SOFT MARGIN ESTIMATION FOR AUTOMATIC SPEECH RECOGNITION

A Dissertation
Presented to
The Academic Faculty

By

Jinyu Li

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

School of Electrical and Computer Engineering
Georgia Institute of Technology
December 2008

SOFT MARGIN ESTIMATION FOR AUTOMATIC SPEECH RECOGNITION

Approved by:

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

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

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

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

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

Date Approved: August 2008

ACKNOWLEDGEMENTS

First, I would like to greatly appreciate my advisor, Prof. Chin-Hui Lee, for his great supports and guides during my Ph.D. study. His experience and insight in the speech research area benefit my research a lot from the discussion with him. These years, I enjoyed his education style. Without his supervision, I couldn't finish this thesis on time. I would express my sincere gratitude to Prof. Biing-Hwang (Fred) Juang and Prof. Mark Clements. I have benefited a lot from them in joint projects. I would also like to thank Prof. Anthony Joseph Yezzi and Prof. Ming Yuan for serving on my committee. Special thanks are owed to Prof. Yuan for the fruitful discussion and the joint work in machine learning.

I am grateful to researchers in Microsoft, especially Dr. Li Deng and Dr. Yifan Gong. My summer intern work with Dr. Deng is enjoyable and fruitful. The enormous helps and encouragements from Dr. Gong enable me to continue my career on speech research.

I would also thank all my friends for their great supports during these years. First, I am grateful to my colleagues Yu Tsao, Chengyuan Ma, Xiong Xiao, Sibel Yaman, Sabato Marco Siniscalchi, Qiang Fu, Brett Matthews, Yong Zhao, and Jeremy Reed for the collaboration in different projects. Yu and Xiong have applied SME on their research areas, and demonstrated great advantages of SME. Second, great appreciations are owed to my friends Hua Xu, Wei Zhou, Jian Zhu, Chunpeng Xiao, Junlin Li, Wei Zhang, Kun Shi, and Chunming Zhao. I would also like to thank my previous colleagues in USTC iflytek speech lab for their long term support and friendship. They are Zhijie Yan, Yu Hu, Bo Liu, Xiaobing Li, and Gang Guo. I especially thank Zhijie for his enormous help and collaboration in the work to apply SME on LVCSR tasks.

Finally, I would like to express my deepest gratitude to my parents and my wife, Lingyan, for their great love and consistent help during my Ph.D. study.

TABLE OF CONTENTS

ACKNOWLEDGEMENTS iii

LIST OF TABLES vi

LIST OF FIGURES ix

CHAPTER 1 SCIENTIFIC GOALS 1

CHAPTER 2 BACKGROUND 6
2.1 Background of automatic speech recognition 6
2.1.1 Feature Extraction 7
2.1.2 Language Modeling 8
2.1.3 Acoustic Modeling 8
2.2 Conventional Discriminative Training 10
2.2.1 Maximum Mutual Information Estimation (MMIE) 10
2.2.2 Minimum Classification Error (MCE) 10
2.2.3 Minimum Word/Phone Error (MWE/MPE) 11
2.2.4 Gradient-Based Optimization 12
2.2.5 Extended Baum-Welch (EBW) Algorithm 14
2.3 Empirical Risk 15
2.4 Test Risk Bound 16
2.5 Margin-based Methods in Automatic Speech Recognition 18
2.5.1 Large Margin Estimation 18
2.5.2 Large Margin Gaussian Mixture Model and Hidden Markov Model 20

CHAPTER 3 SOFT MARGIN ESTIMATION (SME) 22
3.1 Approximate Test Risk Bound Minimization 22
3.2 Loss Function Definition 24
3.3 Separation Measure Definition 24
3.4 Solutions to SME 26
3.4.1 Derivative Computation 27
3.5 Margin-Based Methods Comparison 28
3.6 Experiments 31
3.6.1 SME with Gaussian Mixture Model 31
3.6.2 SME with Hidden Markov Model 33
3.7 Conclusion 41

CHAPTER 4 SOFT MARGIN FEATURE EXTRACTION (SMFE) 43
4.1 SMFE for Gaussian Observations 44
4.2 SMFE for GMM Observations 46
4.3 Implementation Issue 47
4.4 Experiments 47

4.5 Conclusion 49

CHAPTER 5 SME FOR ROBUST AUTOMATIC SPEECH RECOGNITION 50
5.1 Clean Training Condition 52
5.2 Multi-condition Training Condition 56
5.3 Single SNR Training Condition 57
5.3.1 20db SNR Training Condition 59
5.3.2 15db SNR Training Condition 59
5.3.3 10db SNR Training Condition 61
5.3.4 5db SNR Training Condition 63
5.3.5 0db SNR Training Condition 64
5.4 Conclusion 64

CHAPTER 6 THE RELATIONSHIP BETWEEN MARGIN AND HMM PARAMETERS 67
6.1 Mapping between SME and SVMs 68
6.2 SME with Further Generalization 69
6.2.1 Derivative Computation 73
6.3 Comparison with Minimum Divergence Training 74
6.4 Experiments 75
6.5 Conclusion 76

CHAPTER 7 SME FOR LARGE VOCABULARY CONTINUOUS SPEECH RECOGNITION 78
7.1 SME for LVCSR with the Generalized Probabilistic Descent Algorithm . . 78
7.1.1 Experiment of SME with GPD 81
7.2 SME for LVCSR with the Extended Baum-Welch Algorithm 82
7.2.1 SME with Utterance Selection 83
7.2.2 SME with Frame Selection 84
7.2.3 Implementation with EBW 86
7.2.4 Initial Experiments of SME with EBW 86
7.2.5 SME with Various Separation Levels 91
7.2.6 Practical Implementation Issues 95
7.2.7 Experiments of SME with Various Separation Levels 96
7.3 Conclusion 101

CHAPTER 8 CONCLUSION 104

VITA 122

LIST OF TABLES

Table 2.1 Discriminative training target function and loss function 16

Table 3.1 Separation measure for SME 26

Table 3.2 Comparison of margin-based methods 29

Table 3.3 SME: Testing set string accuracy comparison with different methods. Accuracies marked with an asterisk are significantly different from the accuracy of the SME model ($p<0.025$, paired Z-test, 8700 d.o.f. [60]). 35

Table 3.4 Margin value assignment. 35

Table 3.5 Margin value obtained by joint optimization. 35

Table 3.6 Comparison of GPD optimization and Quickprop optimization for SME. 41

Table 4.1 SMFE: Testing set string accuracy comparison with different methods. Accuracies marked with an asterisk are significantly different from the accuracy of the SMFE model ($p<0.1$, paired Z-test, 8700 d.o.f. [60]). . 48

Table 5.1 Detailed test accuracies for MLE, MCE, and SME with different balance coefficient λ using clean training data. 53

Table 5.2 Relative WER reductions for MCE, and SME from MLE baseline using clean training data. 53

Table 5.3 Detailed accuracies on testing set a, b, and c for MLE and SME with different balance coefficient λ using clean training data. 55

Table 5.4 Detailed test accuracies for MLE, MCE, and SME using multi-condition training data. 57

Table 5.5 Relative WER reductions for MCE and SME from MLE baseline using multi-condition training data. 58

Table 5.6 Detailed accuracies on testing set a, b, and c for SME with different balance coefficient λ using multi-condition training data. 58

Table 5.7 Detailed test accuracies for MLE and SME using 20db SNR training data. 59

Table 5.8 Relative WER reductions for SME from MLE baseline using 20db SNR training data. 59

Table 5.9 Detailed accuracies on testing set a, b, and c for SME with different balance coefficient λ using 20db SNR training data. 59

Table 5.10 Detailed test accuracies for MLE and SME using 15db SNR training data. 60

Table 5.11 Relative WER reductions for SME from MLE baseline using 15db SNR training data. 60

Table 5.12 Detailed accuracies on testing set a, b, and c for SME with different balance coefficient λ using 15db SNR training data. 61

Table 5.13 Detailed test accuracies for MLE and SME using 10db SNR training data. 62

Table 5.14 Relative WER reductions for SME from MLE baseline using 10db SNR training data. 62

Table 5.15 Detailed accuracies on testing set a, b, and c for SME with different balance coefficient λ using 10db SNR training data. 62

Table 5.16 Detailed test accuracies for MLE and SME using 5db SNR training data. 63

Table 5.17 Relative WER reductions for SME from MLE baseline using 5db SNR training data. 63

Table 5.18 Detailed accuracies on testing set a, b, and c for SME with different balance coefficient λ using 5db SNR training data. 64

Table 5.19 Detailed test accuracies for MLE and SME using 0db SNR training data. 64

Table 5.20 Relative WER reductions for SME from MLE baseline using 0db SNR training data. 65

Table 5.21 Detailed accuracies on testing set a, b, and c for SME with different balance coefficient λ using 0db SNR training data. 65

Table 6.1 Testing set string accuracy comparison with different methods. 75

Table 6.2 Square root of system divergence (Eq. (6.11)) with different methods. . 76

Table 7.1 Separation measure for SME 79

Table 7.2 Correct and competing words for lattice example 80

Table 7.3 Performance on the 5k-WSJ0 task 82

Table 7.4 Performance comparison for discriminative training methods with EBW on the 5k-WSJ0 task. 89

Table 7.5 Correct and competing paths for word-level separation in Figure 7.6. . 93

Table 7.6 Performance comparison for most discriminative training methods with EBW on the 5k-WSJ0 task. 97

LIST OF FIGURES

Figure 2.1 Flowchart of an automatic speech recognition system. 7

Figure 2.2 Large margin estimation. 20

Figure 3.1 Soft margin estimation. 23

Figure 3.2 EER evolutions for the NIST 03 30-second test set. 33

Figure 3.3 EER evolutions for the NIST 05 30-second test set. 34

Figure 3.4 String accuracy of SME for different models in the TIDIGITS training set. 36

Figure 3.5 The histogram of separation distances of 1-mixture MLE model in the TIDIGITS training set. 36

Figure 3.6 The histogram of separation distances of 1-mixture SME model in the TIDIGITS training set. 37

Figure 3.7 The histogram of separation distances of 16-mixture SME model in the TIDIGITS training set. 37

Figure 3.8 The histogram of separation distances of 16-mix model of MLE, MCE, and SME in the TIDIGITS testing set. The short dashed curve, line curve, and dotted curve correspond to MLE, MCE, and SME models. . 40

Figure 3.9 The histogram of separation distances of 1-mix model of MLE, MCE, and SME in the TIDIGITS testing set. The short dashed curve, line curve, and dotted curve correspond to MLE, MCE, and SME models. . 40

Figure 4.1 Conventional feature extraction for MFCC 43

Figure 4.2 Soft margin feature extraction: jointly optimize feature and HMM parameters. 44

Figure 5.1 Methods for robust speech recognition 50

Figure 5.2 The distortion level from clean condition for different SNRs. 55

Figure 5.3 The distortion level from a 10db SNR training condition for different SNRs. 58

Figure 6.1 A binary separable case of SVMs. 68

Figure 6.2 Divergence computation for HMM/GMM systems. 71

Figure 7.1 Lattice example: the top lattice is obtained in decoding, and the bottom is the corresponding utterance transcription. 80

Figure 7.2 The histogram of the separation measure *d* in Eq. (7.8) of MLE model on training set. 88

Figure 7.3 The histogram of the separation measure *d* in Eq. (7.8) of SME_u model on training set. 88

Figure 7.4 The histogram of the frame posterior probabilities of MLE model on training set. 90

Figure 7.5 The histogram of the frame posterior probabilities of SME_fc model on training set. 90

Figure 7.6 A lattice example to distinguish the string-level separation with the word-level separation 94

Figure 7.7 Evolutions of testing WER for MPE, SME_Phone, and Phone_Sep models on the 5k-WSJ0 task. 98

Figure 7.8 Evolutions of testing WER for MWE and SME_Word models on the 5k-WSJ0 task. 99

Figure 7.9 Evolutions of testing WER for MMIE, MCE, and SME_String models on the 5k-WSJ0 task. 99

Figure 7.10 The histogram of the frame posterior probabilities of MLE model on training set with word-level separation. 100

Figure 7.11 The histogram of the frame posterior probabilities of SME_Word model on training set with word-level separation. 101

Figure 7.12 The histogram of the frame posterior probabilities of MLE model on training set with phone-level separation. 102

Figure 7.13 The histogram of the frame posterior probabilities of SME_Phone model on training set with phone-level separation. 103

CHAPTER 1

SCIENTIFIC GOALS

With the prevailing usage of hidden Markov models (HMMs), rapid progress in automatic speech recognition (ASR) has been witnessed in the last two decades. Usually, HMM parameters are estimated by the traditional maximum likelihood estimation (MLE) method. MLE is known to be optimal for density estimation, but it often does not lead to minimum recognition error, which is the goal of ASR. As a remedy, several discriminative training (DT) methods have been proposed in recent years to boost ASR system accuracy. Typical methods are maximum mutual information estimation (MMIE) [6], [92], [118]; minimum classification error (MCE) [48], [83], [107]; and minimum word/phone error (MWE/MPE) [101]. MMIE training separates different classes by maximizing approximate posterior probabilities. On the other hand, MCE directly minimizes approximate string errors, while MWE/MPE attempts to optimize approximate word and phone error rates. If the acoustic conditions in the testing set match well with those in the training set, these DT algorithms usually achieve very good performance when tested. However, such a good match cannot always be expected for most practical recognition conditions. To avoid the problem of over-fitting on the training set, regularization is achieved by using “I-smoothing” [101] in MMIE and MWE/MPE, while MCE exploits a smoothing parameter in a sigmoid function for regularization [84], [85].

According to statistical learning theory [119], a test risk is bounded by the summation of two terms: an empirical risk (i.e., the risk on the training set) and a generalization function. The power to deal with possible mismatches between the training and testing conditions can often be measured by the generalization function. In particular, large margin learning frameworks, such as support vector machines (SVMs) [11], have demonstrated superior generalization abilities over other conventional classifiers. By securing a margin from the decision boundary to the nearest training sample, a correct decision can still be made if the mismatched testing sample falls within a tolerance region around the original training sample defined by the margin. The idea of SVMs is explored widely in speech research. Different kinds of kernels are employed in the area of speaker recognition, such as the work in [12], [79]. However, this kind of work cannot be easily incorporated into ASR because it is hard to combine with HMMs. SVMs were also used in the framework of landmark-based speech detection [46]; however this framework is not widely used because it deviates from the HMM paradigm. Some technologies (e.g., [26], [117]) loosely couple SVMs with HMMs by using SVMs instead of Gaussian mixture models (GMMs) as the state observation density of HMMs. These frameworks do not take full advantage of SVMs to get better generalization with a larger margin. A combination of SVMs and HMMs, called HM-SVMs, was explored in [3] with discrete distributions, but it is far from being a solution to the state-of-the-art ASR systems, whose state distributions are usually continuous densities. Moreover, like SVMs, HM-SVMs work on the problem of finding the optimal projection matrix. HM-SVMs differ from SVMs in that the observations of HM-SVMs are sequences instead of discrete samples. Therefore, HMMs are used to model the hidden state of the sequences in HM-SVMs. Obviously, HM-SVMs are too simple to solve the ASR problem.

Adopting the concept of enhancing margin separation, large margin estimation (LME) [45], [73] and its variant, large relative margin estimation (LRME) [77], of HMMs have been proposed. In essence, LME and LRME update models only with accurately classified samples. However, it is well known that misclassified samples are also critical for classifier learning. Recently, LRME was modified [78] to consider all of the training samples, especially to move the most incorrectly classified sample toward the direction of correct decision. However, this modification makes the algorithm vulnerable to outliers, and the idea of margin is not very meaningful. In [111], a large margin algorithm for learning GMMs was proposed, but makes some approximations to use GMMs instead of HMMs. More recently, the work of [111] was extended to deal with HMMs in [112] by summing the differences of Mahalanobis distances [21] between the models in the correct and competing strings and comparing the result with a Hamming distance. It is not clear whether it is suitable to directly compare the Hamming distance (the number of different labels of two strings) with the difference of Mahalanobis distance, which is the distance of two Gaussian models given an observation.

The research to integrate margin into ASR is still in its initial stage. The above-mentioned margin-based methods cannot be considered as perfect solutions. They were only reported to be successful in specified ASR tasks. For example, no report from those methods was given for the success on large vocabulary continuous speech recognition (LVCSR) tasks. The success on LVCSR tasks should be an important criterion to evaluate new ASR training methods. A new margin-based method with discriminative power and generalization ability should be designed to work well in different application situations.

The objective of this proposed research is to design a discriminative training method that makes direct use of the successful ideas of soft margin in support vector machines to improve generalization capability and decision feedback learning in discriminative training to enhance model separation in classifier design. The proposed method is called soft margin estimation (SME). This method will be demonstrated successfully on different kinds of tasks, such as spoken language recognition tasks and ASR tasks. SME will also be shown to outperform popular discriminative training methods not only on small ASR tasks but also on challenging ASR tasks, such as noisy ASR tasks and LVCSR tasks.

The research issues surrounding SME are to:

- Build a framework of SME from statistical learning theory. Formulate SME by designing unique separation measures, loss functions, and final optimization target functions.

- Work out different solutions for SME.

- Investigate SME for Gaussian mixture models in applications such as language identification.
- Investigate SME for hidden Markov models in ASR applications.
  - Initiate the theoretical work of SME on small ASR tasks.
  - Apply SME to LVCSR tasks.
  - Apply SME to robust speech recognition tasks.
  - Extend SME to joint optimization of HMM parameters and acoustic features.

