'_{p^{(t)}} \}. \quad (5.34)$$

Here, we used diagonal MLLR [6] for the mapping function, $\mathbf{G}_{\theta_p}(\bullet)$, to compensate each super-vector in an unsupervised manner. With the EC-ESA space, $\Omega_{\mathbf{V}_{ESA}^{(t)}}$, ESSEM used the LC function as indicated in Eq-(3.13) for the mapping structure to estimate the target super-vector, \mathbf{V}_Y . Table 5.7 lists the results for cohort selection and EC-ESA as “ESSEM+cohort(LC)” and “ESSEM+EC-ESA(LC)”, respectively. For ease of comparison, we also lists the ESSEM results with EC as “ESSEM+EC(LC)” in Table 5.7; this set of results is the same to that of $\eta=15$ in Table 5.6.

Table 5.7. WER (%) for ESSEM plus EC, cohort selection, and ESA with linear combination on Framework-2 (GD).

	Set A	Set B	Set C	Overall
ESSEM+EC(LC)	4.53	4.78	5.46	4.82
ESSEM+cohort(LC)	4.53	4.71	5.49	4.79
ESSEM+EC-ESA(LC)	4.41	4.75	4.97	4.66

By comparing Table 5.6 and Table 5.7, we can see that the three ESSEM results are clearly better than the two baseline results. Next from Table 5.7, we observe that “ESSEM+cohort(LC)” can give slightly better performance than “ESSEM+EC(LC)”. It is noted that both EC and cohort selection resemble the subset selection methods [18-20]. EC online selects one sub-space from many prepared EC-structured sub-spaces, while cohort selection online collects super-vectors to construct a cohort ESS space. The testing results from Table 5.7 confirm that online cohort selection can provide relatively better resolution to model the testing condition in this particular task.

We also observe that “ESSEM+EC-ESA(LC)” achieves clearly better performance than “ESSEM+EC(LC)”. Especially for test Set C, where an additional channel diction is included, EC-ESA gives a clear improvement of 8.97% (5.46% to 4.97% WER) relative WER reduction over EC alone. This result suggests that ESA can online generate an ESS space that provides better coverage that can facilitate to characterize the testing conditions, especially for those containing distortions not included in the training set.

5.4.5 Complex Online Mapping Structure on Framework-2 (GD)

Next, we compare the same three techniques—EC, cohort selection, and EC-ESA—with a more complex mapping structure, linear combination with correction bias (LCB) as shown in Eq-(5.2). Again, we incorporated the weighted N -best information in this set of experiments. For EC with weighted N -best information, Eq-(5.3) now becomes:

$$\mathbf{V}_Y = \sum_{p=1}^{P^{(t)}} \hat{w}_p \mathbf{V}_p + \hat{b}, \quad (5.35)$$

with

$$\{\hat{w}_p, \hat{b}\}_{p=1}^{P^{(t)}} = \arg \max_{\{w_p, b\}_{p=1}^{P^{(t)}}} \sum_{n \in N} \lambda_n P(F_Y, w_n | \sum_{p=1}^{P^{(t)}} w_p \mathbf{V}_p + b, W_n). \quad (5.36)$$

Table 5.8 lists the EC result as “ESSEM+EC(LCB)”. We use the same step as stated in the previous section to implement cohort selection and EC-ESA; the corresponding results are listed as “ESSEM+cohort(LCB)” and “ESSEM+EC-ESA(LCB)”.

As shown in Table 5.8, the results for all three sets are now similar. However, EC-ESA still gives slightly better performance than the other two techniques and provides WER reductions of 15.64% and 14.23%, respectively, over “Baseline(AGI)” (5.50% to 4.64% WER) and “Baseline(EC)” (5.41% to 4.64% WER).

Table 5.8. WER (%) for ESSEM plus EC, cohort selection, and ESA with linear combination with bias on Framework-2 (GD).

	SetA	SetB	SetC	Overall
ESSEM+EC(LCB)	4.43	4.74	4.98	4.66
ESSEM+cohort(LCB)	4.48	4.74	4.92	4.67
ESSEM+EC-ESA(LCB)	4.41	4.73	4.92	4.64

5.4.6 ESSEM with Best Offline and Online Configurations on Framework-2 (GD)

In this section, we present our optimal offline and online configuration that gives the best performance on the Aurora-2 task. Based on our testing with different combinations of the proposed techniques, we found the following configuration enables ESSEM to achieve the best performance: 1) offline: an SME-based intraEnv training followed by MCE-based interEnv training plus EC with a two layer tree structure; 2) online: weighted N -best information technique with MCM algorithm using linear combination with a correction bias as both the mapping structure and MCM functions. The overall performance under each testing condition of the Aurora-2 test set is listed in Table 5.9. The average WER of this best configuration is 4.54% WER (95.46% Word Accuracy), which corresponds to a 16.08% WER reduction (from 5.41% to 4.54% WER) over our best baseline, “Baseline(EC)” reported in Table 5.6.

Table 5.9. ESSEM with the best offline and online configuration (in accuracy %).

	Set A					Set B					Set C			Overall
	Subway	Babble	Car	Exhibition	Ave.	Restaurant	Street	Airport	Station	Ave.	Subway M	Street M	Ave.	Ave.
20 dB	99.66	99.49	99.58	99.48	99.55	99.72	99.40	99.49	99.75	99.59	99.69	99.40	99.55	99.57
15 dB	99.48	99.24	99.43	99.17	99.33	99.51	99.09	99.55	99.51	99.42	99.54	99.12	99.33	99.36
10 dB	98.89	98.52	98.75	98.18	98.59	98.68	97.76	98.60	98.55	98.40	98.28	97.94	98.11	98.42
5 dB	96.41	94.98	96.87	94.63	95.72	95.49	94.62	95.62	95.87	95.40	96.01	94.29	95.15	95.48
0 dB	87.66	79.41	89.02	84.73	85.21	81.92	83.10	86.16	85.10	84.07	85.69	82.19	83.94	84.50
Ave.	96.42	94.33	96.73	95.24	95.68	95.06	94.79	95.88	95.76	95.37	95.84	94.59	95.22	95.46

5.4.7 EC, PCA, and HDR on Framework-1 (GI)

In the previous sections, we present the results of online techniques that improve the precision of online super-vector estimation. In the following discussion, we will

demonstrate algorithms that enhance the efficiency of the ESSEM online operation.

From Section 4.3.2, we verified that the EC algorithm can also reduce the dimensionality of the environment configuration and provide high resolution to characterize the testing condition. In the first set of experiments, we compare performances of ESSEM with the ESS spaces structured by EC and reformed by PCA and HDR. We illustrated the results in Figure 5.1. Each result in Figure 5.1 is an average WER of 50 testing results for the three testing sets (Set A, Set B, Set C) and five SNR levels (0dB to 20dB). We denote “Full” for ESSEM with entire ESS space ($P=34$) and “EC(1)” for the EC-structured ESS space with a one-layer (cluster number $C=3$) tree structure. The results of ESSEM using the HDR-built and PCA-built ESS spaces are presented as “HDR” and “PCA”, respectively in Figure 5.1. To have a fair comparison, we intentionally set the three ESS spaces having a same number of super-vector axes. In another word, since “EC(1)” has 17 super-vectors in each leaf node, we use $K=17$ (K is the number of super-vector axes) for both “HDR”, and “PCA”.

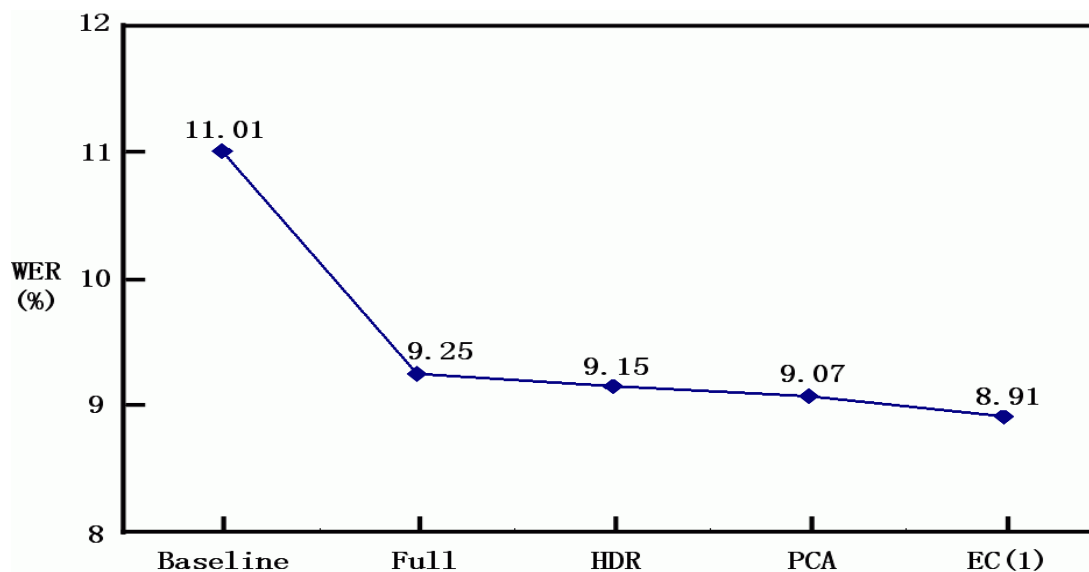


Figure 5.1. ESSEM with different environment spaces.