To build the framework of SME from statistical learning theory, the famous test risk bound from Vapnik [119] needs to be studied. The object function of SME will be designed to reflect the essence of that bound.

Different solutions need to be worked out for SME. The generalized probabilistic descent (GPD) [50] is a convenient method to optimize parameters in relatively small tasks. In contrast, the extended Baum-Welch (EBW) algorithm [91] is a popular algorithm for LVCSR tasks. Both methods will be employed to solve the optimization problem for SME.

SME is targeted as a generalized machine learning method. Therefore, different kinds of applications will be explored. SME also needs to work with different kinds of models. The most popular models in the area of speech research are GMMs and HMMs. SME will be used to boost the performance of a language identification system, which uses GMMs as underlying models.

The major research in this study is to use SME to improve the discrimination and generalization of HMMs. The whole framework of SME will first be built and tested on relatively small tasks, such as TIDIGITS [61]. To be a discriminative training method with great influence, SME needs to be applied to LVCSR tasks.

SME is proposed to have a nice property of generalization. To test this property, SME will be applied to robust speech recognition, where there is a mismatch between the training and testing conditions.

Feature exaction is an important component in ASR systems. Most current ASR systems use Mel-frequency cepstrum coefficients (MFCCs) [16] as their standard input acoustic features, but MFCCs may not be optimal for all ASR tasks. SME will be extended to get optimal acoustic features and HMM parameters jointly.

This dissertation will be organized as follows. In Chapter 2, a very rough overview of ASR is first given. Then, two topics related with this dissertation are discussed. The first is conventional discriminative training methods in ASR. The second is statistical learning theory and previous margin-based methods. Chapter 3 presents the formulation of the proposed SME method, gives its solution, compares with other margin-based methods, and tests it on a connected-digit task and a spoken language recognition task. Chapter 4 extends SME to get optimal acoustic features and HMM parameters jointly. Chapter 5 discusses the generalization issue of discriminative training methods on a noise-robust ASR task. Chapter 6 tries to link the margin in SME with HMM parameters. Chapter 7 works on how to apply SME to LVCSR tasks. The EBW algorithm, the most widely used optimization method on LVCSR tasks, is adopted, comparing to the GPD method used in previous chapters. Different formulations of SME are studied and compared comprehensively with the popular discriminative training methods in ASR. Chapter 8 concludes the study of this dissertation.

CHAPTER 2

BACKGROUND

In this chapter, we first give a brief overview of automatic speech recognition (ASR) technology. Conventional discriminative training methods and the optimization methods are described. Then, we show that there is a gap between the empirical risk and the test risk. The theory of statistical learning explains this gap and gives insight into current state-of-the-art HMM learning algorithms for designing ASR systems. Two margin-based ASR algorithms are discussed.

2.1 Background of automatic speech recognition

The research of automatic speech recognition (ASR) has been developed for several decades [38], [42], [58], [104]. The goal of ASR is to get a word sequence W, given the spoken speech signal O. This is a decision problem, and can be formulated as a well-known maximum *a posteriori* problem:

$$\tilde{W} = arg \max_{W} P_{\Lambda}(W) P_{\Gamma}(O|W), \tag{2.1}$$

where $P(O|W)$ is the acoustic model (AM) likelihood and $P(W)$ is the language model (LM) probability. They are characterized by the AM parameter, Λ, and the LM parameter, Γ. The state of the art ASR systems usually use the framework in Figure 2.1. The speech signal is first processed by the feature extraction module to get the acoustic feature. The feature extraction module is often referred as the front end of ASR systems. The acoustic feature will be passed to acoustic model and language model to compute the likelihood score. The output is a word sequence with the largest probability from AM and LM. The combination of AM and LM are usually referred as the back end of ASR systems.

Given a training set, we can estimate the AM parameter, $\hat{\Lambda}$, and the LM parameter, $\hat{\Gamma}$. Then these estimated parameters are plugged into the maximum *a posteriori* decision

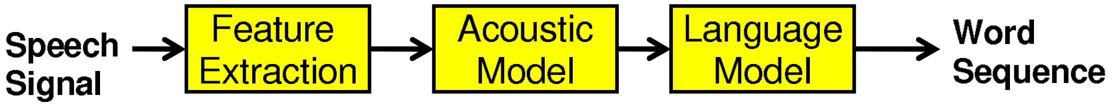

Figure 2.1. Flowchart of an automatic speech recognition system.

process as:

$$\hat{W} = arg \max_{W} P_{\hat{\Lambda}}(W)P_{\hat{\Gamma}}(O|W). \tag{2.2}$$

2.1.1 Feature Extraction

The most widely used acoustic features are Mel-frequency cepstrum coefficients (MFCCs) [16] and perceptual linear prediction (PLP) [34]. Both features are perceptually motivated representations and have been used successfully in most ASR systems. However, if the training and testing conditions are severely mismatched, these features cannot work well. Therefore, feature-domain methods are proposed to enhance the distorted speech with advanced signal processing methods on noise-robust ASR tasks. Spectral subtraction (SS) [10] is widely used as a simple technique to reduce additive noise in the spectral domain. Cepstral Mean Normalization (CMN) [5] removes the mean vector in the acoustic features of the utterance in order to reduce or eliminate the convolutive channel effect. As an extension to CMN, Cepstral Variance Normalization (CVN) [86] also adjusts the feature variance to improve ASR robustness. Relative spectra (RASTA) [35] employs a long span of speech signals in order to remove or reduce the acoustic distortion. All these traditional feature-domain methods are relatively simple, and are shown to have achieved medium-level distortion reduction. In recent years, new feature-domain methods have been proposed using more advanced signal processing techniques to achieve more significant performance improvement in noise-robustness ASR tasks than the traditional methods. Examples include feature space non-linear transformation techniques [86],[94], the ETSI advanced front end (AFE) [82], and stereo-based piecewise linear compensation for environments (SPLICE) [17].

2.1.2 Language Modeling

Language model provides a way to estimate the probability of a possible word sequence. In current large vocabulary continue speech recognition (LVCSR) systems, the overwhelming language model technology is still the n-gram language model. The kernel issue for n-gram language model is how to deal with data sparseness. The general technology is smoothing, which is widely used in these years' LVCSR systems (e.g., [116], [52]). The popular smoothing algorithms are deleted interpolation smoothing [43], Katz smoothing [51], Witten-Bell smoothing [120], and Kneser-Ney smoothing [53]. According to [14], Kneser-Ney smoothing works better than other classical smoothing method because of its unique back-off distribution where a fixed discount was subtracted from each nonzero count. In [14], detailed comparison of different smoothing methods was given and modified Kneser-Ney smoothing was proposed and outperformed all other smoothing methods.

2.1.3 Acoustic Modeling

Acoustic model is used to characterize the likelihood of acoustic feature with respect to (w.r.t.) the underlying word sequence. The research in ASR has been fast developed since hidden Markov models (HMMs) were introduced [47], [102]. HMMs gracefully handle the dynamic time evolution of speech signals and characterize it as a parametric random process. HMMs are now used in most of the state of the art ASR systems. According to how the acoustic emission probabilities are modeled for the HMMs' state, we can have discrete HMMs [76], semi-continuous HMMs [39], and continuous HMMs [62]. For the continuous output density, the most popular used one is Gaussian mixture model (GMM), in which the state output density is:

$$P_\Lambda(o) = \sum_i c_i N(o; \mu_i, \sigma_i^2), \tag{2.3}$$

where $N(o; \mu_i, \sigma_i^2)$ is a Gaussian with mean μ_i and variance σ_i^2, and c_i is the weight for the i-th Gaussian component. Three fundamental problems of HMMs are probability evaluation, determination of the best state sequence, and parameter estimation [102]. The probability evaluation can be realized easily with the forward algorithm [102].

The determination of the best state sequence is often referred as a decoding or search process. Viterbi search [87] and A^* search [96] are two major search algorithms. Search is an important issue for implementing LVCSR in realtime. A lot of methods have been proposed for fast decoding, such as beam search [89], lexical tree [88], factored language model probabilities [2], language model look-ahead [93], fast likelihood computation [9], and multi-pass search [109].

The parameter estimation in ASR is solved with the well-known maximum likelihood estimation (MLE) [19] using a forward-backward procedure [103]. The quality of acoustic model is the most important issue for ASR.

MLE is known to be optimal for density estimation, but it often does not lead to minimum recognition error, which is the goal of ASR. As a remedy, several discriminative training (DT) methods have been proposed in recent years to boost ASR system accuracy. Typical methods are maximum mutual information estimation (MMIE) [6], [92], [118]; minimum classification error (MCE) [48], [83], [107]; and minimum word/phone error (MWE/MPE) [101]. These methods will be discussed in the next section.

Model adaptation is another important research issue for acoustic modeling. Adaptation is used to modify the original trained model to fit a new speaker, style, or environment. Maximum likelihood linear regression (MLLR) [59] and maximum a posteriori adaptation (MAP) [29] are two most popular adaptation methods for speaker adaptation. Both adaptation methods have some variants. For example, constrained MLLR considers the transformation of variance matrices [24]. Structural MAP uses a hierarchical tree to automatically allocate adaptation data [113]. MLLR-MAP combines the advantage of MLLR and MAP together [115]. Speaker adaptive training (SAT) [4] is proposed to extend MLLR with better generative models. For environment adaptation, in addition to MLLR and MAP, parallel model combination (PMC) [25] and joint compensation of additive and convolutive distortions (JAC) with vector Taylor series (VTS) approximation [1], [31], [64] are widely used.

2.2 Conventional Discriminative Training

As stated in [49], modern ASR can be viewed with a communication-theoretic formulation. Because of lots of variations in the generation process for the speech signal O, we do not know what the real forms of $P(O|W)$ and $P(W)$ are. We also cannot have an explicit knowledge of the AM parameter Λ and LM parameter Γ. Therefore, the estimated parameters used in the plug-in decision process (Eq. (2.2)) may not be optimal for ASR. We need better models for the purpose of recognition. Discriminative training were then proposed to get better models. Typical methods are maximum mutual information estimation (MMIE) [6], [92], [118]; minimum classification error (MCE) [48], [83], [107]; and minimum word/phone error (MWE/MPE) [101].

2.2.1 Maximum Mutual Information Estimation (MMIE)

MMIE training separates different classes by maximizing approximate posterior probabilities. It maximizes the following objective function:

$$\sum_{i=1}^{N} log \frac{P_{\Lambda}(O_i|S_i)P(S_i)}{\sum_{\hat{S}_i} P_{\Lambda}(O_i|\hat{S}_i)P(\hat{S}_i)}, \tag{2.4}$$

where S_i is the correct transcription and $\hat{S}_i$ is the possible string sequence for utterance O_i (may include the correct string). $P_{\Lambda}(O_i|\hat{S}_i)$ (or $P_{\Lambda}(O_i|S_i)$) and $P(\hat{S}_i)$ (or $P(S_i)$) are acoustic and language model scores, respectively. When used on LVCSR tasks, a probability scale is usually applied to get better generalization [122]. This is also applied to other discriminative training methods on LVCSR tasks.

2.2.2 Minimum Classification Error (MCE)

MCE directly minimizes approximate string errors. It defines the likelihood function for class i as

$$g_i(O_i, \Lambda) = logP_{\Lambda}(O_i|S_i). \tag{2.5}$$

A misclassification measure is then defined to separate the correct class from the competing classes [48]:

$$d_i(O_i, \Lambda) = -g_i(O_i, \Lambda) + log \left[\frac{1}{N-1} \sum_{j,j \neq i} expg_j(O_i, \Lambda)\eta \right]^{\frac{1}{\eta}} . \tag{2.6}$$

The misclassification measure can also be defined by taking into account the language model probability [81]:

$$d_i(O_i, \Lambda) = \sum_{i=1}^{N} log \frac{P_\Lambda(O_i|S_i)P(S_i)}{\sum_{\hat{S}_i \neq S_i} P_\Lambda(O_i|\hat{S}_i)P(\hat{S}_i)}. \tag{2.7}$$

Then, the misclassification measure is embedded into a sigmoid function:

$$\frac{1}{1 + exp(-\gamma d_i(O_i, \Lambda) + \theta)}. \tag{2.8}$$

γ and θ are parameters for a sigmoid function. Eq. (2.8) can be considered as a smoothed count for the number of misclassification utterance. This is because if an utterance is misclassified, Eq.(2.7) will have a positive value, resulting a value around 1 for Eq. (2.8). In contrast, a correct classified utterance will get a value around 0 for Eq. (2.8).

MCE can also be applied to optimize acoustic feature. For example, the cepstral lifter is optimized with MCE in [8].

It should be noted that the smoothed 0-1 count in Eq. (2.8) is very attractive. The same idea can be applied not only to speech recognition, but also to all kinds of applications (e.g., minimum verification error [105], maximal figure-of-merit [27], minimize call routing error [56], and discriminative language model training [55]) by mapping the target discrete count with a smoothed function of system parameters. This nice property is not easy got from MMIE or MPE.

2.2.3 Minimum Word/Phone Error (MWE/MPE)

MWE/MPE attempt to optimize approximate word and phone error rates. This is directly related with the target of ASR. Therefore, MWE/MPE have achieved great success in recent years. As a result, several variations have been proposed recently, such as minimum divergence training [20] and minimum phone frame error training [127].

The objective function of MPE is:

$$\sum_{i=1}^{N} \frac{\sum_{\hat{S}_i} P_\Lambda(O_i|\hat{S}_i) P(\hat{S}_i) RawPhoneAccuracy(\hat{S}_i)}{\sum_{\hat{S}_i} P_\Lambda(O_i|\hat{S}_i) P(\hat{S}_i)}, \tag{2.9}$$

where $RawPhoneAccuracy(\hat{S}_i)$ is the phone accuracy of the string $\hat{S}_i$ comparing with the ground truth S_i. For MWE, $RawWordAccuracy(\hat{S}_i)$ is used to replace $RawPhoneAccuracy(\hat{S}_i)$ in Eq. (2.9).

As an extension of MPE, feature-space minimum phone error (fMPE) [100] works on how to get the discriminative acoustic feature. The acoustic feature is modified online with an offset:

$$\tilde{O}_t = O_t + Wh_t, \tag{2.10}$$

where O_t is the original low-dimension acoustic feature, and W is a big matrix to project the high-dimension feature h_t to the low-dimension space. h_t is got by computing the posterior probabilities of each Gaussian in the system. Eq. (2.10) is then plugged into the framework of MPE for optimization. This method has achieved great success and shares a similar idea with SPLICE [17], [18], which is a successful method for robust speech recognition.

2.2.4 Gradient-Based Optimization

Gradient-based optimization methods are widely in MCE, and also was used in MMIE during its early developing stage. The most popular one is the generalized probabilistic descent (GPD) algorithm [50], which has a nice convergence property. The GPD algorithm takes the first-order derivative of the loss function L w.r.t. the model parameter Λ, and update the model parameter iteratively:

$$\Lambda_{t+1} = \Lambda_t - \eta_t \nabla L|_{\Lambda=\Lambda_t}. \tag{2.11}$$

Since the loss function L can be expressed as a function of the misclassification measure in Eq. (2.7), which is in turn a function of the density function, the key of Eq. (2.11) is to compute the derivative of the density function. Suppose we are using GMM as the density function for state j:

$$b_j(o) = \sum_k c_{jk} N(o; \mu_{jk}, \sigma_{jk}^2). \tag{2.12}$$

Then the following formulations for the derivatives w.r.t. mean and variance parameters are listed according to [48]. We will use them in the GPD optimization from Chapter 3 to Chapter 6.

For the l-th dimension of mean μ_{jk} and variance σ_{jk}^2, the following parameter transformation is used to kept the original parameter relationship intact after the parameter update.

$$\mu_{jkl} \rightarrow \tilde{\mu}_{jkl} = \frac{\mu_{jkl}}{\sigma_{jkl}} \tag{2.13}$$

$$\sigma_{jkl} \rightarrow \tilde{\sigma}_{jkl} = log \sigma_{jkl} \tag{2.14}$$

The derivatives are:

$$\frac{\partial log b_j(o)}{\partial \tilde{\mu}_{jkl}} = c_{jk} N(o; \mu_{jk}, \sigma_{jk}^2)(b_j(o))^{-1} \left(\frac{o_l - \mu_{jkl}}{\sigma_{jkl}} \right), \tag{2.15}$$

and

$$\frac{\partial log b_j(o)}{\partial \tilde{\sigma}_{jkl}} = c_{jk} N(o; \mu_{jk}, \sigma_{jk}^2)(b_j(o))^{-1} \left[\left(\frac{o_l - \mu_{jkl}}{\sigma_{jkl}} \right)^2 - 1 \right]. \tag{2.16}$$

GPD is a first-order gradient method, which is slow in convergence [90]. There is an attempt to use second-order gradient methods (the family of Newton method) for MCE training [83]. The original Newton method for parameter update is [90]:

$$\Lambda_{t+1} = \Lambda_t - (\nabla^2 L)^{-1} \nabla L|_{\Lambda=\Lambda_t}. \tag{2.17}$$

$\nabla^2 L$ is called a Hessian matrix, and is hard to compute and store. All kinds of approximations are made to approximate $\nabla^2 L$ with reasonable computation efforts. One popular method is Quickprop [83], which uses an approximated diagonal Hessian matrix. For a scalar parameter λ (e.g., μ_{jkl}), the approximation is made as:

$$\frac{\partial^2 L}{\partial \lambda^2}|_{\lambda=\lambda_t} \approx \frac{\frac{\partial L}{\partial \lambda}|_{\lambda=\lambda_t} - \frac{\partial L}{\partial \lambda}|_{\lambda=\lambda_{t-1}}}{\lambda_t - \lambda_{t-1}}. \tag{2.18}$$

Since the output in Eq. (2.18) may be negative, Eq. (2.17) is modified to ensure the second order components are positive by adding an positive offset:

$$\Lambda_{t+1} = \Lambda_t - [(\nabla^2 L)^{-1} + \epsilon]\nabla L|_{\Lambda=\Lambda_t}. \tag{2.19}$$

In [83], other second (e.g., Rprop) order optimization methods are also used for MCE training. Compared with GPD and Rprop, Quickprop showed better performance in [83].