From Figure 5.1, we first observe that ESSEM with full set of ESS space can already achieve better performance than “Baseline”. Next, we observe that “EC(1)” provides better performance than using a full set of ESS spaces and gives a 19.07% (11.01% to 8.91%) WER reduction over “Baseline”. Moreover, we found EC(1) achieves even better performance than both “HDR” and “PCA”. From this set of results, we verified that when reducing the original ESS space into a same dimensionality, EC alone can achieve better performance than both PCA and HDR.

5.4.8 Dimensionality Reduction on EC on Framework-2 (GD)

From Section 4.3.2, we observe that the EC algorithm can enhance the resolution to model the testing conditions. Moreover, using a better EC tree structure (two-layered EC tree) can further reduce the number of super-vectors in each cluster than a single-layered EC tree. In this section, we use the EC with a two-layered hierarchical tree as the basis system and adopt dimensionality reduction techniques to further reduce complexity and enhance the efficiency of the online operation. As mentioned earlier, for a two-layered EC tree, the root and intermediate nodes have 34 and 17 super-vectors, while each leaf node has 12 to 14 super-vectors. In this set of experiments, we again adopted PCA and HDR as two representative dimensionality reduction techniques.

More recently, a study indicates that a hierarchical PCA (H-PCA) algorithm can provide better performance than direct PCA [104] for particular applications. In this study, we also implemented the H-PCA in the ESSEM framework and compare its performance with direct PCA. When applying H-PCA on the EC-structured ESS space with two-layered EC tree, the first stage PCA is first performed on each of the four leaf nodes. Then, the top ten principal components (PCs) are collected from each leaf node.

Each intermediate node (each corresponding to one of the two genders) now has 20 PCs collected from the two leaf nodes. Then, we perform the second stage PCA on the 20 PCs to obtain the final PCs for each intermediate node gender.

Figure 5.2 demonstrates ESSEM performances with applying PCA, H-PCA, and HDR on the EC-structured ESS space. Each result in Figure 5.2 is an average digit word accuracy (in %) of 50 testing results for the three testing sets (Set A, Set B, Set C) and five SNRs (0dB to 20dB). In Figure 5.2, we also list the results of cohort selection introduced in Section 5.1.4. To reduce the dimensionality reduction with a fixed ESS space, we believe cohort selection can serve as the upper bound algorithm. K in Figure 5.2 equals to the number of super-vector used to characterize the target super-vector. As mentioned earlier, the original system uses 12 to 14, and we reduce the complexity to $K=10$ (complexity is roughly reduced to 2/3) and $K=5$ (complexity is roughly reduced to 1/3). In the experiments, we used MCE-based intraEnv training followed by MCE-based interEnv training (as presented in Section 4.3.6) and adopted the linear combination function in Eq-(3.13) as the mapping function.

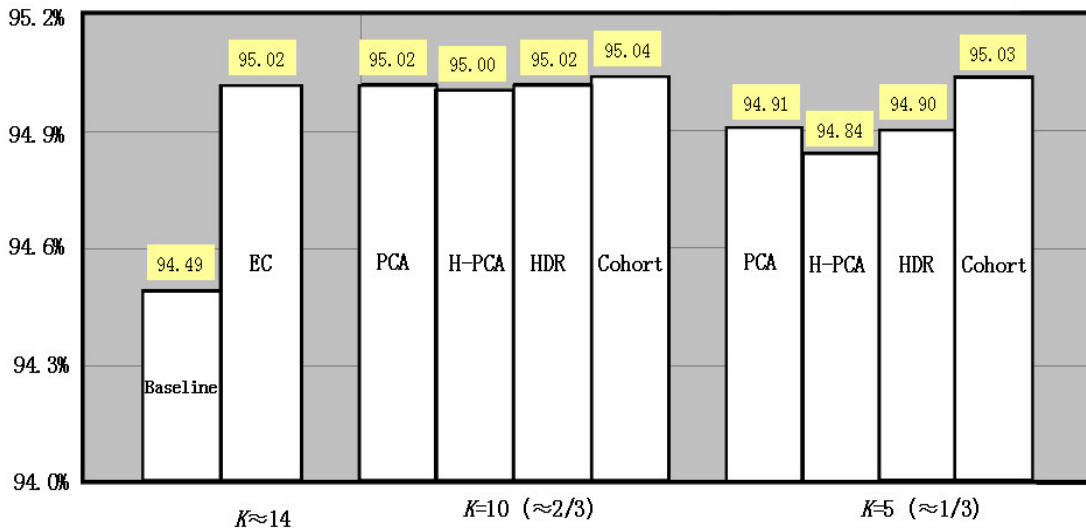


Figure 5.2. PCA, HDR, H-PCA, cohort selection, on the EC-structured ESS space.

From Figure 5.2, we can see that when reducing complexity roughly to 2/3, the performances for the four algorithms were not degraded much. Among them, cohort selection provides the best performance (95.04% word accuracy). When reducing complexity roughly to 1/3, the performance achieved by cohort selection maintains at a similar level, while performances for the other three algorithms (PCA, H-PCA, HDR) are degraded; specially, H-PCA gives the worst performance. Nevertheless, when compared with the baseline result, H-PCA still provides a 6.35% relative WER reduction (5.51% WER to 5.16%WER). Since PCA gives the best performance among the three techniques, we only present a further study of PCA on ESSEM in the following section.

5.4.9 Integration of EC and PCA (EC-PCA) on Framework-2 (GD)

In this section, we report the ESSEM performance with an ESS space built by using an integration of EC and PCA (denoted as EC-PCA). In the offline phase, we prepared a same ESS space as that was used in Section 5.4.3 to Section 5.4.6; after that, we applied PCA on each EC sub-space. In the online phase, we first determined a proper dimensionality for each sub-space; then, we used the weighted N -best information technique and adopted the linear combination function with a correction bias as the mapping function to estimate the target super-vector, V_Y .

Figure 5.3 illustrates the EC-PCA results with different number of eigenvectors used in each EC sub-space. Each result in Figure 5.3 is an average WER of 50 results for the three testing sets (Set A, Set B, Set C) and five SNRs (0dB to 20dB). When setting the dimensionality to zero, no prior knowledge is used, and ESSEM becomes a single bias vector stochastic matching [12, 13]. In Figure 5.3, “Full” denotes the original EC technique that uses all the super-vectors without performing PCA complexity reduction.

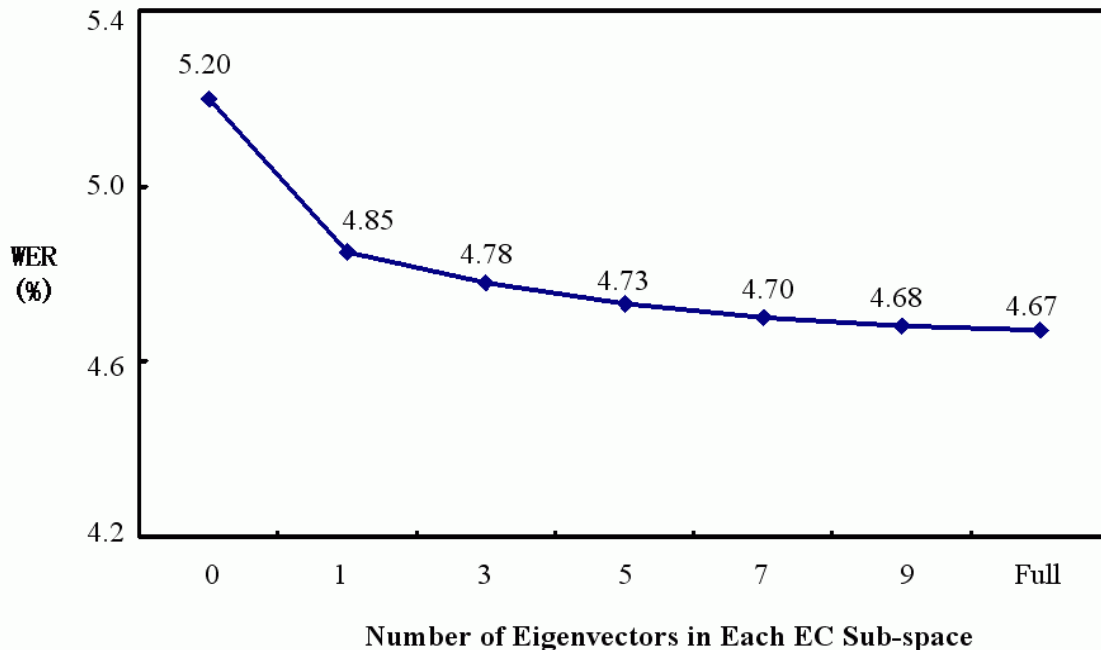


Figure 5.3. Performance of EC-PCA with different dimensions in EC subspaces.

From the results, we observed that WER decreases consistently when more prior knowledge is incorporated. The performance of EC-PCA almost converges to EC when the dimensionality of each EC-PCA sub-space is larger than five. Please note that when setting the dimensionality to five, the complexity of an EC node has been reduced to more than 2.4 fold with respect to the original EC configuration. In the real word applications, since we prepare the EC-PCA structure in the offline phase, we only need to online determine a proper dimension for each EC-PCA sub-space based on the amount of available adaptation data and computational constraints.

5.4.10 Generalized ESSEM on Framework-1 (GD)

In this section, we present the GESSEM performance in a supervised adaptation manner on the Aurora-2 task. We conducted experiments on the Framework-1, GD system. Each of the 70 testing conditions in Aurora-2 has 1001 speech utterances recorded from 104

speakers, 52 male and 52 female speakers. Each speaker pronounced nine or ten utterances. We used the first two utterances as the adaptation data and the rest seven or eight utterances for testing. Accordingly, we collected 208 adaptation utterances and 793 testing utterances for each testing condition. For the ESSEM framework, we used an EC-structured ESS space with a two-layer hierarchical tree structure as presented in Section 4.3.3. The two adaptation utterances were first used to identify the speaker's gender by an AGI unit and then used to estimate the super-vector for the testing condition. We used the AGI process followed by one-layer speaking environment cluster selection as presented in Eq-(4.17) to locate one set of HMMs. Then, we tested recognition with the located HMM set to obtain the baseline results. The linear combination function as shown in Eq-(3.13) was used as the online mapping structure.

Table 5.10 lists the GESSEM results of using both two-stage and combination methods introduced in Section 5.3. The baseline results are also provided in the same table. For the combination method, we tested performance with different weighting coefficients, α , from 0.0 to 1.0. When $\alpha=1.0$, we actually reached to the original ESSEM model adaptation; when $\alpha=0.0$, we obtained a likelihood-ratio-based ESSEM adaptation.

By investigating the results in Table 5.10, we first observe that all the methods achieve clear performance improvements over “Baseline”. Next, it is clear that either the combination method ($\alpha=0.25, 0.5, \text{ or } 0.75$) or the original ESSEM ($\alpha=1.0$) achieve better performance than the two-stage method. Finally, we find that the combination method provides better performance than either the likelihood-ratio-based ESSEM ($\alpha=0.0$) or the original ESSEM ($\alpha=1.0$). The results verify our assumption that if we incorporate more knowledge sources into the objective function, the acoustic models can be better

calculated and accordingly achieve better recognition performance.

Table 5.10. WER (%) for GESSEM with two-stage and combination methods.

Two-Stage	0dB	5dB	10dB	15dB	20dB	Average
	27.17	8.09	2.95	1.67	1.16	8.21
Combined	0dB	5dB	10dB	15dB	20dB	Average
$\alpha=0.00$	27.61	8.15	3.08	2.11	1.55	8.50
$\alpha=0.25$	26.66	8.01	2.90	1.67	1.15	8.08
$\alpha=0.50$	26.64	8.01	2.90	1.65	1.16	8.07
$\alpha=0.75$	26.62	8.00	2.89	1.66	1.16	8.07
$\alpha=1.00$	26.98	8.06	2.94	1.66	1.15	8.16
Baseline	32.96	8.58	3.09	1.95	1.51	9.62

In our current GSM and GESSEM studies, we consider the conventional likelihood ratio as the separation measure and as an alternative knowledge source. In [27], a more discriminative separation measure is defined by including a frame selection mechanism. Such a mechanism only counts the frames with different labels in the target and competing strings. Then, the original LLR in Eq-(5.22) is modified to a normalized LLR:

$$D(F_Y, \Omega_V, \varphi) = \frac{1}{N^{Dis}} \left(\sum_j \log [P(F_{Y,j} | \Omega_V, \varphi, W_c)] - \log \left[\frac{1}{N} \sum_{n=1}^N P(F_{Y,j} | \Omega_V, \varphi, W_n) \right] I(F_{Y,j} \in O^{Dis}) \right), \quad (5.37)$$

where O^{Dis} denotes the frames with different labels in the target and competing strings, and N^{Dis} is the total number of such discriminative frames in the adaptation utterance, F_Y .

Another possible direction to further improve the GESSEM approach is to address the problem that each hypothesis of the N -best list should provide different level of discriminative information. In other words, each list has a different weight, and Eq-(5.27) can be rewritten as:

$$D(F_Y, \Omega_V, \varphi) = \log [P(F_Y | \Omega_V, \varphi, W_c)] - \log \left[\sum_{n=1}^N \lambda_n P(F_Y | \Omega_V, \varphi, W_n) \right]. \quad (5.38)$$

where W_n and λ_n are the decoded transcription and weight for the n -th hypothesis.

5.5 Summary

In this chapter, we study methods to enhance the precision and efficiency of the online process of the ESSEM approach. To enhance precision, we investigate five directions. First, we study different mapping functions and compare their performances. In Section 5.4.1, the experimental results indicate that by using a more properly specified mapping structure, testing environments can be better characterized. Second, based on the ensemble estimator algorithm, we propose a multiple cluster matching (MCM) algorithm to further improve the online modeling precision. Experimental results in Section 5.4.2 suggest that the MCM algorithm enables ESSEM to not only achieve a significant performance improvement over the baseline but also produce a further improvement over the ESSEM without the MCM algorithm. Third, we incorporate the weighted N -best information to improve ESSEM framework in an unsupervised compensation mode. From testing results in Section 5.4.3, we verify that ESSEM indeed achieves a better performance with weighted N -best information than either a 1-best system or an equal weighted system. Finally, we introduce cohort selection and EC-ESA to online refine the ESS spaces. Cohort selection and EC-ESA, respectively, enable ESSEM to have better resolution and coverage to characterize unknown testing conditions. Results in Section 5.4.4 verify these two online methods do improve the overall ESSEM performance. With the offline algorithms presented in Chapter 4 and the online algorithms introduced in this chapter, we obtained our best ESSEM performance on the Aurora-2 task. The average WER of our best result is 4.54% over 50 testing conditions (0dB-20dB, 10 different noise types); the complete testing results are listed in Table 5.9. We also incorporated two types of dimensionality reduction techniques, PCA and HDR to reduce the complexity of the

ESS space. In Section 5.4.9, experimental results verify that an integration of PCA with EC can significantly enhance the efficiency while maintaining a satisfactory performance. Next, we propose a GESSEM approach by extending the original ESSEM framework. We evaluated the GESSEM framework in a supervised adaptation mode. From the experimental results in Section 5.4.10, we verified that GESSEM can further enhance original ESSEM by incorporating the log-likelihood ratio into the objective function. In the future, we can further study methods to enable GSM and GESSEM working in an unsupervised adaptation mode.