2.2.5 Extended Baum-Welch (EBW) Algorithm

EBW is now the most popular optimization methods for discriminative training on LVCSR tasks. It can be applied to MMIE, MCE, and MPE/MWE training. It use an easily-optimized weak-sense auxiliary function to replace the original target function, and optimize this weak-sense auxiliary function. In LVCSR training, there are two lattices, one is called numerator lattice, which is for the correct transcription. The other is denominator lattice, which has all the decoded strings. Forward-backward method is used to get occupancy probabilities for arcs within the lattices. EBW is then used to optimize a function of the separation between the likelihood of the numerator lattice and the likelihood of the denominator lattice [99].

Following [99], the update formulas for the k-th Gaussian component of the j-th state for MMIE are:

$$\mu_{jk} = \frac{\theta_{jk}^{num}(O) - \theta_{jk}^{den}(O) + D_{jk}\mu'_{jk}}{\gamma_{jk}^{num} - \gamma_{jk}^{den} + D_{jk}} \tag{2.20}$$

$$\sigma_{jk}^2 = \frac{\theta_{jk}^{num}(O^2) - \theta_{jk}^{den}(O^2) + D_{jk}(\sigma'^2_{jk} + \mu'^2_{jk})}{\gamma_{jk}^{num} - \gamma_{jk}^{den} + D_{jk}} - \mu_{jk}^2, \tag{2.21}$$

where

$$\gamma_{jk}^{num} = \sum_{q=1}^{Q}\sum_{t=s_q}^{e_q} \gamma_{qjk}^{num}(t)\gamma_q^{num} \tag{2.22}$$

$$\theta_{jk}^{num}(O) = \sum_{q=1}^{Q}\sum_{t=s_q}^{e_q} \gamma_{qjk}^{num}(t)\gamma_q^{num}O(t) \tag{2.23}$$

$$\theta_{jk}^{num}(O^2) = \sum_{q=1}^{Q}\sum_{t=s_q}^{e_q} \gamma_{qjk}^{num}(t)\gamma_q^{num}O(t)^2. \tag{2.24}$$

D_{jk} is a Gaussian dependent constant, decided by a routine described in [99]. $\gamma_{qjk}^{num}(t)$ is the within-arc probability at time t, γ_q^{num} is the occupation probability of arc q in the numerator lattice. s_q and e_q are the starting and ending times of arc q. Similar formulations are for the statistics in the denominator lattice.

EBW can also be applied for MCE [107] and MWE/MPE [99] training. The recent 3.4 version of HTK [125] provides the implementation of MMIE and MWE/MPE using EBW. Chapter 7 will also use EBW for optimization on a LVCSR task.

2.3 Empirical Risk

The purpose of classification and recognition is usually to minimize classification errors on a representative testing set by constructing a classifier f (modeled by the parameter set Λ) based on a set of training samples $(x_1, y_1)...(x_N, y_N) \in X * Y$. X is the observation space, Y is the label space, and N is the number of training samples. We do not know exactly what the property of testing samples is and can only assume that the training and testing samples are independently and identically distributed from some distribution $P(x, y)$. Therefore, we want to minimize the expected classification risk:

$$R(\Lambda) = \int_{X*Y} l(x, y, f_\Lambda(x, y)) dP(x, y). \tag{2.25}$$

$l(x, y, f_\Lambda(x, y))$ is a loss function. There is no explicit knowledge of the underlying distribution $P(x, y)$. It is convenient to assume that there is a density $p(x, y)$ corresponding to the distribution $P(x, y)$ and replace $\int dP(x, y)$ with $\int p(x, y) dxdy$. Then, $p(x, y)$ can be approximated with the empirical density as

$$p_{emp}(x, y) = \frac{1}{N} \sum_{i=1}^{N} \delta(x, x_i) \delta(y, y_i), \tag{2.26}$$

Table 2.1. Discriminative training target function and loss function

	Optimization Object	Loss Function l
MMIE	$max\frac{1}{N}\sum_{i=1}^{N} log\frac{P_\Lambda(O_i \mid S_i)P(S_i)}{\sum_{\hat{S}_i} P_\Lambda(O_i \mid \hat{S}_i)P(\hat{S}_i)}$	$1 - log\frac{P_\Lambda(O_i \mid S_i)P(S_i)}{\sum_{\hat{S}_i} P_\Lambda(O_i \mid \hat{S}_i)P(\hat{S}_i)}$
MCE	$min\frac{1}{N}\sum_{i=1}^{N} \frac{1}{1+exp(-\gamma d(O_i,\Lambda)+\theta)}$	$\frac{1}{1+exp(-\gamma d(O_i,\Lambda)+\theta)}$
MPE	$max\frac{1}{N}\sum_{i=1}^{N} \frac{\sum_{\hat{S}_i} P_\Lambda(O_i \mid \hat{S}_i)P(\hat{S}_i)RawPhoneAccuracy(\hat{S}_i)}{\sum_{\hat{S}_i} P_\Lambda(O_i \mid \hat{S}_i)P(\hat{S}_i)}$	$1 - \frac{\sum_{\hat{S}_i} P_\Lambda(O_i \mid \hat{S}_i)P(\hat{S}_i)RawPhoneAccuracy(\hat{S}_i)}{\sum_{\hat{S}_i} P_\Lambda(O_i \mid \hat{S}_i)P(\hat{S}_i)}$

where $\delta(x, x_i)$ is the Kronecker delta function. Finally, the empirical risk is minimized instead of the intractable expected risk:

$$\begin{aligned} R_{emp}(\Lambda) &= \int_{X*Y} l(x, y, f_\Lambda(x, y))p_{emp}(x, y)dxdy \\ &= \frac{1}{N}\sum_{i=1}^{N} l(x_i, y_i, f_\Lambda(x_i, y_i)). \end{aligned} \tag{2.27}$$

Most learning methods focus on how to minimize this empirical risk. However, as shown above, the empirical risk approximates the expected risk by replacing the underlying density with its corresponding empirical density. Simply minimizing the empirical risk does not necessarily minimize the expected test risk.

In the application of speech recognition, most discriminative training (DT) methods directly minimize the risk on the training set, i.e., the empirical risk, which is defined as

$$R_{emp}(\Lambda) = \frac{1}{N}\sum_{i=1}^{N} l(O_i, \Lambda), \tag{2.28}$$

where $l(O_i, \Lambda)$ is a loss function for utterance O_i, and N is the total number of training utterances. $\Lambda = (\pi, a, b)$ is a parameter set denoting the set of initial state probability, state transition probability, and observation distribution. Table 2.1 lists the optimization objectives and loss functions of MMIE, MCE, and MPE. With these loss functions, these DT methods all attempt to minimize some empirical risks.

2.4 Test Risk Bound

The optimal performance of the training set does not guarantee the optimal performance of the testing set. This stems from statistical learning theory [119], which states that with at least a probability of $1-\delta$ (δ is a small positive number) the risk on the testing set (i.e., the test risk) is bounded as follows:

$$R(\Lambda) \leq R_{emp}(\Lambda) + \sqrt{\frac{1}{N}(VC_{dim}(log(\frac{2N}{VC_{dim}}) + 1) - log(\frac{\delta}{4})}). \quad (2.29)$$

VC_{dim} is the *VC dimension* that characterizes the complexity of a classifier function group G, and means that at least one set of VC_{dim} (or less) number of samples can be found such that G shatters them. Equation (2.29) shows that the test risk is bounded by the summation of two terms. The first is the empirical risk, and the second is a generalization (regularization) term that is a function of the VC dimension. Although the risk bound is not strictly tight [11], it still gives us insight to explain current technologies in ASR:

- The use of more data: In current large scale large vocabulary continuous speech recognition (LVCSR) tasks, thousands of hours of data may be used to get better performance. This is a simple but effective method. When the amount of data is increased, the empirical risk is usually not changed, but the generalization term decreases as the result of increasing N.

- The use of more parameters:With more parameters, the training data will be fit better with reduced empirical risk. However, the generalization term increases at the same time as a result of increasing VC_{dim}. This is because with more parameters, the classification function is more complex and has the ability to shatter more training points. Hence, by using more parameters, there is a potential danger of over-fitting when the empirical error does not drop while the generalization term keeps increasing

- Most DT methods: DT methods, such as MMIE, MCE, and MWE/MPE in Table 2.1, focus on reducing the empirical risks and do not consider decreasing the generalization term in Eq. (2.29) from the perspective of statistical learning theory. However, these DT methods have other strategies to deal with the problem of over-training. "I-smoothing" [101], used in MMIE and MWE/MPE, makes an interpolation between the objective functions of MLE and the discriminative methods. The sigmoid function of MCE can be interpreted as the integral of a Parzen kernel [85], helping MCE for regularization. Parzen estimation has the attractive property that it converges when the number of training sample grows to infinity. In contrast, margin-based methods reduce the test risk from the viewpoint of statistical learning theory with the help of Eq. (2.29).

2.5 Margin-based Methods in Automatic Speech Recognition

Inspired by the great success of margin-based classifiers, there is a trend toward incorporating the margin concept into hidden Markov modeling for speech recognition. Several attempts based on margin maximization were proposed recently. There are three major methods. The first method is large margin estimation, proposed by ASR researchers at York university [45], [66], [73], [74]. The second algorithm is large margin hidden Markov models, proposed by computer science researchers at the University of Pennsylvania [111], [112]. The third one is soft margin estimation (SME), proposed by us [65], [66], [68]. Another attempt is to consider the offset in the sigmoid function as a margin to extend minimum classification error (MCE) training of HMMs [126]. In this section, the first two algorithms are introduced. Our proposed method, SME, is discussed in the preliminary research section.

2.5.1 Large Margin Estimation

Motivated by large margin classifiers in machine learning, large margin estimation (LME) is proposed as the first ASR method that strictly uses the spirit of margin and has achieved success on the TIDIGITS task [61]. In this section, LME is briefly introduced.

For a speech utterance O_i, LME defines the multi-class separation margin for O_i:

$$d(O_i, \Lambda) = p_\Lambda(O_i|S_i) - p_\Lambda(O_i|\hat{S}_i). \tag{2.30}$$

For all utterances in a training set D, LME defines a subset of utterances S as:

$$S = \{O_i | O_i \in D \text{ and } 0 \leq d(O_i, \Lambda) \leq \epsilon\}, \tag{2.31}$$

where $\epsilon > 0$ is a preset positive number. S is called a support vector set and each utterance in S is called a support token.

LME estimates the HMM models Λ based on the criterion of maximizing the minimum margin of all support tokens as:

$$\tilde{\Lambda} = arg \max_{\Lambda} \min_{O_i \in S} d(O_i, \Lambda). \tag{2.32}$$

With Eq. (2.30), large margin HMMs can be equivalently estimated as follows:

$$\tilde{\Lambda} = arg \max_{\Lambda} \min_{O_i \in S} \left[p_{\Lambda}(O_i | S_i) - p_{\Lambda}(O_i | \hat{S}_i) \right] \tag{2.33}$$

The research focus of LME is to use different optimization methods to solve LME. In [73], generalized gradient descent (GPD) [50] is used to estimate Λ. GPD is widely used in discriminative training, especially in MCE. Constrained joint optimization is applied to LME for the estimation of Λ in [71]. By making some approximations, the target function of LME is converted to be convex and Λ is got with semi-definite programming (SDP) [7] in [74], [72]. Second order cone programming [124] is used to improve the training speed of LME.

One potential weakness of LME is that it updates models only with accurately classified samples. However, it is well known that misclassified samples are also critical for classifier learning. The support set of LME neglects the misclassified samples, e.g., samples 1, 2, 3, and 4 in Figure 2.2. In this case, the margin obtained by LME is hard to justify as a real margin for generalization. Consequently, LME often needs a very good preliminary estimate from the training set to make the influence of ignoring misclassified samples small. Hence, LME usually uses MCE models as its initial model. The above-mentioned problem has been addressed in [44].

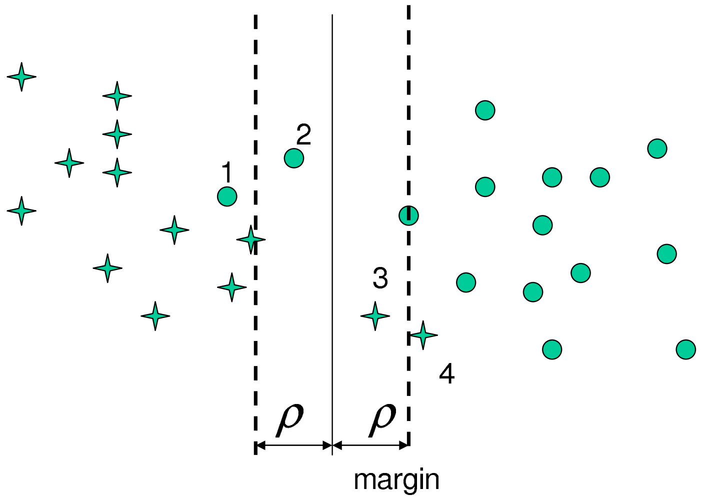

Figure 2.2. Large margin estimation.

2.5.2 Large Margin Gaussian Mixture Model and Hidden Markov Model

Large margin GMMs (LM-GMMs) are very similar to SVMs, using ellipsoids to model classes instead of using half-spaces. For the simplest case, every class is modeled by a Gaussian. Let $(\mu_c, \Psi_c, \theta_c)$ denote the mean, precision matrix, and scalar offset for class c. For any sample x, the classification decision is made by choosing the class that has the minimum Mahalanobis distance [21] :

$$y = arg \min_c \{(x - \mu_c)^T \Psi_c (x - \mu_c) + \theta_c\}. \tag{2.34}$$

LM-GMMs collects the parameters of each class in an enlarged positive semi-definite matrix:

$$\Phi_c = \begin{bmatrix} \Psi_c & -\Psi_c \mu_c \\ -\mu_c{}^T \Psi_c & \mu_c{}^T \Psi_c \mu_c + \theta_c \end{bmatrix}. \tag{2.35}$$

Eq. (2.34) can be re-written as:

$$y = arg \min_c \{z^T \Phi_c z\}, \tag{2.36}$$

where

$$z = \begin{bmatrix} x \\ 1 \end{bmatrix}. \tag{2.37}$$

Parallel to the separable case in SVMs, for the n-th sample with label y_n, LM-GMMs have the following formulation:

$$\forall c \neq y_n, \qquad {z_n}^T \Phi_c z_n \geq 1 + {z_n}^T \Phi_{y_n} z_n \tag{2.38}$$

For the inseparable case, a hinge loss function ($(f)_+ = max(0, f)$) is used to get the empirical loss function for large margin Gaussian modeling. By regularizing with the sum of all traces of precision matrices, the final target function for large margin Gaussian modeling is:

$$L = \lambda \sum_n \sum_{c \neq y_n} [1 + {z_n}^T (\Phi_{y_n} - \Phi_c) z_n]_+ + \sum_c trace(\Psi_c). \tag{2.39}$$

All the model parameters are optimized by minimizing Eq. (2.39). Approximations are made to apply the Gaussian target function in Eq. (2.39). to the case of GMMs [111] and HMMs [112]. Remarkable performance has been achieved on the TIMIT database [28].

In [111], GMMs are used for ASR tasks instead of HMMs. This work is not consistent with the HMMs structure in ASR. The work of [111] was extended to deal with HMMs in [112] by summing the differences of Mahalanobis distances between the models in the correct and competing strings and comparing the result with a Hamming distance. It is not clear whether it is suitable to directly compare the Hamming distance with the difference of Mahalanobis distances. The kernel of LM-GMMs or LM-HMMs is to use the minimum trace for generalization. It is hard to know whether the trace is a good indicator for the generalization of HMMs.

It should be noted that the approximation made for convex optimization sacrifices precision in some extent. The author of LM-HMMs found that if he could not get the exact phoneme boundary information from the TIMIT database, no performance improvement was got if the boundary was determined by force alignment [110]. He doubted it is because of the approximation made for convex target function. In most ASR tasks, it is impossible to get the exact phoneme boundary as the case in the TIMIT database. No report from LM-HMMs on other ASR tasks rather than TIMIT.

CHAPTER 3

SOFT MARGIN ESTIMATION (SME)

In this chapter, soft margin estimation (SME) [68] is proposed as a link between statistical learning theory and ASR. We provide a theoretical perspective about SME, showing that SME relates to an approximate test risk bound. The idea behind the choice of the loss function for SME is then illustrated and the separation functions are defined. Discriminative training (DT) algorithms, such as MMIE, MCE, and MWE/MPE, can also be cast in the rigorous SME framework by defining corresponding separation functions. Two solutions to SME are provided and the difference with other margin-based methods is discussed. SME is test on two different tasks. One is the spoken language recognition task, and the other is a connected-digit recognition task.