CHAPTER 6

CONCLUSION

We present an ESSEM framework that can be applied to enhance performance robustness of ASR under noisy conditions. We also propose techniques to refine the ESS spaces and enhance the online estimation for ESSEM. First, we introduce the main concept and implementation steps of ESSEM in Chapter 3. In Chapter 4, we present offline algorithms to refine the ESS space. We first introduce EC and EP to structure the ESS space well; then, we propose *intraEnv* and *interEnv* training to improve the discriminative power. For EC, although it requires an online cluster selection process before performing stochastic matching, the dimensionality of the selected sub-space is smaller than the original space; the computational cost is therefore lower than the original method. Moreover, the selected sub-space can provide high resolution to model the target super-vector for the testing environment than the entire ESS space. For EP, the parameters belonging to different groups are estimated separately, and therefore the overall estimation of super-vector can be obtained accurately. Although we need to conduct several stochastic matching procedures instead of once, partitioning high-dimensional super-vectors is favorable in applications with limited resources of online operation. Next, we use *intraEnv* and *interEnv* training to enhance confidence interval within one particular environment and increase the distance across different environments, respectively. These offline algorithms were evaluated in an unsupervised compensation (self-learning) mode with very limited adaptation data. Recognition results first indicate that ESSEM achieves better performance by using EC and EP. Next, we verify that the ESSEM performance can be enhanced with an ESS space refined by

MCE-based intraEnv and interEnv training than an ESS space trained by ML. Moreover, we verified that by adopting SME for intraEnv training, the ESSEM performance can be further improved. Based on two measurements, we confirm that intraEnv training enhances the separation between parameters within an HMM set for a particular environment and interEnv training increases the difference across environments. Finally, we integrate all the offline techniques, including intraEnv and interEnv training along with EC and EP, to prepare our best environment configuration in the offline phase.

In Chapter 5, we study methods to enhance the precision and efficiency of the online estimation process of the ESSEM approach. We first presented different mapping functions and compared their performances. Next, we propose a multiple cluster matching (MCM) algorithm to further enhance precision of the online modeling process. Then, we apply weighted N -best information on ESSEM to cope with the problem that the decoded transcription of unsupervised adaption may not be correct to guide the adaptation. Moreover, we introduce cohort selection and EC-ESA to refine the ESS spaces that provide better resolution and coverage, respectively, to aid the environment characterization. Finally, PCA and HDR are used to reduce the complexity of the ESSEM framework and improve its efficiency of online operation. From our experimental results, we first observe that by using a more properly specified mapping function, testing environments can be better characterized. Moreover, we found that MCM algorithm enables ESSEM to not only achieve a significant performance improvement over the baseline result but also produce a further improvement over the ESSEM without the MCM algorithm. We also conducted experiments and verified that after integrated with weighted N -best information, ESSEM can give better performance over an equal-

weighting or a 1-best system. To test the cohort selection and ESA techniques, we compare their performance with the original EC technique. From the experimental results, we observe that both cohort selection and EC-ESA indeed outperform the original EC technique. In Section 5.4.6, we report our current best offline and online ESSEM configuration, which achieves the best performance on the Aurora-2 task. Finally based on the testing results, we see that when using PCA and HDR to reduce complexity of the environment configuration, the efficiency of online operation can be significantly enhanced while maintaining a satisfactory performance. In the second part of Section 5, we extend the original stochastic matching criterion by adopting other acoustic knowledge into the objective function to increase the precision of the nuisance parameter estimation. We call this extension the generalized stochastic matching approach (GSM). Moreover, we apply GSM on the original ESSEM and propose the generalized ensemble speaker and speaking environment modeling (GESSEM) approach. In this thesis, we only incorporate the log-likelihood ratio. When considering the log-likelihood ratio in the objective function, GESSEM attempts to match each pattern to the testing utterances and also increase the separation across different patterns. The GESSEM approach was evaluated in a supervised adaptation mode. Experimental results verified that GESSEM indeed achieves better performance than the original ESSEM. In the future, other acoustic information will be used to enhance the confidence of stochastic matching process.

Two directions can be further investigated in the future. First, we implemented the ESSEM framework with an ESS space formed by 34 different environments in the offline phase. We believe the same approach can be extended to more environments for different ASR tasks. Second, as stated in Section 5.1.5, ESA online adapts the parameters in the

original ESS space. Instead of ESA, another straightforward method is to online sample a new ESS space. By online sampling a new ESS space, we can have a better coverage for the testing conditions. Another advantage of this sampling method is that we can configure the complexity of the ESS space to fit the computation constraints.

From the experimental results, we summarize three contributions for this thesis:

1) Instead of using a multi-style training style, ESSEM uses a single style training criterion to prepare environment-specific models. Multi-style training (also known as multicondition training) trains acoustic models by using speech data collectively from many different training conditions; single-style training obtains acoustic models for a particular environment by using training data from that acoustic condition. Therefore, multi-style trained acoustic models cover many different acoustic conditions, and single-style trained acoustic models focus on one single condition. In previous studies, multi-style trained speaker independent (SI) or environment independent (EI) acoustic models are usually used for ASR systems and show better performance robustness than that are trained on single-style clean condition training. For the proposed ESSEM framework, we prepare the ESS space by using multiple sets of acoustic models trained on single-style training. Each set of acoustic models characterizes a particular speaker and speaking environment condition. Based on the available testing data, a mapping structure is calculated to flexibly model the unknown testing condition. When comparing to multi-style trained models, ESSEM maps models more focused based on speech data from a specific environment. Moreover, the ESS space provides a prior knowledge to facilitate online model characterization. With this prior knowledge, ESSEM can efficiently characterize unknown testing conditions with simple mapping functions.

2) We can artificially simulate the prior environment structure and overcome the difficulty of data collection. An additional advantage of simulating the ESS space is—if the acoustic information for the testing condition is available in advance, we can construct a compact environment structure providing high resolution to model the testing environment. When likely noise types and the range of SNR are specified for a particular ASR application, a proper environment structure can be designed to well cover to the possible testing conditions for that application. On the other hand, when dealing with channel distortions, we can prepare an environment structure incorporating various channel distortions. Additive noise distortions do not need to be incorporated when constructing the environment structure for that particular application.

3) Although the original ESSEM framework already provides significant improvement over the baseline, we propose offline and online techniques to further enhance the ESSEM performance. With the offline techniques, we can prepare a well-structured ESS space with wide coverage of different environments and good discriminative power. With the online methods, we can further enhance the precision and efficiency of the online super-vector estimation process. Experimental results further confirm that with an optimal offline and online configuration, ESSEM can achieve the best performance on the Aurora-2 task.

With the advance of storage and computation power, the ESSEM framework can be easily implemented in real-world ASR systems. One example can be an online client service system. During the offline phase, the system prepares acoustic models for the registered users with many different service conditions. Each set of acoustic models can be stored as a profile. The system saves and well structures these profiles with proposed

offline techniques. During the online phase, for a registered user under prepared service condition, the corresponding profile can be directly retrieved and used for that service; for an unregistered user or a registered user under an unknown service condition, we can locate one profile that best matches to the testing condition or search for a cluster of relevant profiles to online generate the target profile. Moreover, additional inputs that provide acoustic information for the testing environment can facilitate the searching process and reduces time and computational costs in the online operation. When a new user subscribes, or when a new service condition is required, the system will update the necessary profiles and reorganized the structure.

APPENDIX A

ASSOCIATION OF ESSEM WITH EXEMPLAR THEORY

The basic concept of ESSEM resembles that of exemplar theory [105-109]. The exemplar theory is originated from psychology studies and recently adopted in speech perception and processing research. In this section, we introduce the background and main concept of the exemplar theory and also describe its similarity to the ESSEM framework.

In the psychological literature, a prototype theory has a long tradition on categorization tasks [110]. The prototype theory assumes that each category is characterized by an abstract prototype. Although one category may present various outputs due to environment changes, its corresponding abstract prototype is explicitly computed and stored in advance. Moreover, only the abstract representation of each category is considered when processing categorization. More recently, an exemplar theory was proposed. Different from the traditional prototype theory, the exemplar theory assumes that each category is defined by a cloud of specific and episodic memories. Studies related to the exemplar theory have verified that people can retain detailed memories of an event for a long time; all the memories are taken into account when carrying out categorization tasks [105-109].

More recently, the two theories have been extended to the domain of speech perception and production. Figure A.1 diagrams a clear comparison of these two theories representing phonetic categories in a parametric space. For the prototype theory, each phonetic category is defined by an abstract representation. For exemplar theory, on the other hand, each phonetic category is characterized by a ‘cloud’ of remembered

exemplars. Exemplars are organized within the category by similarity across salient features and produce internal category structure. Depending on the category model, new experiences are determined and assigned to relevant a category by comparing to existing exemplars. After performing categorization, the new experience is inserted into the relevant category or raises the activation of an exemplar that was previously stored. Therefore, each experience affects the entire category system by changing the range and/or updating activation of existing exemplars.

The application of exemplar theory on the speech perception and production system successfully explains the apparent sensitivity of the listeners to social information such as age and social class and to gender stereotypicality [108]. Some surprising findings in the literature also support speech perception to be based on an exemplar-based memory system [105-107]. Moreover, exemplar models offer an adaptation mechanism handling various distortions that influence listeners including dialect variation and acoustic environments such as reverberation [109].

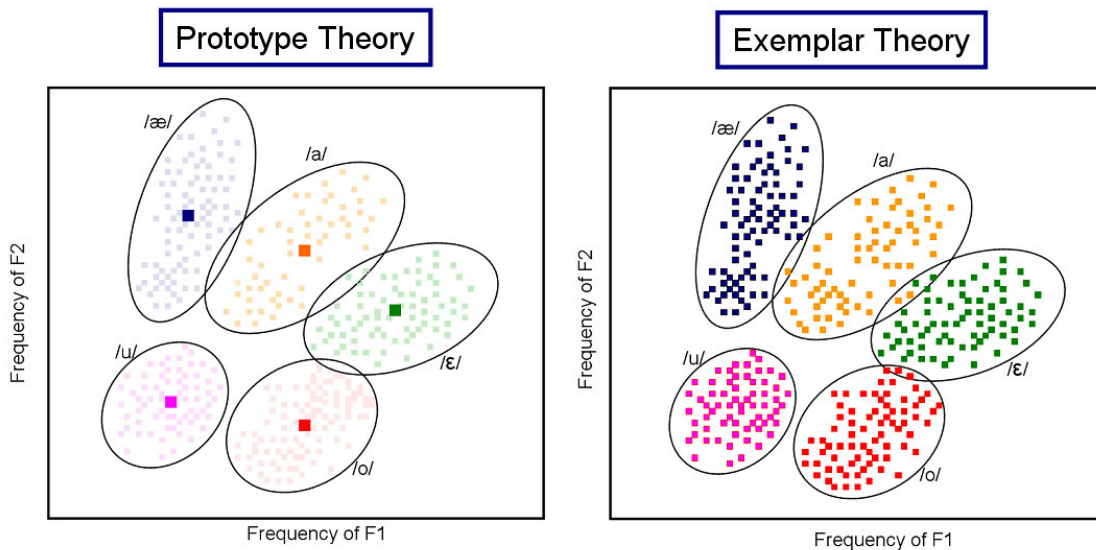


Figure A.1. Prototype theory and exemplar theory.

For the proposed ESSEM approach, we also intend to use a cloud of super-vectors, instead of a single super-vector, to characterize the unknown testing condition. The super-vector cloud provides the prior knowledge and resembles the previous memories of the exemplar theory in the speech perception and prediction system. During the online processing of ESSEM, if we can locate a super-vector that perfectly matches to the testing condition, we directly use that super-vector to perform speech recognition for that particular testing condition. This process is very similar to recruit the experienced memory to aid speech perception for the exemplar theory. When the exactly matched super-vector can not be located, ESSEM uses more complex mapping structures, such as linear combination, to online generate a super-vector that matches to the testing condition. This process is similar to collecting a group of relevant experienced memories to facilitate speech perception when encountering an unknown testing condition.

APPENDIX B

DATABASES USED IN THIS THESIS

B.1 TIMIT

TIMIT contains broadband recordings of 630 speakers from eight major dialect regions of the United States. Each speaker pronounced ten sentences, and each of these utterances was processed into 16-bit, 16kHz waveform file. The TIMIT corpus also includes time-aligned orthographic, phonetic and word transcriptions corresponding to each utterance. The corpus design was a joint effort among the Massachusetts Institute of Technology (MIT), SRI International (SRI) and Texas Instruments Incorporated (TI). The sentences are divided into three categories—SA, SX and SI sentences. The SA sentences (the dialect sentences) were intentionally designed to expose the dialectal variants of speakers and were spoken by all the 630 speakers. The SX sentences (phonetically-compact sentences) were designed to provide a good coverage of pairs of phones, with extra occurrences of phonetic contexts thought to be either difficult or of particular interest. Each speaker read five SX sentences and each text was spoken by seven different speakers. The SI sentences (phonetically-diverse sentences) were selected to add diversity in sentence types and phonetic contexts. Each speaker read three SI sentences, with each sentence being read only by a single speaker.

TIMIT contains two testing sets, core and complete test sets. The core test set includes 24 speakers, two male and one female from each dialect region. Each speaker pronounced eight different sentences (five SX and three SI sentences). Thus, the core test set contains a total of 192 sentences. The complete test set was obtained by including the sentences from all speakers that read any of the SX texts included in the core test set.

Therefore, no sentence text appears in both the training and test sets. This complete test set contains a total of 168 speakers and 1344 utterances.

B.2 NOISEX-92

The NOISEX-92 corpus includes following 15 different noise sources:

1. Voice Babble: recorded from 100 people speaking in a canteen. The room radius is over two meters; therefore, individual voices are slightly audible. The sound level during the recording process was 88 dBA.
2. Buccaneer1: acquired by the Buccaneer jet moving at a speed of 190 knots, and an altitude of 1000 feet, with airbrakes out. The sound level during the recording process was 109 dBA.
3. Buccaneer2: acquired by Buccaneer moving at a speed of 450 knots, and an altitude of 300 feet. The sound level during the recording process was 116 dBA.
4. Destroyer noises (engine room): recorded in an engine room. The sound level during the recording process was 101 dBA.
5. Operations room: recorded in an operation room. The sound level during the recording process was 70 dBA.
6. F16: recorded at the co-pilot's seat in a two-seat F-16, traveling at a speed of 500 knots, and an altitude of 300-600 feet. The sound level during the recording process was 103 dBA.
7. Factory1: recorded near plate-cutting and electrical welding equipment.
8. Factory2: recorded in a car production hall.
9. HF radio channel noise: Recording of noise in an HF radio channel after demodulation.

10. Leopard: recorded by a Leopard 1 vehicle moving at a speed of 70 km/h. The sound level during the recording process was 114 dBA.
11. M109: acquired by an M109 tank moving at a speed of 30 km/h. The sound level during the recording process was 100 dBA.
12. Machine gun: recorded by a 0.50 calibre gun fired repeatedly.
13. Pink Noise: acquired by sampling high-quality analog noise generator (Wandel & Goltermann). Exhibits equal energy per 1/3 octave.
14. White Noise: acquired by sampling high-quality analog noise generator (Wandel & Goltermann). Exhibits equal energy per Hz. bandwidth.
15. Volvo: acquired at Volvo 340 driving at 120 km/h, in 4th gear, on an asphalt road, in rainy conditions.

B.3 Aurora-2

Speech data in the Aurora-2 corpus are based on the TIDigits [111] speech data downsampled at 8 kHz and filtered through G.712 characteristic [112] to simulate the global system for mobile communications (GSM) channel effect. Eight different noise sources (suburban train, babble, car, exhibition hall, restaurant, street, airport, and train station) were artificially added in a controlled fashion to cover a range of SNR levels. The range includes a no noise condition, referred to as the clean condition, along with six SNR levels—20, 15, 10, 5, 0, and -5 dB.

The Aurora-2 corpus contains two training sets and three test sets. The two training sets are referred to as clean and multicondition training sets. The clean training set does not contain any additive noise. The multicondition training set is representative of four noise types (suburban train, babble, car, and exhibition hall) covering five SNR

ratios—20, 15, 10, 5 dB and the clean condition. Both clean training and multicondition training sets consist of 8440 utterances selected from the training part of the TIDigits containing the recordings of 55 male and 55 female adults.

Three test sets are defined as Test Sets A, B and C. Test Set A is representative of all four noise types seen in the multicondition training set. Test Set B is representative of four noise types not represented in the multicondition training set. Test Set C is filtered through M-IRS filtering [113] to introduce convolutional noise. It contains suburban street and train noises. Each Test set covers all SNR levels—20, 15, 10, 5, 0, -5 dB and the clean condition. Therefore, Aurora-2 includes a total of 70 different testing conditions (ten different noises with seven SNR levels). Each testing condition includes 1001 utterances selected from the TIDigits test set.