3.1 Approximate Test Risk Bound Minimization

Let's revisit the risk bound in statistical learning theory. The bound of the test risk is:

$$R(\Lambda) \leq R_{emp}(\Lambda) + \sqrt{\frac{1}{N}(VC_{dim}(log(\frac{2N}{VC_{dim}}) + 1) - log(\frac{\delta}{4})}). \tag{3.1}$$

If the right hand side of Eq. (3.1) can be directly minimized, it is possible to minimize the test risk. However, as a monotonic increasing function of VC_{dim}, the generalization term cannot be directly minimized because of the difficulty of computing VC_{dim}. It can be shown that VC_{dim} is bounded by a decreasing function of the margin [119]. Hence, VC_{dim} can be reduced by increasing the margin. Now, there are two targets for optimization: one is to minimize the empirical risk, and the other is to maximize the margin. Because the test risk bound of Eq. (3.1) is not tight, it is not necessary to strictly follow Vapnik's theorem. Instead, the test risk bound can be approximated by combining two optimization targets into a single SME objective function:

$$L^{SME}(\rho, \Lambda) = \frac{\lambda}{\rho} + R_{emp}(\Lambda) = \frac{\lambda}{\rho} + \frac{1}{N}\sum_{i=1}^{N} l(O_i, \Lambda), \tag{3.2}$$

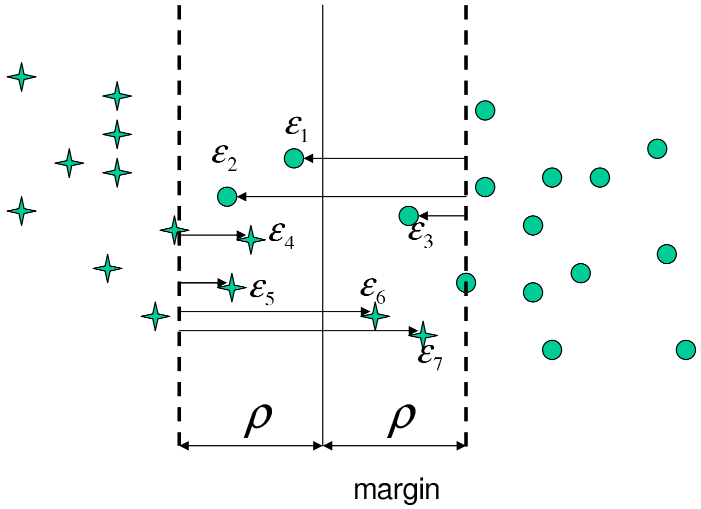

Figure 3.1. Soft margin estimation.

where ρ is the soft margin, and λ is a coefficient to balance the soft margin maximization and the empirical risk minimization. A smaller λ corresponds to a higher penalty for the empirical risk. The soft margin usage originates from the soft margin SVMs, which deal with non-separable classification problems. For separable cases, margin is defined as the minimum distance between the decision boundary and the samples nearest to it. As shown in Figure 3.1, the soft margin for the non-separable case can be considered as the distance between the decision boundary (solid line) and the class boundary (dotted line). The class boundary has the same definition as for the separable case after removing the tokens near the decision boundary and treating these tokens differently using slack variable ϵ_i ($l(O_i, \Lambda)$) in Figure 3.1. The approximate test risk is minimized by minimizing Eq. (3.2).

This view distinguishes SME from both ordinary DT methods and LME. Ordinary DT methods only minimize the empirical risk $R_{emp}(\Lambda)$ with additional generalization tactics. LME only reduces the generalization term by minimizing λ/ρ in Eq. (3.2), and its margin ρ is defined on correctly classified samples.

It should be noted that there is no exact margin for the inseparable classification task, since different balance coefficients λ will result in different margin values. The study here is to bridge the research in machine learning and the research in ASR, and tries to investigate whether embedded the margin into the object function will boost ASR system performance.

3.2 Loss Function Definition

The next issue is to define the loss function $l(O_i, \Lambda)$ for Eq. (3.2). As shown in Eq. (3.2), the essence of the margin-based method is to use a margin to secure some generalization in classifier learning. If the mismatch between the training and testing causes a shift less than this margin, a correct decision can still be made. So, a loss occurs only when $d(O_i, \Lambda)$ is less than the value of the soft margin. It should be emphasized that the loss here is not the recognition error. A recognition error occurs when $d(O_i, \Lambda)$ is less than 0. Therefore, the loss function can be defined with the help of a hinge loss function ($(x)_+ = max(x, 0)$):

$$\begin{aligned} l(O_i, \Lambda) &= (\rho - d(O_i, \Lambda))_+ \\ &= \begin{cases} \rho - d(O_i, \Lambda), & \text{if } \rho - d(O_i, \Lambda) > 0 \\ 0, & otherwise \end{cases} . \end{aligned} \tag{3.3}$$

The SME objective function can be rewritten as

$$\begin{aligned} L^{SME}(\rho, \Lambda) &= \frac{\lambda}{\rho} + \frac{1}{N}\sum_{i=1}^{N}(\rho - d(O_i, \Lambda))_+ \\ &= \frac{\lambda}{\rho} + \frac{1}{N}\sum_{i=1}^{N}(\rho - d(O_i, \Lambda))I(O_i \in U), \end{aligned} \tag{3.4}$$

where I is an indicator function, and U is the set of utterances that have the separation measures less than the soft margin.

3.3 Separation Measure Definition

The third step is to define a separation (misclassification) measure, $d(O_i, \Lambda)$, which is a distance between the correct and competing hypotheses. A common choice is to use a log likelihood ratio (LLR), as in MCE [48] and LME [73]:

$$d^{LLR}(O_i, \Lambda) = log\left[\frac{P_\Lambda(O_i|S_i)}{P_\Lambda(O_i|\hat{S}_i)}\right]. \tag{3.5}$$

If $d^{LLR}(O_i, \Lambda)$ is greater than 0, the classification is correct. Otherwise, a wrong decision is obtained. $P_\Lambda(O_i|S_i)$ and $P_\Lambda(O_i|\hat{S}_i)$ are the likelihood scores for the target and the most competitive strings. In the following, a more precise model separation measure is defined. For every utterance, we select the frames that have different HMM model labels in the target and competitive strings. These frames can provide discriminative information. The model separation measure for a given utterance is defined as the average of those frame LLRs. n_i is used to denote this number of different frames for utterance O_i. Then, the separation of the models is defined as

$$d^{SME\_utter}(O_i, \Lambda) = \frac{1}{n_i} \sum_j log \left[\frac{P_\Lambda(O_{ij}|S_i)}{P_\Lambda(O_{ij}|\hat{S}_i)} \right] I(O_{ij} \in F_i), \tag{3.6}$$

where F_i is the frame set in which the frames have different labels in the competing strings. O_{ij} is the jth frame for utterance O_i. Only the most competitive string is used in the definition of Eq. (3.6).

Our separation measure definition is different from LME or MCE, in which the utterance LLR is used. For the usage in SME, the normalized LLR may be more discriminative because the utterance length and the number of different models in the competing strings affect the overall utterance LLR value. For example, it may not be appropriate that an utterance consisting of five different units in the target and competitive strings has greater separation for models inside it than another utterance with only one different unit because the former has a larger LLR value.

By plugging the quantity in Eq. (3.6) into Eq. (3.4), the optimization function of SME becomes:

$$L^{SME}(\rho, \Lambda) = \frac{\lambda}{\rho} + \frac{1}{N} \sum_{i=1}^{N} \left(\rho - \frac{1}{n_i} \sum_j log \left[\frac{P_\Lambda(O_{ij}|S_i)}{P_\Lambda(O_{ij}|\hat{S}_i)} \right] I(O_{ij} \in F_i) \right) I(O_i \in U). \tag{3.7}$$

As shown in Eq. (3.7), frame selection (by $I(O_{ij} \in F_i)$), utterance selection (by $I(O_i \in U)$), and discriminative separation are unified in a single objective function. This quantity provides a flexible framework for future studies. For example, for frame selection, F_i can be defined as a subset with frames more critical for discriminating HMM models, instead of equally choosing distinct frames in current study. This will be discussed in detail in Chapter 7.

We can also define separations corresponding to MMIE, MCE, and MPE, as shown in Table 3.1. These separations will be studied in the future. All these measures can be put back into Eq. (3.4) for HMM parameter estimation.

Table 3.1. Separation measure for SME

$d^{SME\_utter}(O_i, \Lambda)$	$\frac{1}{n_i} \sum_j log \left[\frac{P_\Lambda(O_{ij} \mid S_i)}{P_\Lambda(O_{ij} \mid \hat{S}_i)} \right] I(O_{ij} \in F_i)$
$d^{SME\_MMIE}(O_i, \Lambda)$	$log \frac{P_\Lambda(O_i \mid S_i) P(S_i)}{\sum_{\hat{S}_i} P_\Lambda(O_i \mid \hat{S}_i) P(\hat{S}_i)}$
$d^{SME\_MCE}(O_i, \Lambda)$	$1 - \frac{1}{1+exp(-\gamma d(O_i,\Lambda)+\theta)}$
$d^{SME\_MPE}(O_i, \Lambda)$	$\frac{\sum_{\hat{S}_i} P_\Lambda(O_i \mid \hat{S}_i) P(\hat{S}_i) RawPhoneAccuracy(\hat{S}_i)}{\sum_{\hat{S}_i} P_\Lambda(O_i \mid \hat{S}_i) P(\hat{S}_i)}$

3.4 Solutions to SME

In this section, two solutions to SME are proposed. One solution is to optimize the soft margin and the HMM parameters jointly. The other is to set the soft margin in advance and then find the optimal HMM parameters.

1) Jointly optimize the soft margin and the HMM parameters: In this solution, the indicator function $I(O_i \in U)$ in Eq. (3.4) is approximated with a sigmoid function. Then Eq. (3.4) becomes

$$L^{SME}(\rho, \Lambda) = \frac{\lambda}{\rho} + \frac{1}{N} \sum_{i=1}^{N} (\rho - d(O_i, \Lambda)) \frac{1}{1 + exp(-\gamma(\rho - d(O_i, \Lambda)))}, \qquad (3.8)$$

where γ is a smoothing parameter for the sigmoid function. Equation (3.8) is a smoothing function of the soft margin ρ and the HMM parameters Λ. Therefore, these parameters can be optimized by iteratively using the GPD algorithm on the training set as in [50], with η_t and κ_t as step sizes for iteration t:

$$\begin{cases} \Lambda_{t+1} = \Lambda_t - \eta_t \nabla L^{SME}(\rho, \Lambda)|_{\Lambda=\Lambda_t} \\ \rho_{t+1} = \rho_t - \kappa_t \nabla L^{SME}(\rho, \Lambda)|_{\rho=\rho_t} \end{cases} \qquad (3.9)$$

We need to preset the coefficient λ, which balances the soft margin maximization and the empirical risk minimization.

2) Presetting the soft margin and optimizing the HMM parameters: For a fixed λ, there is one corresponding ρ as the final solution. Instead of choosing a fixed λ and trying to get the solution of (ρ, Λ) as in the first solution, we can directly choose a ρ in advance. There is no explicit knowledge of what λ should be, so it is not necessary to start from λ and get the exact corresponding solution of ρ. Instead, we will show in the section of experiments that it is easy to draw some knowledge of the range of ρ. Setting ρ in advance is a simple way to solve the SME problem.

Because of a fixed ρ, only the samples with separation smaller than the margin need to be considered. Assuming that there are a total of N_c utterances satisfying this condition, we can minimize the following with the constraint $d(O_i, \Lambda) < \rho$:

$$L^{sub}(\Lambda) = \sum_{i=1}^{N_c} (\rho - d(O_i, \Lambda)). \tag{3.10}$$

Now, this problem can be solved by the GPD algorithm by iteratively working on the training set, with η_t as a step size for iteration t:

$$\Lambda_{t+1} = \Lambda_t - \eta_t \nabla L^{sub}(\Lambda)|_{\Lambda=\Lambda_t}. \tag{3.11}$$

3.4.1 Derivative Computation

The derivatives of SME objective functions with respect to (w.r.t.) Λ and ρ are the key to implement the GPD algorithm (Eq. (3.9) or Eq. (3.11)). In the following, we give the deduction of those derivatives using Eq. (3.8) as the objective function.

For the derivative w.r.t. model parameters Λ, we have the following equations.

$$\begin{aligned} \frac{\partial L^{SME}(\rho, \Lambda)}{\partial \Lambda} &= \frac{1}{N} \sum_{i=1}^{N} \frac{\partial (\rho - d(O_i, \Lambda)) 1/[1 + exp(-\gamma(\rho - d(O_i, \Lambda)))]}{\partial \Lambda} \\ &= \frac{1}{N} \sum_{i=1}^{N} \{A + B\}, \end{aligned} \tag{3.12}$$

where

$$A = \frac{\partial (\rho - d(O_i, \Lambda))}{\partial \Lambda} \frac{1}{1 + exp(-\gamma(\rho - d(O_i, \Lambda)))}, \tag{3.13}$$

and

$$B = (\rho - d(O_i, \Lambda))\frac{\partial 1/[1 + exp(-\gamma(\rho - d(O_i, \Lambda)))]}{\partial \Lambda}. \tag{3.14}$$

The two derivatives in Eq. (3.13) and Eq. (3.14) can be further written as:

$$\frac{\partial(\rho - d(O_i, \Lambda))}{\partial \Lambda} = \frac{\partial(-d(O_i, \Lambda))}{\partial \Lambda}, \tag{3.15}$$

and

$$\begin{aligned} &\tfrac{\partial 1/[1+exp(-\gamma(\rho-d(O_i,\Lambda)))]}{\partial \Lambda} \\ &= -\left\{\tfrac{1}{1+exp(-\gamma(\rho-d(O_i,\Lambda)))}\right\}^2 exp[-\gamma(\rho - d(O_i, \Lambda))](-\gamma)\tfrac{\partial(\rho-d(O_i,\Lambda))}{\partial \Lambda} \\ &= \gamma\left\{1 - \tfrac{1}{1+exp(-\gamma(\rho-d(O_i,\Lambda)))}\right\}\tfrac{1}{1+exp(-\gamma(\rho-d(O_i,\Lambda)))}\tfrac{\partial(\rho-d(O_i,\Lambda))}{\partial \Lambda}. \end{aligned} \tag{3.16}$$

Putting above two equations together, we get

$$\begin{aligned} &\tfrac{\partial L^{SME}(\rho,\Lambda)}{\partial \Lambda} \\ &= \tfrac{1}{N}\sum_{i=1}^{N}\left\{\tfrac{\partial(\rho-d(O_i,\Lambda))}{\partial \Lambda}\tfrac{1}{1+exp(-\gamma(\rho-d(O_i,\Lambda)))}(1 + \gamma(\rho - d(O_i, \Lambda)))\left\{1 - \tfrac{1}{1+exp(-\gamma(\rho-d(O_i,\Lambda)))}\right\}\right\}. \end{aligned} \tag{3.17}$$

Since $d(O_i, \Lambda)$ is a normalized LLR, its derivative w.r.t. Λ can be computed similarly to what has been done in MCE training. Please refer [48] and Eqs. (2.15), (2.16) for the detailed formulations of those derivatives.

For the derivative w.r.t. ρ, we have

$$\begin{aligned} &\partial L^{SME}(\rho, \Lambda)/\partial\rho \\ &= -\tfrac{\lambda}{\rho^2} + \tfrac{1}{N}\sum_{i=1}^{N}\left\{\begin{array}{c} \tfrac{\partial(\rho-d(O_i,\Lambda))}{\partial\rho}\tfrac{1}{1+exp(-\gamma(\rho-d(O_i,\Lambda)))} \\ +(\rho - d(O_i, \Lambda))\tfrac{\partial\frac{1}{1+exp(-\gamma(\rho-d(O_i,\Lambda)))}}{\partial\rho} \end{array}\right\} \\ &= -\tfrac{\lambda}{\rho^2} + \tfrac{1}{N}\sum_{i=1}^{N}\left\{\begin{array}{c} \tfrac{1}{1+exp(-\gamma(\rho-d(O_i,\Lambda)))} \\ +\gamma\tfrac{1}{1+exp(-\gamma(\rho-d(O_i,\Lambda)))}(1 - \tfrac{1}{1+exp(-\gamma(\rho-d(O_i,\Lambda)))})(\rho - d(O_i, \Lambda)) \end{array}\right\} \end{aligned} \tag{3.18}$$

3.5 Margin-Based Methods Comparison

In this section, SME is compared with two margin-based method groups. One group is LME [45], [73], [74], and the other is large margin GMM (LM-GMM) [111] and large margin HMM (LM-HMM) [112]. LM-HMM and LM-GMM are very similar, except that

LM-HMM measures model distance in a whole utterance, while LM-GMM measures in a segment. The differences of these margin-based methods are listed in Table 3.2 and are discussed in the following.