REFERENCES

- [1] L. R. Rabiner, "A tutorial on hidden Markov models and selected applications in speech recognition," *Proc. IEEE*, vol. 77, pp.257-286, Feb. 1989.
- [2] B.-H. Juang, and L. R. Rabiner, "The segmental K-means algorithm for estimating parameters of hidden Markov models," *IEEE Trans. Speech Audio Processing*, vol. 38, pp. 1639-1641, Sept. 1990.
- [3] B.-H. Juang, W. Chou, C.-H. Lee, "Statistical and discriminative methods for speech recognition," In: C.-H. Lee, K.K. Soong, F.K. Paliwal, *Automatic Speech and Speaker Recognition: Advanced Topics*, Ch. 5. Kluwer Academic Publishers, Dordrecht, 1996.
- [4] J.-L. Gauvain and L. Lamel, "Large-vocabulary continuous speech recognition: advances and applications," *IEEE Trans. on Speech and Audio Processing*, vol. 88, No. 8, pp. 1181-1200, Aug. 2000.
- [5] J.-L. Gauvain and C.-H. Lee, "Maximum a posteriori estimation for multivariate Gaussian mixture observations of Markov chains," *IEEE Trans. Speech Audio Processing*, vol. 2, no. 2 , pp.291-99, Apr. 1994.
- [6] C. Leggetter and P. Woodland, "Maximum likelihood linear regression for speaker adaptation of continuous density hidden Markov models," *Comput. Speech and Lang.*, vol. 9, pp.171-185, 1995.
- [7] T. J. Hazen, "A comparison of novel techniques for rapid speaker adaptation," *Speech Comm.*, pp.15-33, 2000.
- [8] M. J. F. Gales, "Cluster adaptive training of hidden Markov models," *IEEE Trans. Speech Audio Processing*, pp. 417-428, 2000.

- [9] R. Kuhn, J.-C. Junqua, P. Nguyen, and N. Niedzielski, "Rapid speaker adaptation in Eigenvoice space," *IEEE Trans. Speech Audio Processing*, vol. 8, pp.695-707, Nov. 2000.
- [10] M. Akbacak and J. H. L. Hansen, "Environmental sniffing: noise knowledge estimation for robust speech systems," *IEEE Trans. Lang. Speech Audio Processing*, pp.465-477, 2007.
- [11] Z. Zhang and S. Furui, "Piecewise-linear transformation-based HMM adaptation for noisy speech," *Speech Comm.*, 2004.
- [12] A. Sankar and C.-H. Lee, "A maximum-likelihood approach to stochastic matching for robust speech recognition," *IEEE Trans. Speech Audio Processing*, vol. 4, pp.190-202, May.1996.
- [13] A. C. Suredran, C.-H. Lee, and M. Rahim, "Nonlinear compensation for stochastic matching," *IEEE Trans. Speech Audio Processing*, vol. 7, pp.643-655, Nov.1999.
- [14] Y. Tsao and C.-H. Lee, "A vector space approach to environment modeling for robust speech recognition," *Proc. ICSLP 2006*, pp.785-788, Sept. 2006.
- [15] Y. Tsao and C.-H. Lee, "An ensemble modeling approach to joint characterization of speaker and speaking environments," *Proc. Interspeech 2007*, pp. 1050-1053, Aug. 2007.
- [16] Y. Tsao and C.-H. Lee, "Two extensions to ensemble speaker and speaking environment modeling for robust automatic speech recognition," *ASRU 2007*, pp. 77-80, Dec. 2007.
- [17] Y. Tsao and C.-H. Lee, "Improving the ensemble speaker and speaking environment modeling approach by enhancing the precision of the online estimation process," *Proc. Interspeech 2008*, pp. 1265-1268, 2008.

- [18] S. Chen, D. Donoho, M. A. Saunders, "Atomic decomposition by basis pursuit," *SIAM J. on Scientific Computing*, vol. 20, No. 1, pp. 33-61, 1998.
- [19] R. Coifman and M. Wickerhauser, "Entropy-based algorithms for best basis selection," *IEEE Trans. on Info. Theory*, vol. 38, pp.713-718, March 1992.
- [20] S. Mallat and Z. Zhang, "Matching pursuit with time-frequency dictionaries," *IEEE Trans. on Signal Processing*, vol. 41, pp. 3397-3415, Dec. 1993.
- [21] Y. Tsao, S.-M. Lee, F.-C. Chou, and L.-S. Lee, "Segmental eigenvoice for rapid speaker adaptation," *Proc. Eurospeech 2001*, pp. 1269-1272, 2001.
- [22] Y. Tsao, S.-M. Lee and L.-S. Lee, "Segmental eigenvoice with delicate eigenspace for improved speaker adaptation," *IEEE Trans. Speech Audio Processing*, vol.13, No.3, pp.399-411, May 2005.
- [23] Y. Tsao and C.-H. Lee, "An ensemble speaker and speaking environment modeling approach to robust speech recognition," accepted, to be appeared in *IEEE Trans. on Audio, Speech, and Language Processing*.
- [24] G. Meinardus, G. Nurnberger, M. Sommer, and H. Strauss, "Algorithms for piecewise polynomials and splines with free knots" *Mathematics of Computation*, vol. 53, no. 187, pp. 235-247, Jul. 1989.
- [25] B.-H. Juang, W. Chou, and C.-H. Lee, "Minimum classification error rate methods for speech recognition," *IEEE Trans. Speech Audio Processing*, vol. 5, no. 3, pp. 257-265, May, 1997.
- [26] J. Li, M. Yuan, and C.-H. Lee, "Approximate test risk bound minimization through soft margin estimation," *IEEE Trans. on Audio, Speech, and Language Processing*, vol. 15, no. 8, pp. 2393-2404, 2007.

- [27] J. Li, "Soft margin estimation for automatic speech recognition," Ph.D. Dissertation, School of ECE, Georgia Institute of Technology, 2008.
- [28] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification*, New York, Wiley, 2001.
- [29] Y. Tsao, J. Li, and C.-H. Lee "Ensemble speaker and speaking environment modeling approach with advances online estimation process," submitted to *Proc. ICASSP 2009*.
- [30] R. P. Lippmann, E. A. Martin, and D. B. Paul, "Multi-style training for robust isolated-word speech recognition," *Proc. ICASSP 1987*, Dallas, TX, pp. 705-708, Apr. 1987.
- [31] S. Furui, "Unsupervised speaker adaptation method based on hierarchical spectral clustering," *Proc. ICASSP 1989*, pp. 286-289, 1989.
- [32] T. Kosaka and S. Sagayama, "Tree-structured speaker clustering for fast speaker adaptation," *Proc. ICASSP 1994*, pp. 245-248, 1994
- [33] X. Huang, A. Acero, H. Hon, *Spoken Language Processing: A Guide to Theory, Algorithm and System Development*, Prentice Hall, 2001.
- [34] L. R. Rabiner and B. H. Juang, *Fundamentals of Speech Recognition*, Prentice-Hall, 1993.
- [35] F. Jelinek, *Statistical Methods for Speech Recognition*. Cambridge, MA: MIT Press, 1997.
- [36] S. Davis and P. Mermelstein, "Comparison of parametric representations for monosyllable word recognition in continuously spoken sentences," *Proc. ICASSP 1980*, vol. 28, no.4, pp. 357-366, 1980.

- [37] H. Hermansky. "Perceptual linear predictive (PLP) analysis of speech," *Journal Acoustical Society of America*, vol. 87, no. 4, pp. 1738-1752, 1990.
- [38] L. A. Liporace, "Maximum likelihood estimation for multivariate observations of markov sources," *IEEE Trans. Information Theory*, vol.28, no.5, pp. 729-734, 1982.
- [39] X. D. Huang and M. A. Jack. "Semi-continuous hidden Markov models for speech signals," *Computer Speech and Language*, vol. 3, no. 3, pp. 329-252, 1989.
- [40] R. G. Leonard, "A database for speaker-independent digit recognition," *Proc. ICASSP 1984*, pp.328-331, 1984.
- [41] H. Ney. "The use of a one-stage dynamic programming algorithm for connected word recognition," *Proc. ICASSP 1984*, vol. 32, no. 2, pp. 263-271, 1984.
- [42] D. B. Paul. "Algorithms for an optimal A^* search and linearizing the search in the stack decoder," *Proc. ICASSP 1991*, vol. 1, pp. 693-696, 1991.
- [43] S. Ortmanns, H. Ney, and A. Eiden, "Language-model look-ahead for large vocabulary speech recognition," *Proc. ICSLP 1996*, pp. 2095-2098, 1996.
- [44] E. Bocchieri, "Vector quantization for efficient computation of continuous density likelihoods," *Proc ICASSP 1993*, vol. II, pp. 692-695, 1993.
- [45] H. Ney, D. Mergel, A. Noll, and A. Paeseler, "A data-driven organization of the dynamic programming beam search for continuous speech recognition," *Proc. ICASSP 1987*, pp. 833-836, 1987.
- [46] H. Ney, R. Haeb-Umbach, B.-H. Tran, and M. Oerder, "Improvements in beam search for 10000-word continuous speech recognition," *Proc. ICASSP 1992*, vol. 1, pp. 9-12, 1992.
- [47] R. A. Sukkar and C.-H. Lee, "Vocabulary independent discriminative utterance

verification for non-keyword rejection in subword based speech recognition,” *IEEE Trans. on Speech and Audio Processing*, vol. 4, No. 6, pp. 420-429, Nov. 1996.

[48] C.-S. Huang, H.-C. Wang, and C.-H. Lee, “A study on model-based error rate estimation for automatic speech recognition,” *IEEE Trans. on Speech and Audio Processing*, vol. 11, pp. 581-589, Nov. 2003.

[49] M. Rahim and C.-H. Lee, “String-based minimum verification error (SB-MVE) training for speech recognition,” *Computer Speech and Language*, vol. 11, pp. 147-160, 1997.

[50] E. L. Lehmann, *Testing Statistical Hypothesis*, Wiley, New York, 1959.

[51] S. Katagiri, B.-H. Juang, and C.-H. Lee, “Pattern Recognition Using A Generalized Probabilistic Descent Method,” *Proc. IEEE*, vol. 86, No. 11, pp. 2345-2373, Nov. 1998.

[52] C.-H. Lee, “From knowledge-ignorant to knowledge-rich modeling: a new speech research paradigm for next generation automatic speech recognition,” *Proc. ICSLP 2004*, Jeju, South Korea, Oct. 2004.

[53] C.-H. Lee, “A tutorial on speaker and speech verification,” *Proc. NORISIG*, Vigso, Denmark, 1998.

[54] A. E. Rosenberg, J. DeLong, C.-H. Lee, B.-H. Juang, and F. K. Soong, “The use of cohort normalized scores for speaker recognition,” *Proc. ICSLP 1992*, pp. 599-602, Banff, Oct. 1992.

[55] J. S. Garofolo, L. F. Lamel, W. M. Fisher, J. G. Fiscusk, D. S. Pallett, and N. L. Dahlgren, , “DARPA TIMIT acoustic-phonetic continuous speech corpus,” U.S. Dept. of Commerce, NIST, Gaithersburg, MD, Feb. 1993.

- [56] N. Niyogi and P. Ramesh, "A detection framework for locating phonetic events," *Proc. ICSLP 1998*, Sydney, 1998.
- [57] C. Ma and C.-H. Lee "Speaker verification based on combining speaker individuality parameter selection and decision," *ASRU 2005*, pp. 71-74, Dec. 2005.
- [58] D. V. Compennolle, "Noise adaptation in a hidden Markov model speech recognition system," *Comput. Speech and Lang.*, vol. 3, pp.151-167, 1989.
- [59] S. F. Boll, "Suppression of acoustic noise in speech using spectral subtraction," *IEEE Trans. Acoustics, Speech, Signal Processing*, vol. 27, pp.113-120, Apr. 1979.
- [60] L. Arslan, A. McCree, and V. Viswanathan, "New methods for adaptive noise suppression," *Proc. ICASSP 1995*, pp. 812-815, May 1995.
- [61] B. Atal, "Effectiveness of linear prediction characteristics of the speech wave for automatic speaker identification and verification," *J. Acoust. Soc. Amer.*, vol. 55, pp. 1304-1312, June 1974.
- [62] S. Furui, "Cepstral analysis technique for automatic speaker verification," *IEEE Trans. Acoustics., Speech, Signal Processing*, vol. 29, no. 2, pp. 254-272, 1981.
- [63] F. Liu, R. Stern, X.-D. Huang, and A. Acero, "Efficient cepstral normalization for robust speech recognition," *Proc. ARPA Human Language Technology Workshop*, March, 1993.
- [64] C.-W. Hsu and L.-S. Lee, "Higher order cepstral moment normalization (HOCMN) for robust speech recognition," *Proc. ICASSP 2004*, pp.197-200, May 2004.
- [65] Y. H. Suk, S. H. Choi, and H. S. Lee, "Cepstrum third-order normalization method for noisy speech recognition," *Electronics Letters*, vol. 35, no. 7, pp. 527-528, 1999.
- [66] D. Macho, L. Mauuary, B. Noe, Y. M. Cheng, D. Ealey, D. Jouver, H. Kelleher, D.

Pearce, and F. Saadoun, "Evaluation of a noise-robust DSR front-end on Aurora databases," *Proc. ICSLP 2002*, pp. 17-20, Denver, 2002.

[67] A. Acero, "Acoustical and environmental robustness in automatic speech recognition," *Ph.D. Dissertation*, ECE, Department, CMU, Sept. 1990.

[68] L. Deng, J. Droppo, and A. Acero, "Recursive estimation of nonstationary noise using iterative stochastic approximation for robust speech recognition," *IEEE Trans. Speech Audio Processing*, vol. 11, pp.568-580, Nov.2003.

[69] M. J. F. Gales, "Maximum likelihood linear transformations for HMM-based speech recognition," *Technical report, TR 291, Cambridge University*, 1997.

[70] K. Visweswariah, V. Goel, and R. Gopinath, "Structuring linear transforms for adaptation using training time information," *Proc. ICASSP 2002*, pp. 585-588, 2002.

[71] C.-H. Lee and Q. Huo, "On adaptive decision rules and decision parameter adaptation for automatic speech recognition," *Proc. IEEE*, vol. 88, pp. 1241-1269, 2000.

[72] O. Siohan, C. Chesta, and C.-H. Lee, "Hidden Markov model adaptation using maximum *a posteriori* linear regression," *Proc. Workshop Robust Methods for Speech Recognition in Adverse Conditions*, pp. 147-150, 1999.

[73] Y. Gong, "A method of joint compensation of additive and convolutive distortions for speaker-independent speech recognition," *IEEE Trans. Speech Audio Processing*, vol. 13, pp. 975-983, 2005.

[74] A. Acero, L. Deng, T. Kristjansson, and J. Zhang, "HMM adaptation using vector Taylor series for noisy speech recognition," *Proc. ICSLP 2002*, pp. 869-872, 2000.

[75] J. Li , L. Deng, D. Yu, Y. Gong, and A. Acero, "High-performance HMM adaptation with joint compensation of additive and convolutive distortions via vector Taylor series,"

ASRU 2007, pp. 65-70, 2007.

[76] K. Shinoda and C.-H. Lee, "A structural Bayes approach to speaker adaptation," *IEEE Trans. Speech Audio Processing*, vol. 9, no. 3, pp. 276-287, Mar. 2001.

[77] O. Siohan, C. Chesta, and C.-H. Lee, "Joint maximum a posteriori adaptation of transformation and HMM parameters," *IEEE Trans. Speech Audio Processing*, vol. 9, pp.417-428, May.2001..

[78] C.-H. Lee, "On stochastic feature and model compensation approaches to robust speech recognition," *Speech Commun.*, vol.25, pp.29-47, 1998.

[79] T. Anastasakos, J. McDonough, R. Schwartz, and J. Makhoul, "A compact model for speaker-adaptive training," *Proc. ICSLP 1996*, pp. 1137-1140, 1996.

[80] K.-T. Chen, W.-W. Liao, H.-M. Wang, and L.-S. Lee, "Fast speaker adaptation using eigenspace-based maximum likelihoods linear regression," *Proc. ICSLP 2000*, 2000.

[81] B. Mak, J. T. Kwok, and S. Ho, "Kernel eigenvoice speaker adaptation," *IEEE Trans. Speech Audio Processing*, vol. 13, no. 5, pp. 984-992, Sep. 2005.

[82] B. Mak and R. Hsiao, "Kernel Eigenspace-based MLLR adaptation," *IEEE Trans. Speech Audio Processing*, vol. 15, pp. 784-795, Mar. 2007.

[83] A. P. Dempster, N. M. Laird, and D.B. Rubin, "Maximum likelihood from incomplete data via the EM algorithm," *J. Roy. Statist. Soc. B*, vol. 39, pp. 1-38, 1977.

[84] M. Rahim and B.-H. Juang, "Signal bias removal by maximum likelihood estimation for robust telephone speech recognition," *IEEE Trans. Speech Audio Processing*, vol. 4, pp. 19-30, Jan. 1996.