Table 3.2. Comparison of margin-based methods

	LME	LM-GMM [111], LM-HMM [112]	SME
Training samples	correctly classified samples	all samples	all samples
Separation measure	utterance LLR	Mahalanobis distance	LLR with frame selection
Segmental modeling	HMM	GMM [111], HMM [112]	HMM
Target function	margin maximization	penalized trace minimization	penalized margin maximization
Convex problem	No [45], [73] Yes [74]	Yes	No

- **Training sample usage**: Both LM-GMM/LM-HMM and SME use all the training samples, while LME only uses correctly classified samples. The misclassified samples are important for classifier learning because they carry the information to discriminate models. Except for LME, discriminative training methods usually use all the training samples.

- **Separation measure**: It is crucial to define a good separation measure because it directly relates to margin. LME uses utterance-based LLR as a measure while in SME it is carefully represented by a normalized LLR measure over only the set of different frames. With such normalization, the utterance separation values can be more closely compared with a fixed margin than an un-normalized LLR without being affected by different numbers of distinct units and length of the utterances. LM-GMM and LM-HMM use Mahalanobis distance [21], which makes it hard to be directly used in the context of mixture models. In [111] and [112], an approximation to the mixture component with the highest posterior probability under GMM is applied.

- **Segmental training**: Speech is segment based. Both SME and LME use HMMs, while LM-GMM uses frame-averaged GMM to approximate segmental training. As an improvement, LM-HMM directly works on the whole utterance. It sums the difference of the Mahalanobis distances between the models in the correct and competing strings and compares it with a Hamming distance. That Hamming distance is the number of mismatched labels of recognized string. Although similar distance (raw phone accuracy) has been used in MPE [101] for weighting the contribution from different recognized strings, it is not clear whether Hamming distance is suitable to be directly used to compare with the Mahalanobis distance because these two distances are very different types of measures (one is for string labels and the other is for Gaussian models).

- **Target function**: SME maximizes the soft margin penalized with the empirical risk as in Eq. (3.2). This objective directly relates to the test risk bound shown in Eq. (3.1). LME only maximizes its margin, assuming the empirical risk is 0. The idea of LME is to define the minimum positive separation distance as a margin and then maximize it. Because of this, the technology dealing with misclassified samples by making use of a soft margin or slack variable cannot be easily incorporated in LME. LM-GMM/LM-HMM minimizes the summation of all the traces of Gaussian models, penalized with a Mahalanobis-distance-based misclassification measure.

- **Convex problem**: LME has several different solutions. In [45],[73], the target function is non-convex. By using a series of transformations and constraints [74], LME can have a convex target function. Also, LM-GMM and LM-HMM formularize their target function as a convex one. The convex function has the nice property that its local minimum is a global minimum. This will make the parameter optimization much easier. To get a convex target function, it needs to approximate the GMM with a single mixture component of the GMM. It should be noted that the approximation made for convex optimization sacrifices precision in some extent. The author of LM-HMM found that if he could not get the exact phoneme boundary information from the TIMIT database, no performance improvement was got if the boundary was determined by force alignment [110]. He doubted it is because of the approximation made for convex target function. The target function of SME is not convex. Therefore, SME is subject to local minima like most other DT methods. In the future, we will investigate whether SME can also get a convex target function with the cost of approximation and some transformations.

3.6 Experiments

In this section, SME is evaluated on two tasks. The first is a spoken language recognition task, with GMM as the underlying model. The other is a connected-digit recognition task, with HMM as the underlying model. On both tasks, SME demonstrated its superiority over MCE.

3.6.1 SME with Gaussian Mixture Model

SME is designed for ASR applications. In the case of designing ASR systems, the classifier parameters are often related to defining a set of HMMs, one for each of the fundamental speech units. To show SME is a generalized machine learning method, we should not constrain SME in ASR applications. Many applications in the speech research area use models rather than HMMs. For example, GMMs are widely used in language identification [114] and speaker verification [106].

The same framework of SME will be applied to GMMs for the application of language identification (LID) in the following. NIST (National Institute of Standards and Technology) has coordinated evaluations of automatic language recognition technologies in 1996, 2003 and, recently, in 2005 to promote spoken language recognition research. Several techniques have achieved recent successes. The most popular framework is parallel phone recognition followed by language model (P-PRLM) [128]. It uses multiple sets of phone models to decode spoken utterances into phone sequences, and builds one set of phone language model (LM) for each P-PRLM tokenizer-target language pair. The P-PRLM scores are computed by combining acoustic and language scores and the language with the maximum combination score is determined to be the recognized language. Another recently proposed approach is to use bag-of-sounds (BOS) models of phone-like units, such as acoustic segment units [80], to convert utterances into text-like documents. Then vector-based techniques, such as GMM and support vector machine (SVM), can easily be adopted for language recognition [80],[114],[13].

In [70], we have presented a language recognition system designed for 2005 NIST LRE. Instead of using the scores computed from P-PRLM and BOS systems directly to make language recognition decisions, we used the scores from them for all competing languages to serve as input features to train the linear discriminative function (LDF) and artificial neural network (ANN) verifiers, and fuse the output verification scores to make final decisions. Both the LDF and ANN classifiers can be obtained with discriminative training. For the LDF verifier with a small number of parameters we achieved a comparable performance with that of the ANN verifier, which is much more complex than the LDF verifier. We have also shown that the distribution of confidence scores from the ANN and LDF verifiers exhibited large diversity, which is ideal for score fusion. Experiments have demonstrated the fused system achieved a better performance than systems based on the individual LDF and ANN classifies. However, the performance for that system is not desirable. For the 30-second test set, we only get 13% equal error rate (EER).

In [63], a method called vector space modeling (VSM) output coding is proposed to form the feature vector for back-end processing. Here, we directly use 110-dimension vector generated by that method as the input feature for every utterance, and use GMM to train the classifiers for 2003 and 2005 NIST 30-second evaluation set. We obtained models for 15 languages/dialects in the training stage. They are Arabic, Farsi, French, German,

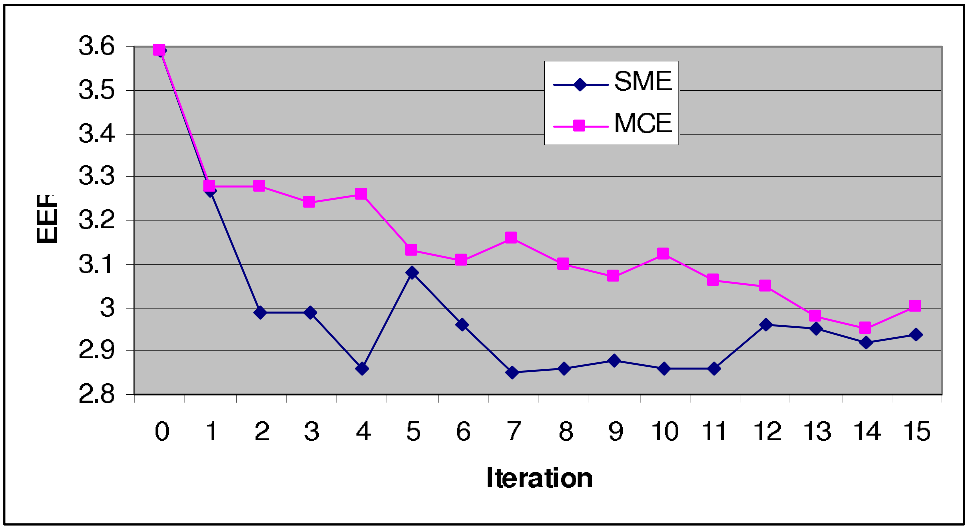

Figure 3.2. EER evolutions for the NIST 03 30-second test set.

Hindi, Japanese, Korean, Tamil, Vietnamese, 2 English dialects, 2 Mandarin dialects and 2 Spanish dialects. The training database consists of the 15 languages/dialects from the Call-Friend corpus [75]. The NIST 2003 test set has 12 target languages (as listed above) and one out-of-target (OOT) language . The NIST 2005 test set consists of 7 target languages (English, Hindi, Japanese, Korean, Mandarin, Spanish, and Tamil) and one OOT language.

In the training stage, there are two GMMs for each language or dialect: a target GMM with 32 Gaussian components and a filler GMM with 256 Gaussian components. The baseline GMM models (got from the researchers in Institute for Infocomm Research) were trained with MLE. The EER for the NIST 2003 30-second test set is 3.59% while the EER for the NIST 2005 30-second evaluation set is 5.46%.

Two discriminative training (DT) methods were applied. One uses SME, and the other uses MCE. These two DT methods share most implementations, differing only in individual algorithm parts. Figures 3.2 and 3.3 show the EER evolutions for NIST 03 and 05 30-second test sets. Almost in every iteration, SME performs better than MCE.

3.6.2 SME with Hidden Markov Model

The proposed SME framework was evaluated on the TIDIGITS [61] connected-digit task. For the TIDIGITS database [61], there are 8623 digit strings in the training set and 8700

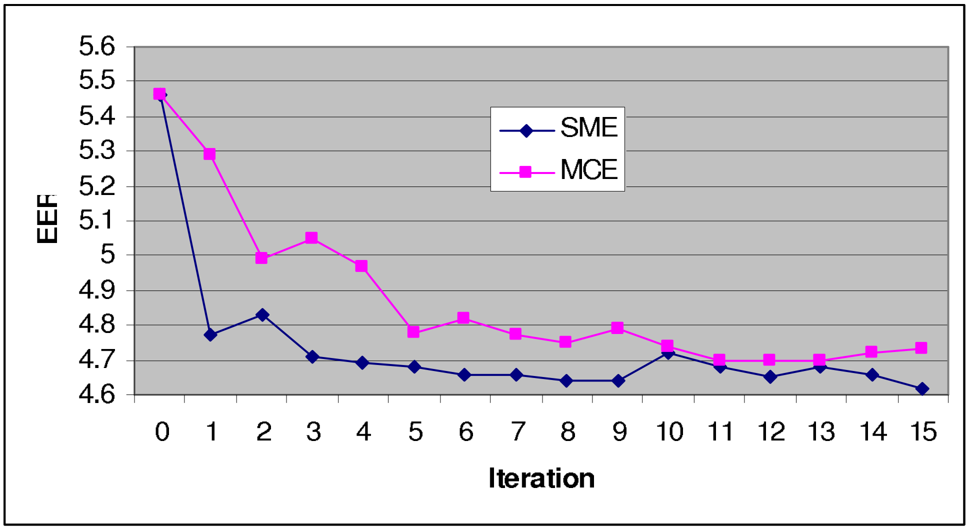

Figure 3.3. EER evolutions for the NIST 05 30-second test set.

digit strings for testing. The hidden Markov model toolkit (HTK) [125] was first used to build the baseline MLE HMMs. There were 11 whole-digit HMMs: one for each of the 10 English digits, plus the word “oh”. Each HMM has 12 states and each state observation density is characterized by a mixture Gaussian density. GMM Models with 1, 2, 4, 8, and 16 mixture components were trained. The input features were 12 Mel-frequency cepstrum coefficients (MFCCs) [16] + energy, and their first and second order time derivatives. MCE models were also trained for comparison. N-best incorrect strings were used for training. The performance of this choice was better than the implementation with the top incorrect string. Different smoothing parameters were tried and the reported results were for the best one. SME models were initiated with the MLE models. This is in clear contrast with the LME models [73], [45] and [74], which are typically built upon the well-performed MCE models. Digit decoding was based on unknown length without imposing any language model or insertion penalty.

$d^{SME\_utter}(O_i, \Lambda)$ was used as the separation measure, which means that only the most competitive string was used in SME training. Two different solutions of SME are compared in this study. The column labeled SME in Table 3.3 presets the soft margin values. Various soft margin values were set corresponding to different model complexities as shown in

Table 3.3. SME: Testing set string accuracy comparison with different methods. Accuracies marked with an asterisk are significantly different from the accuracy of the SME model (p<0.025, paired Z-test, 8700 d.o.f. [60]).

	MLE	MCE	LME	SME	SME_joint
1-mix	95.20%*	96.94%*	96.23%*	98.76%	98.74%
2-mix	96.20%*	97.40%*	98.30%*	98.95%	98.92%
4-mix	97.80%*	98.24%*	98.76%*	99.20%	99.11%
8-mix	98.03%*	98.66%*	99.13%	99.29%	99.26%
16-mix	98.36%*	98.87%*	99.18%	99.30%	99.32%

Table 3.4. Margin value assignment.

1-mix	2-mix	4-mix	8-mix	16-mix
5	6	7.5	8.5	9

Table 3.4. These soft margin values were empirically chosen as the mode of all the separation distances obtained from the MLE model. For example, in Figure 3.5, the mode of the separation distance of the 1-mixture MLE model is about 5. Therefore, the soft margin value for the 1-mixture SME model was set as 5. Slightly changing values in Table 3.4 only made very little difference in the final results. While this setting produced satisfactory results, we believe it is too heuristic and suboptimal, and we will investigate in future work whether any plausible theory underlies it.

The column labeled SME_joint in Table 3.3 solves SME by optimizing the soft margin and HMM parameters jointly. For the purpose of comparison, the final margin values achieved by SME_joint are listed in Table 3.5. These values are similar to those margin values preset in Table 3.4. There are only very small differences between the performance of SME and SME_joint in Table 3.3. This demonstrates that the two proposed solutions are nearly equivalent because of the mapping relationship between λ and ρ.

Figure 3.4 shows the string accuracy improvement of SME in the training set for different SME models after 200 iterations. Although the initial string accuracies (got from

Table 3.5. Margin value obtained by joint optimization.

1-mix	2-mix	4-mix	8-mix	16-mix
5.2	5.9	7.1	7.4	9.6

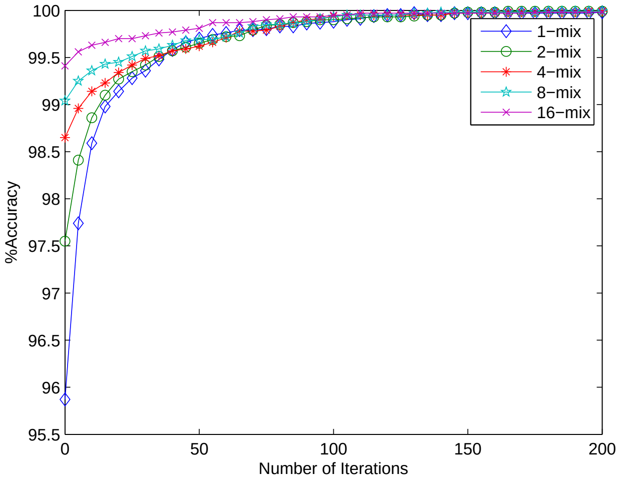

Figure 3.4. String accuracy of SME for different models in the TIDIGITS training set.

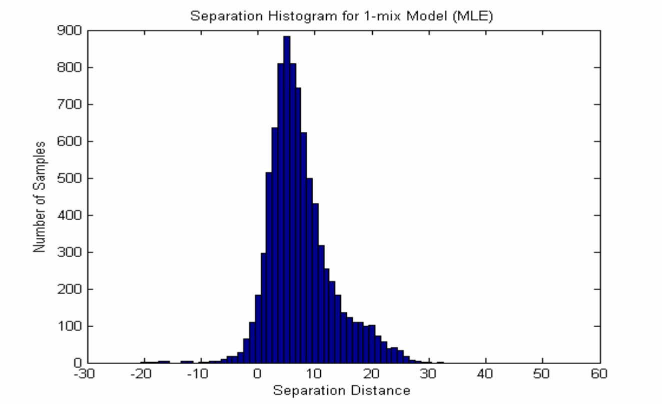

Figure 3.5. The histogram of separation distances of 1-mixture MLE model in the TIDIGITS training set.

MLE models) were very different, all SME models ended up with nearly the same accuracies of 99.99%. The training errors are nearly the same for all of these different mixture models, and the margin plays a significant role in the test risk bound, resulting in different test errors, which are listed in Table 3.3 (to be discussed later).

Figures 3.5 and 3.6 compare histograms of the measure defined in Eq. (3.6) with the normalized LLR for the case of a 1-mixture GMM before and after SME training. Usually, the larger the separation value, the better the models are. We observe in Figure 3.6 a very sharp edge around a value of 5, which is the soft margin value for the 1-mixture model update shown in the left-most column of Table 3.4. It is clear that when SME finishes

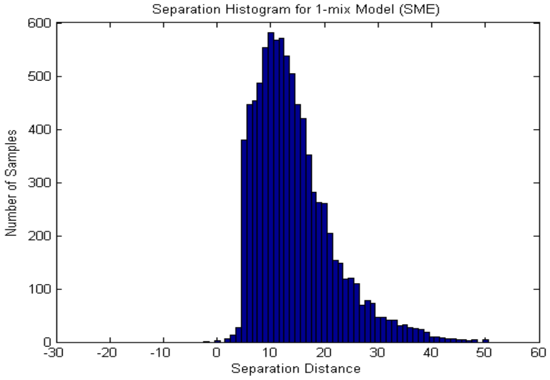

Figure 3.6. The histogram of separation distances of 1-mixture SME model in the TIDIGITS training set.

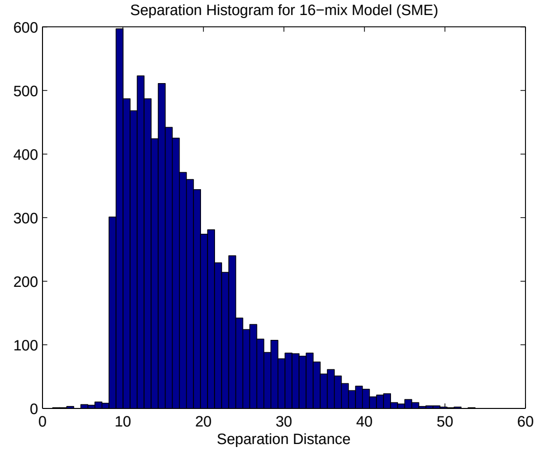

Figure 3.7. The histogram of separation distances of 16-mixture SME model in the TIDIGITS training set.

parameter update, most samples that have separation values less than the specified margin move to the right side of histogram, resulting in separation values greater than the margin value. This demonstrates the effectiveness of the SME algorithms. We can also see the effect in Figure 3.7 with the histogram separation for the 16-mixture case after SME update. The sharp edge now is around 11, the margin shown in the right-most column in Table 3.4. With a greater margin, the 16-mixture model can attain a string accuracy of 99.32% in the testing set, while the 1-mixture model can only get 98.76%, although both models have nearly the same string accuracy in the training set. This observation is greatly consistent with the test risk bound inequality of Eq. (3.1).

For high-accuracy tasks such as TIDIGITS, it is interesting to test the significance of SME compared with other methods. For each mixture model setting, we denote p_1 as the accuracy of the SME model and denote p_2 as the accuracy for other models from MLE, MCE, and LME. If p_1 and p_2 are assumed independent, then we have the following hypothesis testing problem [60]:

$$\begin{aligned} H_0 &: p_1 = p_2 \\ H_1 &: p_1 > p_2 \end{aligned} \tag{3.19}$$

The testing statistic is defined as:

$$z = \frac{\sqrt{N}(p_1 - p_2)}{\sqrt{(p_1(1-p_1))} + \sqrt{(p_2(1-p_2))}}, \tag{3.20}$$

where N denotes the total number of samples (8700 here).

A decision is made according to the following:

$$\begin{cases} \text{accept } H_0, & \text{if } z < Z_\alpha \\ \text{reject } H_0, & \text{if } z > Z_\alpha \end{cases}. \tag{3.21}$$

Z_α is called the upper quantile. Here, Z_α was set as 0.025. If hypothesis H_0 is rejected, the SME model is significantly better at the confidence level of 97.5%. For every mixture setting, the hypothesis testing was performed according to Eq. (3.21). In Table 3.3, an asterisk is used to denote when the performance of SME is significantly better.

Table 3.3 compares different training methods with various numbers of mixture components. Only string accuracies are listed in Table 3.3. At this high level of performance in TIDIGITS, the string accuracy is a strong indicator of model effectiveness. For the task of string recognition, the interest is usually in whether the whole string is correct. Therefore, string accuracy is more meaningful than word accuracy in TIDIGITS.

Clearly SME outperforms MLE and MCE significantly and is consistently better than LME. For 1-mixture SME models, the string accuracy is 98.76%, which is better than that of the 16-mixture MLE models. The goal of our design is to separate the models as far as possible instead of modeling the observation distributions. With SME, even 1-mixture models can achieve satisfactory model separation. The excellent SME performance is attributed to the well-defined model separation measure and good objective function for generalization.

To compare the generalization capability of SME with MLE and MCE, we plot the histograms of the separation measure defined in Eq. (3.6) for the testing utterances in Figure 3.8 for the 16-mixture MLE, MCE, and SME models. As indicated in the right-most curve, SME achieves a significantly better separation than both MLE and MCE in the testing set because of direct model separation maximization and better generalization.

It is interesting to compare the generalization capability of MLE, MCE, and SME for the 1-mixture setting. In Figure 3.9, MCE has a rightmost tail, which means that MCE has a better separation for samples in its right tail. However, the performance in Table 3.3 shows that SME outperforms MCE in the 1-mixture case. The reason is that SME has fewer samples that have separation distances smaller than 0, as shown in Figure 3.9.

As a conclusion, SME puts more focus on samples that are possibly misclassified. If the underlying model has less modeling power with few parameters, SME increases the separation distances of the samples with small distances, as shown in Figure 3.9. In contrast, if the underlying model has better modeling power with many parameters, SME consistently increases the separation distances of all the samples, as shown in Figure 3.8.

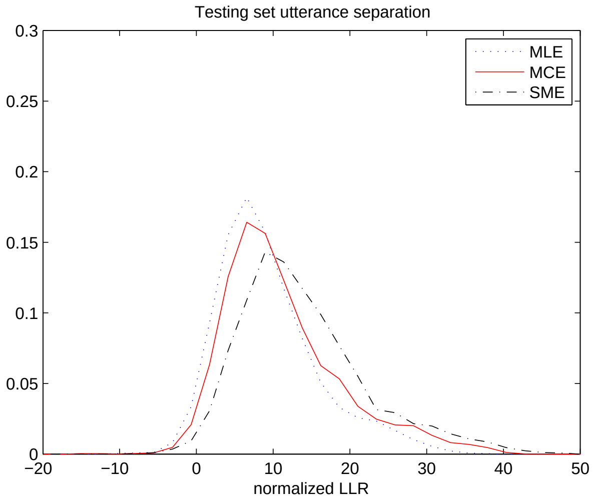

Figure 3.8. The histogram of separation distances of 16-mix model of MLE, MCE, and SME in the TIDIGITS testing set. The short dashed curve, line curve, and dotted curve correspond to MLE, MCE, and SME models.

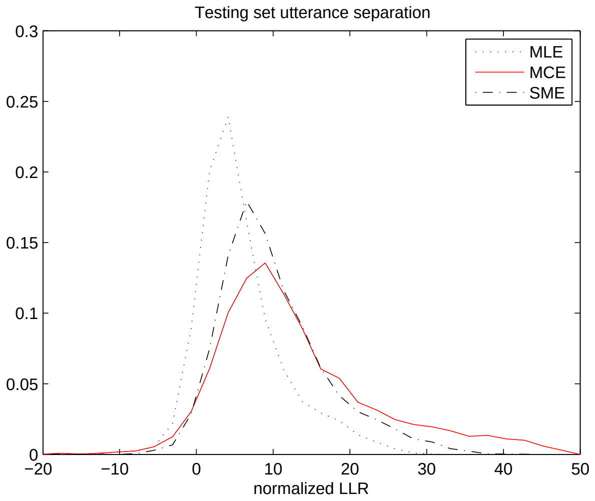

Figure 3.9. The histogram of separation distances of 1-mix model of MLE, MCE, and SME in the TIDIGITS testing set. The short dashed curve, line curve, and dotted curve correspond to MLE, MCE, and SME models.

Table 3.6. Comparison of GPD optimization and Quickprop optimization for SME.

	SME (GPD)	SME (Quickprop)
1-mix	98.76%	98.82%
2-mix	98.95%	99.15%
4-mix	99.20%	99.26%
8-mix	99.29%	99.31%
16-mix	99.30%	99.39%

Quickprop [83] is also a gradient-based optimization method. Here, we just give the comparison of the GPD and Quickprop algorithms in Table 3.6.

In every mixture case, Quickprop-optimized SME gets better performance than GPD-optimized SME with small gaps. The best string accuracy (99.39%) is got by Quickprop-optimized SME in the 16-mixture case. Quickprop also gives fast convergence speed on the TIDIGITS task. Usually, after around 200 iterations GPD-optimized SME converges while SME with Quickprop takes about 50 iterations.

3.7 Conclusion

We have proposed a novel discriminative training method, called SME, to achieve both high accuracy and good model generalization. This proposed method utilizes the successful ideas of soft margin in SVMs to improve generalization capability. It directly maximizes the separation of competing models to enhance the testing samples to approach a correct decision if the deviation from training models is within a safe margin. Frame and utterance selections are integrated into a unified framework to select the training utterances and frames critical for discriminating competing models. From the view of statistical learning theory, we show that SME can minimize the approximate risk bound on the test set. The choice of various loss functions is illustrated and different kinds of separation measures are defined under a unified SME framework.

Tested on the NIST 03 and 05 LID evaluation sets, SME outperformed MCE on both sets in almost every iteration. Tested on the TIDIGITS database, even 1-mixture SME models can better separate different words and produce better string accuracy than 16-mixture MLE models. The performance of SME is consistently better than that of LME, and significantly better than those of MLE and MCE. The experiment coincides with the inequality of the test risk bound, showing that even though all the models have the same training errors, the test string accuracies differ because of different margin values associated with various models. SME has been optimized by GPD and Quickprop. The result here shows that SME with Quickprop has slightly better string accuracy than GPD-optimized SME. Quickprop can also provide fast convergence.

CHAPTER 4

SOFT MARGIN FEATURE EXTRACTION (SMFE)

Feature extraction is an important component in automatic speech recognition (ASR) systems. Most current ASR systems use Mel-frequency cepstrum coefficients (MFCCs) [16] as their standard input acoustic features. As shown in Figure 4.1, usually, lower dimension MFCCs are got by applying a discrete cosine transform (DCT) on higher dimension log filter bank energies and following an optional cepstrum lifter. The DCT transformation matrix is fixed for all tasks. It has been demonstrated that data-dependent transformation is usually better than data-independent transformation in classification tasks [21]. Therefore, the DCT transformation may not be the best choice for specific ASR tasks. We hope to get better features with more discriminative information, and these discriminative features will benefit our ASR backend.

Linear discriminant analysis (LDA) [21] may be a choice for the data-dependent transformation. In [33],[41], LDA has been successfully used in ASR systems for feature extraction. However, there are two major disadvantages of LDA. One is that LDA assumes the underlying sample distribution is Gaussian. The other is that it assumes that the class samples are of equal variance. Heteroscedastic discriminant analysis (HDA) [54] was proposed to remove the equal variance assumption. However, HDA is under the framework of maximum likelihood estimation (MLE). Discriminative training (DT) technologies can also be applied to feature extraction for better performance. MCE was used in the work of

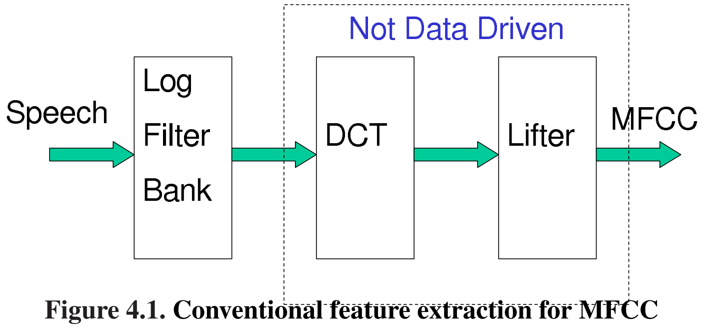

Figure 4.1. Conventional feature extraction for MFCC

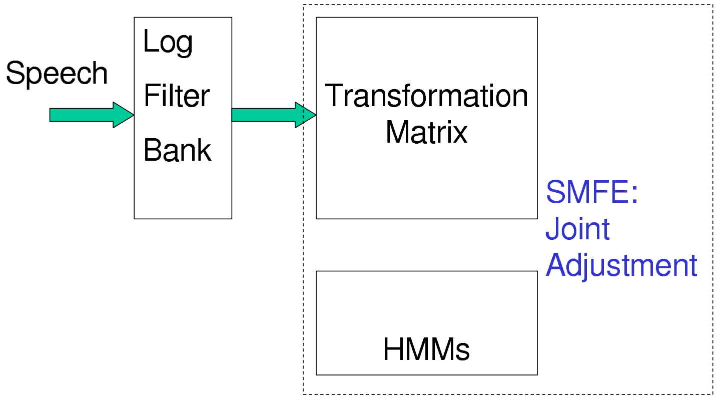

Figure 4.2. Soft margin feature extraction: jointly optimize feature and HMM parameters.

discriminative feature extraction [8] to get optimal lifters. Feature-space MPE was used to get discriminative feature in [100].

In this section, soft margin feature extraction (SMFE) [65] is proposed to jointly optimize the feature transformation matrix and HMM parameters under the framework of SME, as shown in Figure 4.2. Because the purpose of feature extraction in this study is to find a transformation matrix rather than DCT to reduce the dimension of log filter bank energies, dimension reduction and feature extraction are interchanged frequently in this section.

4.1 SMFE for Gaussian Observations

The objective of feature extraction is to find a matrix W to transform the original n-dimension feature vector x into a new $d(d < n)$-dimension vector y. It is formulated as: $y = Wx$. To simplify the formula, we use x to stand for O_{ij}, which is used in Eq. (3.7). To embed W into the framework of SMFE, we need to express $logp(x|S_i) - logp(x|\hat{S}_i)$ as a function of W.

We first investigate a simple case, in which the state observation probability is modeled by a Gaussian distribution.

An $n * n$-dimension matrix V is used to obtain $z = Vx$ (z is a n-dimension vector). If the probability density function (pdf) of z is modeled by a Gaussian with (μ_n, Σ_n) as the mean and covariance, the pdf of x can be expressed as:

$$p(x|\mu_n, \Sigma_n) = \frac{|V|}{(2\pi)^{n/2}|\Sigma_n|^{1/2}} exp\left\{-\frac{1}{2}(Vx - \mu_n)^T \Sigma_n^{-1}(Vx - \mu_n)\right\}. \tag{4.1}$$

For the purpose of dimension reduction, only the first d-dimension feature is needed. Therefore, the n-dimension vector of z and μ_n can be split into two sub-vectors with dimension d and $n - d$, respectively. Also, the $n * n$-matrix of V can be split into two matrices with dimension $d * n$ and $(n - d) * n$, respectively. That is

$$z = \begin{bmatrix} y \\ y_{n-d} \end{bmatrix}, \mu_n = \begin{bmatrix} \mu \\ \mu_{n-d} \end{bmatrix}, V = \begin{bmatrix} W \\ W_{(n-d)*n} \end{bmatrix}. \tag{4.2}$$

Block approximation for the covariance matrix is made as:

$$\Sigma_n = \begin{bmatrix} \Sigma & 0 \\ 0 & \Sigma_{n-d} \end{bmatrix}, \tag{4.3}$$

where Σ and Σ_{n-d} are the matrices with dimension $d * d$ and $(n - d) * (n - d)$, respectively.

Let (μ_{n+}, Σ_{n+}) and (μ_{n-}, Σ_{n-}) denote the means and covariance matrices of Gaussians of the target and competing labels, respectively. The difference between the log values of two Gaussian distributions will be

$$\begin{aligned} logp(x|S_i) - logp(x|\hat{S}_i) &= logp(x|\mu_{n+}, \Sigma_{n+}) - logp(x|\mu_{n-}, \Sigma_{n-}) \\ &= log|V| - \frac{1}{2}log|\Sigma_{n+}| - \frac{1}{2}(Vx - \mu_{n+})^T \Sigma_{n+}^{-1}(Vx - \mu_{n+}) \\ &\quad - \left\{log|V| - \frac{1}{2}log|\Sigma_{n-}| - \frac{1}{2}(Vx - \mu_{n-})^T \Sigma_{n-}^{-1}(Vx - \mu_{n-})\right\} \\ &= R + R_{n-d}, \end{aligned} \tag{4.4}$$

where

$$R = \frac{1}{2}log|\Sigma_-| + \frac{1}{2}(Wx - \mu_-)^T \Sigma_-^{-1}(Wx - \mu_-) - \frac{1}{2}log|\Sigma_+| - \frac{1}{2}(Wx - \mu_+)^T \Sigma_+^{-1}(Wx - \mu_+), \tag{4.5}$$

and

$$\begin{aligned} R_{n-d} &= \frac{1}{2}log|\Sigma_{(n-d)-}| + \frac{1}{2}(W_{(n-d)*n}x - \mu_{(n-d)-})^T \Sigma_{(n-d)-}^{-1}(W_{(n-d)*n}x - \mu_{(n-d)-}) \\ &\quad -\frac{1}{2}log|\Sigma_{(n-d)+}| - \frac{1}{2}(W_{(n-d)*n}x - \mu_{(n-d)+})^T \Sigma_{(n-d)+}^{-1}(W_{(n-d)*n}x - \mu_{(n-d)+}). \end{aligned} \tag{4.6}$$

If $|\Sigma_{n-d}|^{-1/2} exp\left\{-\frac{1}{2}(W_{(n-d)*n}x-\mu_{n-d})^T\Sigma_{n-d}^{-1}(W_{(n-d)*n}x-\mu_{n-d})\right\}$ is assumed to be a constant, then R_{n-d} is equal to 0. This assumption makes sense because the target of dimension reduction is to discard unnecessary information and this additional dimension cannot be removed if the $(n-d)$-dimension features of different classes are very different. A similar assumption is used in HDA [54]. Now, we only need to be concerned about R, which is a function of W. As given in Eq. (3.7), the objective function of SME is

$$L^{SME}(\rho,\Lambda)=\frac{\lambda}{\rho}+\frac{1}{N}\sum_{i=1}^{N}\left(\rho-\frac{1}{n_i}\sum_j log\left[\frac{P_\Lambda(O_{ij}|S_i)}{P_\Lambda(O_{ij}|\hat{S}_i)}\right]I(O_{ij}\in F_i)\right)I(O_i\in U). \tag{4.7}$$

Then, R in Eq. (4.5) is used to replace $logp(O_{ij}|S_i)-logp(O_{ij}|\hat{S}_i)$ in Eq. (4.7).