- [85] A. P. Varga, H. J. M. Steeneken, M. Tomlinson, and D. Jones, "The NOISEX-92 study on the effect of additive noise on automatic speech recognition," *Tech. Rep.*, 1992.
- [86] D. Pearce and H.-G. Hirsch, "The Aurora experimental framework for the performance evaluation of speech recognition systems under noisy conditions," *Proc. ISCA ITRW ASR'2000*.
- [87] B. Mak, T.-C. Lai, and R. Hsiao "Improving reference speaker weighting adaptation by the use of maximum-likelihood reference speakers," *Proc. ICASSP 2006*, vol.1, pp. 229-232, May 2006.
- [88] V. Valtchev, J. Odell, P. C. Woodland and S. Young, "MMIE training of large vocabulary recognition systems," *Speech Communication*, vol. 22, no. 4, pp. 303-314, 1997.
- [89] D. Povey and P. C. Woodland, "Minimum phone error and i-smoothing for improved discriminative training," *Proc. ICASSP 2002*, pp. I105-I108, 2002.
- [90] V. Vapnik, *The Nature of Statistical Learning Theory*, Springer-Verlag, New York, 1995.
- [91] A. J. Hayter, *Probability and Statistics for Engineers and Scientists*, Duxbury Press; 3 edition (February 3, 2006).
- [92] Alan Agresti (Author), Christine A. Franklin, *Statistics: The Art and Science of Learning from Data (MyStatLab Series)*, Prentice Hall; 2 edition (March 2, 2008).
- [93] J. Wu and Q. Huo, "Several HKU approaches for robust speech recognition and their evaluation on Aurora connected digit recognition tasks," *Proc. Eurospeech 2003*, 2003.
- [94] Y. Tsao, J. Li, and C.-H. Lee, "A study on separation between acoustic models and its applications," *Proc. Interspeech 2005*, pp. 1109-1112, 2005.

- [95] A. Bruce, H. Y. Gao, and W. Stuetzle, "Wavelet denoising: a comparison of subset-selection and ensemble methods," *Statistica Sinica*, vol. 9, pp. 167-182, 1999.
- [96] L. Breiman, "Heuristics of instability and stabilization in model selection," *Annals of Statistics*, vol. 24, pp.2350-2383, Dec., 1996.
- [97] P. Nguyen, P. Gelin, J.-C. Junqua, and J.-T. Chien, "N-best based supervised and unsupervised adaptation for native and non-native speakers in cars," *Proc. ICASSP 1997*, pp. 257-265, 1997.
- [98] J. Wu, Eric Chang "Cohorts based custom models for rapid speaker and dialect adaptation", *Proc. Eurospeech 2001*, vol. 2, pp. 1261-1264, 2001.
- [99] I. T. Jolliffe, *Principal Component Analysis*. Berlin, Germany: Springer-Verlag, 1986.
- [100] M. W. Koo, C. H. Lee, and B. H. Juang, "Speech recognition and utterance verification based on a generalized confidence score," *IEEE Trans. Speech Audio Processing*, vol. 9, pp. 821-832, Nov. 2001.
- [101] H. Jiang, X. Li, and C. Liu, "Large margin hidden Markov models for speech recognition," *IEEE Trans. on Audio, Speech, and Language Processing*, vol. 14, no. 5, pp. 1584-1595, Sept. 2006.
- [102] J. Wu and Q. Huo, "A study of minimum classification error (MCE) linear regression for supervised adaptation of MCE-trained continuous-density hidden Markov models," *IEEE Trans. Speech Audio Processing*, vol. 15, pp. 478-488, 2007.
- [103] X. He and W. Chou, "Minimum classification error linear regression for acoustic model adaptation of continuous density HMMs," *Proc. ICASSP 2003*, vol. 1, pp. 556-559, Apr. 2003.

- [104] A. Agarwal, H.M. El-Askary, T. El-Ghazawi, M. Kafatos, and J. Le-Moigne, “Hierarchical PCA Techniques for Fusing Spatial and Spectral Observations With Application to MISR and Monitoring Dust Storms,” *IEEE Geoscience and Remote Sensing Letters*, vol. 4, pp. 678-682, Oct. 2007.
- [105] A. Boomershine, “The perceptual processing of variable input in Spanish: An exemplar-based approach to speech perception,” dissertation, Ohio State University.
- [106] J. Pierrehumbert, “Exemplar dynamics: word frequency, lenition, and contrast,” *Frequency effects and the emergence of linguistic structure*, pp. 137-157.
- [107] K. Johnson, “Decisions and mechanisms in exemplar-based phonology,” *Experimental Approaches to Phonology*, pp. 25-40, Oxford University Press.
- [108] K. Johnson, “Speech perception without speaker normalization: an exemplar model,” *Talker Variability in Speech Processing*, pp. 145-165, Academic Press.
- [109] K. Johnson, “Speaker normalization in speech perception,” *The handbook of speech perception*, pp. 363–389, Oxford, UK: Blackwell.
- [110] J. Elman and J. McClelland, “The TRACE model of speech perception,” *Cognitive Psychology* 18, pp. 1-86.
- [111] R. G. Leonard, “A database for speaker-Independent digit recognition,” *Proc. ICASSP 1984*, pp. 328-331, 1984.
- [112] “Recommendation G.712—Transmission performance characteristics of pulse code modulation channels,” International Telecommunication Union (ITU), Geneva, Switzerland, November 1996.
- [113] ITU Recommendation P.830, Subjective Performance Assessment Telephone Band Wideband Digital Codecs, February 1996.

VITA

Yu Tsao received his B.S. and M.S. degrees in electrical engineering from National Taiwan University (NTU), Taipei, Taiwan, R.O.C., in 1999 and 2001, respectively. Since 2003, he is a Ph.D. student under the supervision of Professor Chin-Hui Lee at the school of electrical and computer engineering, Georgia Institute of Technology, Atlanta, U.S. He has research experiences on detection-based speech recognition, noise robustness, speaker adaptation, and speaker recognition. His publications are listed in the following.

1. **Y. Tsao**, J. Li, and C.-H. Lee, "Ensemble speaker and speaking environment modeling approach with advanced online estimation process," submitted to *ICASSP 2009*.
2. **Y. Tsao** and C.-H. Lee, "An ensemble speaker and speaking environment modeling approach to robust speech recognition," accepted, to be appeared in *IEEE Trans. on Audio, Speech, and Language Processing*.
3. **Y. Tsao** and C.-H. Lee, "Improving the ensemble speaker and speaking environment modeling approach by enhancing the precision of the online estimation process," *Proc. Interspeech 2008*, pp. 1265-1268, 2008.
4. S.-Y. Peng, **Y. Tsao**, P. E. Hasler, and D. V. Anderson "A programmable analog radial-basis-function based classifier," *Proc. ICASSP 2008*, pp. 1425-1428, 2008.
5. **Y. Tsao** and C.-H. Lee, "An ensemble modeling approach to joint characterization of speaker and speaking environments," *Proc. Interspeech 2007*, pp. 1050-1053, 2007.
6. I. Bromberg, Q. Fu, J. Hou, J. Li, C. Ma, B. Matthews, A. Moreno-Daniel, J. Morris, S. M. Siniscalchi, **Y. Tsao**, and Y. Wang, "Detection-based ASR in the automatic speech attribute transcription project," *Proc. Interspeech 2007*, pp. 1829-1832, 2007.
7. **Y. Tsao** and C.-H. Lee, "Two extensions to ensemble speaker and speaking

- environment modeling for robust automatic speech recognition,” *ASRU 2007*, pp. 77-80, 2007.
8. **Y. Tsao** and C.-H. Lee, “A vector space approach to environment modeling for robust speech recognition,” *Proc. Interspeech 2006*, pp. 785-788, 2006.
 9. C. Ma, **Y. Tsao**, and C.-H. Lee, “A study on detection based automatic speech recognition,” *Proc. Interspeech 2006*, pp. 2350-2353, 2006.
 10. J. Li, **Y. Tsao**, and C.-H. Lee, “A study on knowledge source integration for candidate rescoring in automatic speech recognition,” *Proc. ICASSP 2005*, pp. I837-I840, 2005.
 11. **Y. Tsao**, S.-M. Lee, and L.-S. Lee, “Segmental eigenvoice with delicate eigenspace for improved speaker adaptation,” *IEEE Trans. Speech Audio Processing*, vol. 13, pp.399-411, May 2005,
 12. **Y. Tsao**, J. Li, and C.-H. Lee, “A study on separation between acoustic models and its applications,” *Proc. Interspeech 2005*, pp. 1109-1112, 2005.
 13. **Y. Tsao**, S.-M. Lee, and L.-S. Lee, “Segmental eigenvoice for rapid speaker adaptation,” *Proc. Eurospeech 2001*, pp. 1269-1272, 2001.
 14. K. Yao and **Y. Tsao**, “Efficient speech recognition with cluster methods,” TI disclosure number: TI-63194, (patent pending).