4.2 SMFE for GMM Observations

For the case when the state observation probability is modeled by a GMM:

$$logp(x|\mu_n,\Sigma_n)=log|V|+log\left(\sum_j\frac{c_j}{(2\pi)^{n/2}|\Sigma_{jn}|^{1/2}}exp\left\{-\frac{1}{2}(Vx-\mu_{jn})^T\Sigma_{jn}^{-1}(Vx-\mu_{jn})\right\}\right). \tag{4.8}$$

Here, $(c_j,\mu_{jn},\Sigma_{jn})$ are the weight, mean, and covariance matrix of the j-th Gaussian component of GMM. The $log|V|$ item still can be removed by the subtraction of two log values of state observation probabilities of the target and competing classes. By applying the same splitting strategy for each component of GMMs as the case for Gaussian and making the constant assumption for $|\Sigma_{j,n-d}|^{-1/2}exp\left\{-\frac{1}{2}(W_{(n-d)*n}x-\mu_{j,n-d})^T\Sigma_{j,n-d}^{-1}(W_{(n-d)*n}x-\mu_{j,n-d})\right\}$, the difference of log values can be written as

$$\begin{aligned}
log\quad & p(x|S_i)-logp(x|\hat{S}_i)\\
= \quad & R\\
= \quad & log\left(\sum_j\frac{c_{j+}}{(2\pi)^{n/2}|\Sigma_{j+}|^{1/2}}exp\left\{-\frac{1}{2}(Wx-\mu_{j+})^T\Sigma_{j+}^{-1}(Wx-\mu_{j+})\right\}\right)\\
& -log\left(\sum_j\frac{c_{j-}}{(2\pi)^{n/2}|\Sigma_{j-}|^{1/2}}exp\left\{-\frac{1}{2}(Wx-\mu_{j-})^T\Sigma_{j-}^{-1}(Wx-\mu_{j-})\right\}\right),
\end{aligned} \tag{4.9}$$

where $(c_{j+},\mu_{j+},\Sigma_{j+})$ and $(c_{j-},\mu_{j-},\Sigma_{j-})$ are the weight, mean, and covariance matrix of the

j-th component of GMM for target and competing classes, respectively. Equation (4.10) can then be plugged into Eq. (4.7) for the solution.

4.3 Implementation Issue

In our implementation of SMFE, we simplify the process by using the same transformation matrix W for the static, first and second order time derivatives of the log filter bank energies. Let x denote the static log filter bank energies, Δx and $\Delta\Delta x$ denote the first and second order derivatives of x. Then the new transformed static feature vector is given by: $y = Wx$, and the dynamic features of y are: $\Delta y = W\Delta x$ and $\Delta\Delta y = W\Delta\Delta x$. The final feature Q is composed of y, Δy, $\Delta\Delta y$, log energy e and its derivatives Δe, $\Delta\Delta e$ as $Q = (Wx, W\Delta x, W\Delta\Delta x, e, \Delta e, \Delta\Delta e)^T$. Then, R in Eq. (4.10) can be expressed:

$$\begin{aligned} R = log&\left(\sum_j \frac{c_{j+}}{(2\pi)^{n/2}|\Sigma_{j+}|^{1/2}} exp\left\{-\frac{1}{2}(Q-\mu_{j+})^T\Sigma_{j+}^{-1}(Q-\mu_{j+})\right\}\right) \\ -log&\left(\sum_j \frac{c_{j-}}{(2\pi)^{n/2}|\Sigma_{j-}|^{1/2}} exp\left\{-\frac{1}{2}(Q-\mu_{j-})^T\Sigma_{j-}^{-1}(Q-\mu_{j-})\right\}\right). \end{aligned} \quad (4.10)$$

Eq. (4.10) is a function of the matrix W. Now, we can embed R into the SME formulation to replace $logp(O_{ij}|S_i) - logp(O_{ij}|\hat{S}_i)$ and use GPD to get the final parameters of transformation matrix W and all the HMM parameters.

4.4 Experiments

All the experiments in Chapter 3 use MFCCs as the front end feature. As discussed earlier, SMFE should boost the ASR performance with joint optimization of the acoustic feature and HMMs. The proposed SMFE framework was evaluated in the following.

Five sets of models are trained and compared in Table 4.1. The MLE and SME models trained with MFCCs are denoted as MLE_M and SME_M in Table 4.1. In parallel, LDA is used to extract the acoustic features. For each speech frame, there are 24 log filter bank energies. LDA was applied to reduce the dimension from 24 to 12. To get the LDA transformation, each HMM-state was chosen as a class. This dimension-reduced feature

Table 4.1. SMFE: Testing set string accuracy comparison with different methods. Accuracies marked with an asterisk are significantly different from the accuracy of the SMFE model ($p<0.1$, paired Z-test, 8700 d.o.f. [60]).

	MLE_M	SME_M	MLE_L	SME_L	SMFE
1-mix	95.20%*	98.76%*	96.82%*	98.91%*	99.13%
2-mix	96.20%*	98.95%*	97.82%*	99.15%*	99.36%
4-mix	97.80%*	99.20%*	98.51%*	99.31%	99.44%
8-mix	98.03%*	99.29%*	98.63%*	99.39%*	99.56%
16-mix	98.36%*	99.30%*	98.93%*	99.46%*	99.61%

is concatenated with energy and then extended with its first and second order derivatives to form a new 39-dimension feature. MLE and SME models were also trained based on this new LDA-based feature. These models are MLE_L and SME_L in Table 4.1. Finally, initiated with the models MLE_L and the LDA transformation matrix, SMFE models were trained to get the optimal HMM parameters and transformation matrix. The results of SMFE are also listed in Table 4.1.

It is clear that LDA-based features outperform MFCCs for both the MLE and SME models. Shown in the last column of Table 4.1, SMFE models achieved the best performance. Even the 1-mixture SMFE model can get better performance than 16-mixture MLE models with MFCCs or LDA-based features. SMFE achieved 99.61% string accuracy for 16-mixture model. This is a large improvement from the original SME work, in which MFCCs were used. The excellent SMFE performance is attributed to the joint optimization of the acoustic feature and HMM parameters.

For every mixture setting, hypothesis testing was performed according to Eq. (3.21). Z_α is set to 0.1. If hypothesis H_0 is rejected, SMFE is significantly better at the confidence level of 90%. In Table 4.1, an asterisk is used to denote when the performance of SMFE is significantly better. It is shown that SMFE is significantly better than nearly all the other models at a significance level of 90%. The only exception is the 4-mixture SME_L model.

4.5 Conclusion

By extending our previous work of SME, we proposed a new discriminative training method, called SMFE, to achieve even higher accuracy and better model generalization. By jointly optimizing the acoustic feature and HMM parameters under the framework of SME, SMFE performs much better than SME, and significantly better than MLE. Tested on the TIDIGITS database, even 1-mixture model can get string accuracy of 99.13%. And 99.61% string accuracy was got with 16-mixture SMFE model. This is a great improvement comparing to our original SME work that uses MFCCs as acoustic feature. This SMFE work again demonstrates the success of soft margin based method, which directly makes usage of the successful ideas of soft margin in support vector machines to improve generalization capability, and of decision feedback learning in minimum classification error training to enhance model separation in classifier design.

This is our initial study. We need to work on many related research issues to further complete the work of SMFE. In this study, feature transformation matrix only works on the static log filter bank energies of the current frame. In [54], great benefits were obtained by using the frames in context before and after the current frame. We will try to incorporate these context frames into SMFE optimization. Secondly, in [67] we have shown that SME also worked well on a large vocabulary continuous speech recognition task. We will try to demonstrate the effectiveness of SMFE on the Wall Street Journal task in future work.

CHAPTER 5

SME FOR ROBUST AUTOMATIC SPEECH RECOGNITION

Environment robustness in speech recognition remains a difficult problem despite many years of research and investment [98]. The difficulty arises due to many possible types of distortions, including additive and convolutive distortions and their mixes, which are not easy to predict accurately during recognizers' development. As a result, the speech recognizer trained using clean speech often degrades its performance significantly when used under noisy environments if no compensation is applied [30], [57].

Different methodologies have been proposed in the past for environment robustness in speech recognition over the past two decades. As shown in Figure 5.1, there are three main classes of approaches [57]. In the signal domain, the testing speech signal can be cleaned with classic speech enhancement technologies (e.g., spectral subtraction [10]). In the feature domain, the distorted acoustic feature can be normalized to match training feature (e.g., cepstral mean normalization [5], and stereo-based piecewise linear compensation for environments [17]). In the model domain, the original trained model can be adapted to match the testing environment (e.g., maximum likelihood linear regression [59], and maximum a posteriori adaptation [29]).

In contrast to the above methods, margin-based learning may provide a set of models with generalization capabilities to deal with noise robustness without actual compensation at operating time. The formulation of margin-based methods allows some mismatch

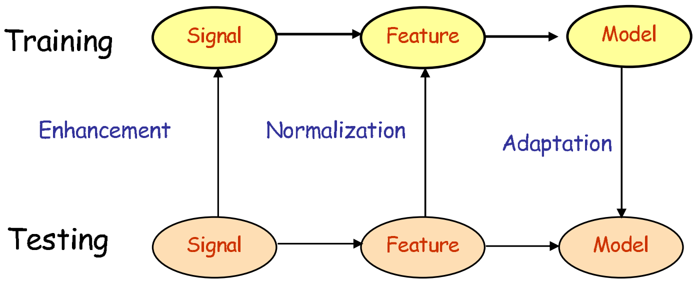

Figure 5.1. Methods for robust speech recognition

between the training and testing conditions. By securing a margin from the decision boundaries to the training samples, a correct decision can still be made if the mismatches between the testing and training samples are smaller than the value of the margin. Although this nice property of margin-based methods is quite desirable, we are not aware of any previously reported work on robust ASR with margin-trained HMMs. We study discriminative training (DT) methods, such as minimum classification error (MCE) and SME training, and investigate if they generalize well to adverse conditions without applying any special compensation techniques.

The generalization issue for the above DT methods was evaluated on the standard Aurora 2 task of recognizing digit strings in noise and channel-distorted environments. The clean training set and multi-condition training set, which consist of 8440 clean utterances and multi-condition utterances, individually, were used to train the baseline maximum likelihood estimation (MLE) HMMs. The test material consists of three sets of distorted utterances. The data in set a and set b consist of eight different types of additive noise, while set c contains two different types of noise plus additional channel distortion. Each type of noise is added into a subset of clean speech utterances, with seven different levels of SNRs. This generates seven SNR-specific subgroups, namely clean, 20db, 15db, 10db, 5db, 0db, and -5db SNRs, of testing sets for each specified noise type. The baseline experiment configuration follows the standard script provided by ETSI [37], including the simple "backend" of HMMs trained using HTK. The acoustic features are 13-dimension MFCCs, appended by their first- and second-order time derivatives. The baseline clean-trained and multi-condition-trained HMMs got 60.06% and 86.39% word accuracy (Acc), separately.

For a fair comparison, both SME and MCE were trained with similar implementations and differed only in the individual algorithm parts. Because the Aurora2 task is a connected-digit task, we need not use lattice for competing strings and extended Baum-Welch (EBW) for parameter optimization, GPD was used to train HMM parameters and N-Best lists were used to construct competing strings. SME used only one competing string while MCE used 5 competing strings. If only one competing string was used for MCE, a worse performance was obtained. Therefore, we just reported the best performance for MCE with 5 competing strings.

For easy access, we rewrite the formulation and solution of SME in the following. SME optimizes a combination of empirical risk minimization and soft margin maximization:

$$L^{SME}(\rho, \Lambda) = \frac{\lambda}{\rho} + R_{emp}(\Lambda) = \frac{\lambda}{\rho} + \frac{1}{N}\sum_{i=1}^{N} l(O_i, \Lambda), \tag{5.1}$$

where ρ is the soft margin, and λ is a coefficient to balance the soft margin maximization and the empirical risk minimization. A larger λ favors a larger margin value, which corresponds to better generalization.

The soft margin ρ and the HMM parameters Λ can be optimized by iteratively using the generalized probabilistic descent (GPD) algorithm on the training set as in [50], with η_t and κ_t as step sizes for iteration t:

$$\begin{cases} \Lambda_{t+1} = \Lambda_t - \eta_t \nabla L^{SME}(\rho, \Lambda)|_{\Lambda=\Lambda_t} \\ \rho_{t+1} = \rho_t - \kappa_t \nabla L^{SME}(\rho, \Lambda)|_{\rho=\rho_t} \end{cases} \tag{5.2}$$

5.1 Clean Training Condition

Usually, DT methods are only reported to work on the multi-style training set or on the denoised testing condition combined with other noise robustness techniques (e.g., [123]). In this section, we investigate whether DT methods can work well in a mismatched condition (i.e., clean training case) without using other noise robust technologies.

Table 5.1 lists the detailed test accuracies for MLE and DT methods (MCE and SME with different values of the balance coefficient λ) trained with clean data. The average digit accuracy was evaluated by averaging the accuracies on the subgroups with the SNRs from 0db to 20db, as described by the ETSI standard [37]. Table 5.2 shows the relative WER reductions for MCE and SME from the baseline. In average, SME with different balance coefficients achieved about 17%-29% relative WER reductions from MLE while

Table 5.1. Detailed test accuracies for MLE, MCE, and SME with different balance coefficient λ using clean training data.

Word Acc	clean	20db	15db	10db	5db	0db	-5db	Avg.
MLE	99.03	94.07	85.04	65.52	38.61	17.09	8.53	60.06
MCE	99.50	95.43	86.88	66.95	37.29	14.27	6.58	60.16
SME (λ=10)	**99.58**	97.00	91.39	75.73	48.61	21.59	8.53	66.86
SME (λ=50)	99.56	**97.61**	93.42	80.74	55.32	27.29	11.01	70.88
SME (λ=100)	99.54	97.60	**93.85**	**81.98**	56.59	28.19	12.28	71.64
SME (λ=200)	99.49	97.55	93.76	81.94	**56.96**	28.67	13.14	**71.78**
SME (λ=300)	99.46	97.41	93.55	81.43	56.54	**28.86**	13.33	71.56
SME (λ=400)	99.45	97.34	93.49	81.57	56.32	28.68	**13.44**	71.48

Table 5.2. Relative WER reductions for MCE, and SME from MLE baseline using clean training data.

Rel. WER red.	clean	20db	15db	10db	5db	0db	-5db	Avg.
MCE	48.45%	22.93%	12.30%	4.15%	-2.15%	-3.4%	-2.1%	0.25%
SME (λ=10)	**56.70%**	49.41%	42.45%	29.6%	16.29%	5.43%	0.00%	17.03%
SME (λ=50)	56.64%	**59.70%**	56.02%	44.14%	27.22%	12.30%	2.71%	27.09%
SME (λ=100)	52.58%	59.53%	**58.89%**	**47.74%**	29.29%	13.39%	4.10%	28.99%
SME (λ=200)	47.42%	58.68%	58.29%	47.62%	**29.89%**	13.97%	5.04%	**29.34%**
SME (λ=300)	44.33%	56.32%	56.89%	46.14%	29.21%	**14.20%**	5.25%	28.79%
SME (λ=400)	43.30%	55.14%	56.48%	46.55%	28.85%	13.98%	**5.37%**	28.59%

MCE obtained only 0.2% relative WER reduction due to poor performance in low SNR conditions. Examining the results in detail, we see that the characteristics of individual recognition accuracies for different SNR subgroups are very different for these two DT methods.

For each SNR subgroup (column) in Tables 5.1-5.2, the best performance is shown in bold font. In all cases in Tables 5.1-5.2, SME outperformed MCE. It is interesting to note that for the clean testing subgroup, both SME and MCE got comparable accuracies, with the relative WER reductions ranging from 43% to 57%. This implies that under matched conditions, both DT methods performed similarly on this Aurora2 task. The major difference occurred in the mismatched testing subgroups. In 20db and 15db SNR conditions that are not severely distorted from the clean training conditions, MCE got 23% and 12% relative WER reductions. In contrast, all the SME methods achieved at least 42% relative WER reductions in the 20db and 15db SNR cases, with the best performance around 60%

relative WER reduction. In the 10db case, MCE got less than 5% relative WER reductions while most SME can still get more than 40%. In the 5db, 0db, and -5db SNR scenarios, which are severely distorted conditions, MCE can get even worse performance than the MLE baseline. In contrast, SME still obtained satisfactory relative WER reductions in all cases.

The above observations show that all these discriminative training methods have no big difference in matched testing conditions on the Aurora2 task. Big difference exists in the mismatched testing conditions. Because of the margin, SME greatly improved the generalization ability, allowing the classifier to make a correct decision as long as the testing samples deviate within the margin from the training samples.

Examining the results of Table 5.2 in detail, we see that the best relative WER reductions for clean, 20db, and 15db SNR testing cases are 57%, 60%, and 59%. This demonstrates the effectiveness of the margin in the cases of 20db and 15db SNR testing conditions which are not as severely distorted from clean training condition, the margin can easily cover the distortion as shown in Figure 5.2. However, for the 10db, 5db, 0db, and -5db cases, the relative WER reductions keep decreasing since these conditions are increasingly pulled away from clean training and the distortions cannot be easily covered by a margin.

SME, with different balance coefficients λ, affects the performance in different testing conditions. In the cleaning testing case, SME with the smallest λ (λ=10) gives the best accuracy, since it is a matched testing and the classifier with a focus on empirical risk minimization works the best. With the testing SNR decreasing, larger λ is required to give more weights to margin maximization which in turn gives better generalization. As a result, SME with λ=50 works best in the 20db SNR case, while SME with λ=100 gets best accuracies in the 15db and 10db SNR cases. SME with λ=200, 300, and 400 obtain the most word error reduction in the 5db, 0db, and -5db SNR testing conditions. We can easily see the trend of best SME with respect to the coefficient.

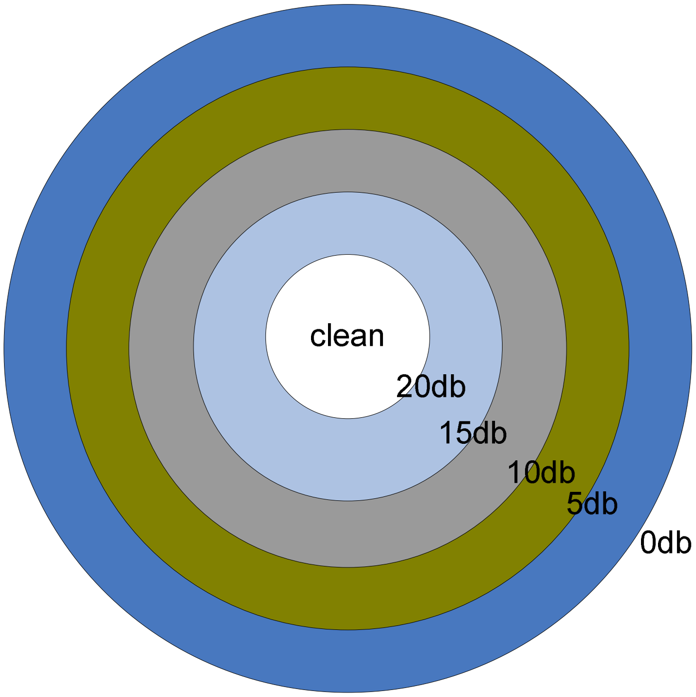

Figure 5.2. The distortion level from clean condition for different SNRs.

Table 5.3. Detailed accuracies on testing set a, b, and c for MLE and SME with different balance coefficient λ using clean training data.

Word Acc	set a	set b	set c	Avg.
MLE	61.34	55.75	66.14	60.06
SME (λ=10)	68.35	63.79	70.04	66.86
SME (λ=50)	72.30	69.76	**70.25**	70.88
SME (λ=100)	72.79	71.77	69.08	71.64
SME (λ=200)	72.86	**72.06**	69.04	**71.78**
SME (λ=300)	**72.90**	71.46	69.06	71.56
SME (λ=400)	72.65	71.94	68.21	71.48

Table 5.3 lists the detailed accuracies on testing set a, b, and c for MLE and SME with different balance coefficient λ using clean training data. We can see that a clear accuracy increasing trend from λ=10 to λ=200 in set a and set b. However, the trend cannot keep increasing for the reason we analyzed above. For set c, the increasing trend stops only at λ=50. This maybe set c is more difficult than set a and b, with additional channel distortion. In all cases, the margin provides SME better accuracies, compared to the original MLE baseline.

The original generalization property of margin-based classifiers in statistical learning theory [119] requires the training and testing samples to be from the identically independent distributions (i.i.d.). The results here show that SME performs very well even if the training and testing distributions are very different, which means that SME may have even better generalization property. We believe this is because the formulation of SME can push training samples away from the decision boundary with a distance of the margin. That margin in turn allows correct decision be made as long as the testing samples are distorted from training samples within a distance less than the value of the margin.

5.2 Multi-condition Training Condition

Table 5.4 lists testing results for MLE, MCE, and SME using multi-condition training data. Table 5.5 shows the relative WER reductions for MCE and SME from the baseline using multi-condition training data. Table 5.6 gives the detailed word accuracies for testing set a, b, and c. Both DT methods obtained similar performance, with around 10% relative WER reduction, which is similar to the relative WER reduction reported in [123] when MCE was only applied to multi-condition trained HMMs without combining with other methods.

The improvement of SME in multi-condition training is not as impressive as that in the clean training case. The possible reason is given in the following. For model trained from the clean data, the accuracy is as high as 99% on the clean testing data. From Eq.

Table 5.4. Detailed test accuracies for MLE, MCE, and SME using multi-condition training data.

Word Acc	clean	20db	15db	10db	5db	0db	-5db	Avg.
MLE	98.52	97.35	96.29	93.79	85.52	59.00	24.50	86.39
MCE	98.79	98.23	97.37	95.23	86.40	60.37	24.69	87.52
SME (λ=5)	98.99	98.29	97.48	95.22	87.03	60.58	24.43	87.72
SME (λ=10)	98.95	98.20	97.41	95.25	87.25	61.16	25.33	87.86
SME (λ=20)	98.97	98.29	97.52	95.27	87.02	60.05	24.71	87.63
SME (λ=50)	99.00	98.30	97.51	95.17	86.67	59.05	24.65	87.34

(5.1), we can see that classifier learning balances empirical risk minimization and margin maximization. Because the empirical risk (i.e., the risk on the training set) is already very small, the focus of classifier learning is to maximize the margin. The resulted large margin in turn gives better generalization for the classifier, making it perform very well in mismatched testing conditions although the classifier is trained only with clean data. In contrast for models trained from multi-condition data, the accuracy is only around 86% on the multi-condition test data. The classifier training has to care about both the empirical risk minimization and the margin maximization. As a result, the margin cannot play a significant role to contribute to significantly improving the generalization capabilities of SME-trained models.

Another possible reason is that the multi-condition training data already covers different environments. SME may be hard to generalize the multi-condition trained model. We may see this from Table 5.6. The word accuracy for test set a decreases as λ increases since the noises in test set a has already been observed in the multi-condition training set. In contrast, the noises in test set b doesn't occur in the training set, therefore bigger margin in SME plays a role for generalization, resulting the increasing word accuracies for test set b.

5.3 Single SNR Training Condition

Although 29% relative WER reduction was achieved in the clean training multi-condition testing case, the average word accuracy is only 71.78. That number is relatively low for real

Table 5.5. Relative WER reductions for MCE and SME from MLE baseline using multi-condition training data.

Rel. WER red.	clean	20db	15db	10db	5db	0db	-5db	Avg.
MCE	18.24%	33.21%	29.11%	23.19%	6.08%	3.34%	0.25%	8.30%
SME (λ=5)	31.76%	35.47%	32.08%	23.03%	10.43%	3.85%	-0.09%	9.77%
SME (λ=10)	29.05%	32.08%	30.19%	23.51%	11.95%	5.27%	1.10%	10.80%
SME (λ=20)	30.41%	35.47%	33.15%	23.83%	10.36%	2.56%	0.28%	9.11%
SME (λ=50)	32.43%	35.85%	32.88%	22.22%	7.94%	0.12%	0.20%	6.98%

Table 5.6. Detailed accuracies on testing set a, b, and c for SME with different balance coefficient λ using multi-condition training data.

Word Acc	set a	set b	set c	Avg.
SME (λ=5)	89.75	87.54	84.02	87.72
SME (λ=10)	89.62	87.83	84.40	87.86
SME (λ=20)	89.27	87.97	83.66	87.63
SME (λ=50)	88.65	88.06	83.25	87.34

life applications. As discussed in Section 5.1, larger λ will give better generalization. This may cause some decreased performance on the matched testing condition. As shown in Figure 5.2, the model trained with clean data requires large margin to cover the distortions from SNR -5db to SNR 20db. We may add noises into the clean training data to get a single SNR training condition. In such a condition, it may be easy for margin to cover all the distortion cases, such as the case in Figure 5.3.

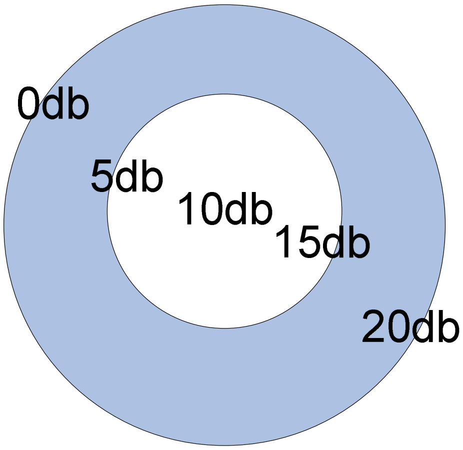

Figure 5.3. The distortion level from a 10db SNR training condition for different SNRs.

In the following, we will add noise into the clean training set, and get 5 different training sets with 20db, 15db, 10db, 5db, and 0db SNRs. As the multi-condition training, four types

Table 5.7. Detailed test accuracies for MLE and SME using 20db SNR training data.

Word Acc	clean	20db	15db	10db	5db	0db	-5db	Avg.
MLE	95.65	97.00	93.66	83.39	57.14	21.40	1.14	70.04
SME (λ=50)	96.92	98.71	97.14	91.38	71.19	33.39	11.78	78.36
SME (λ=100)	97.41	98.60	97.08	91.12	70.46	33.83	12.53	78.22

Table 5.8. Relative WER reductions for SME from MLE baseline using 20db SNR training data.

Rel. WER red.	clean	20db	15db	10db	5db	0db	-5db	Avg.
SME (λ=50)	29.20%	57.00%	54.89%	48.10%	32.78%	15.25%	10.76%	27.77%
SME (λ=100)	40.46%	53.33%	53.94%	46.54%	31.08%	15.81%	11.52%	27.30%

of noise are randomly added. They are: subway, babble, car, and exhibition. These four types of noise are the same as those used in test set a. In every distorted training set, there are still 8440 utterances. SME with different balance coefficients λ are investigated. Since the focus is on SME, MCE will not be compared in the following experiments.

5.3.1 20db SNR Training Condition

Tables 5.7, 5.8, and 5.9 give detailed test accuracies, relative WER reductions, and detailed accuracies on test set a, b, and c using the 20db SNR training data. SME with λ=50 and λ=100 are tested. SME methods can get about 27% relative WER reductions, with around 78% word accuracies. Small λ (50) gives better performance around 20db SNR testing condition while big λ (100) generates better accuracy for testing conditions far away from 20db SNR.

5.3.2 15db SNR Training Condition

Tables 5.10, 5.11, and 5.12 give detailed test accuracies, relative WER reductions, and detailed accuracies on test set a, b, and c using the 15db SNR training data. SME with

Table 5.9. Detailed accuracies on testing set a, b, and c for SME with different balance coefficient λ using 20db SNR training data.

Word Acc	set a	set b	set c	Avg.
SME (λ=50)	80.91	77.90	74.19	78.36
SME (λ=100)	80.73	78.12	73.39	78.22

Table 5.10. Detailed test accuracies for MLE and SME using 15db SNR training data.

Word Acc	clean	20db	15db	10db	5db	0db	-5db	Avg.
MLE	90.21	96.79	94.18	86.92	66.44	27.22	2.84	74.31
SME (λ=10)	90.44	98.24	96.95	92.35	76.34	39.08	12.04	80.59
SME (λ=30)	93.00	98.49	97.27	93.12	77.57	41.48	13.70	81.59
SME (λ=50)	93.38	98.47	97.36	93.22	77.72	42.11	13.95	81.78
SME (λ=80)	94.29	98.44	97.33	93.32	77.42	42.03	14.38	81.71
SME (λ=100)	94.93	98.48	97.27	93.24	77.17	41.95	14.53	81.62
SME (λ=200)	96.17	98.39	97.24	93.02	76.65	41.73	15.38	81.41

Table 5.11. Relative WER reductions for SME from MLE baseline using 15db SNR training data.

Rel. WER red.	clean	20db	15db	10db	5db	0db	-5db	Avg.
SME (λ=10)	2.35%	45.17%	47.59%	41.51%	29.50%	16.30%	9.47%	24.45%
SME (λ=30)	28.50%	52.96%	53.09%	47.40%	33.16%	19.59%	11.18%	28.34%
SME (λ=50)	32.38%	52.34%	54.64%	48.17%	33.61%	20.46%	11.43%	29.08%
SME (λ=80)	41.68%	51.40%	54.12%	48.93%	32.72%	20.35%	11.88%	28.80%
SME (λ=100)	48.21%	52.65%	53.09%	48.32%	31.97%	20.24%	12.03%	28.45%
SME (λ=200)	60.88%	49.84%	52.58%	46.64%	30.42%	19.94%	12.91%	27.64%

λ=10, 30, 50, 80, 100, and 200 are tested. The best SME option can get about 29% relative WER reduction, with around 82% word accuracy. For 20db and 15db SNR testing cases, SME can get more than 50% relative WER reduction. For 10db and 5db SNR testing cases, more than 40% and 30% relative WER reductions can be got. Large λ results in good performance on severely mismatched testing conditions. For example, the best word accuracies for clean and -5db testing cases are achieved by SME with the largest λ (200). Especially for clean condition, relative 61% relative WER reduction is obtained. However, because the accuracy for clean testing condition is not counted for the final average accuracy on the Aurora2 task, SME with λ=200 is not the best one in terms of the average accuracy. For practical applications, the accuracy for the clean testing condition should be an important criterion to evaluate the effectiveness of robust algorithms.

In Table 5.12, the word accuracy increases from SME with λ=10 to SME with λ=50, and then drops after SME with λ=80 for test set a. For test set b, the word accuracy keeps increasing with the increasing value of λ. This phenomenon also demonstrates the generalization property of SME. As stated previously, the 15db SNR training set has four

Table 5.12. Detailed accuracies on testing set a, b, and c for SME with different balance coefficient λ using 15db SNR training data.

Word Acc	set a	set b	set c	Avg.
SME (λ=10)	83.86	78.29	78.67	80.59
SME (λ=30)	84.68	80.16	78.25	81.59
SME (λ=50)	84.75	80.77	77.84	81.78
SME (λ=80)	84.62	81.13	77.04	81.71
SME (λ=100)	84.60	81.08	76.74	81.62
SME (λ=200)	84.13	81.70	75.39	81.41

types of noise, which are same as the noise used in test set a. The noise in test set b is unseen in the training set. Therefore, the performance on test set b is better with larger λ, which corresponds to better generalization. We cannot see any improvement in test set c. Test set c has one type of noise (totally two types of noise) occurred in the training set. A guess is that the additional channel distortion in test set c may bring trouble to SME.

5.3.3 10db SNR Training Condition

Tables 5.13, 5.14, and 5.15 give detailed test accuracies, relative WER reductions, and detailed accuracies on test set a, b, and c using the 10db SNR training data. SME with λ=50, 70, 80, 100, and 200 are tested. All SME options get around 29% relative WER reductions. The best one achieves 30% relative WER reduction, with around 84% word accuracy. For 20db, 15db, and 10db SNR testing cases, SME can get more than 40% relative WER reduction. For 5db and 0db SNR testing cases, more than 30% and 20% relative WER reductions can be got. Large λ results in good performance for clean testing case, with 90% word accuracy. This is the first time that nearly all SME methods get less than 90% word accuracy for the clean testing case. The word accuracy of 84% in this 10db SNR training condition is approaching the word accuracy of 87% in the multi-condition training condition. However, the performance in clean testing case is not what we want to see.

Looking at Table 5.15, we can also find that SME generalize well in test set b, which has noise unseen in the training set.

Table 5.13. Detailed test accuracies for MLE and SME using 10db SNR training data.

Word Acc	clean	20db	15db	10db	5db	0db	-5db	Avg.
MLE	86.42	95.78	93.63	87.98	72.95	38.99	6.56	77.87
SME (λ=50)	88.18	97.58	96.57	93.52	82.42	51.49	17.14	84.32
SME (λ=70)	88.16	97.59	96.63	93.69	82.64	51.84	17.87	84.48
SME (λ=80)	88.24	97.59	96.62	93.76	82.52	51.64	18.01	84.43
SME (λ=100)	88.19	97.63	96.71	93.77	82.47	51.43	18.04	84.40
SME (λ=200)	90.00	97.74	96.84	93.76	81.95	50.07	17.84	84.07

Table 5.14. Relative WER reductions for SME from MLE baseline using 10db SNR training data.

Rel. WER red.	clean	20db	15db	10db	5db	0db	-5db	Avg.
SME (λ=50)	12.96%	42.65%	46.15%	46.09%	35.01%	20.94%	11.32%	29.15%
SME (λ=70)	12.81%	42.89%	47.10%	47.50%	35.82%	21.06%	12.10%	29.87%
SME (λ=80)	13.40%	42.89%	46.94%	48.09%	35.38%	20.73%	12.25%	29.64%
SME (λ=100)	13.03%	43.84%	48.35%	48.17%	35.19%	20.39%	12.29%	29.51%
SME (λ=200)	26.36%	46.45%	50.39%	48.09%	33.27%	18.16%	12.07%	28.02%

Table 5.15. Detailed accuracies on testing set a, b, and c for SME with different balance coefficient λ using 10db SNR training data.

Word Acc	set a	set b	set c	Avg.
SME (λ=50)	88.22	82.22	80.69	84.32
SME (λ=70)	88.19	82.75	80.50	84.48
SME (λ=80)	87.99	83.03	80.11	84.43
SME (λ=100)	87.91	83.01	80.16	84.40
SME (λ=200)	87.51	83.01	79.32	84.07

Table 5.16. Detailed test accuracies for MLE and SME using 5db SNR training data.

Word Acc	clean	20db	15db	10db	5db	0db	-5db	Avg.
MLE	71.79	93.22	91.44	85.36	71.84	43.90	9.17	77.15
SME (λ=50)	71.75	93.70	93.79	91.31	82.45	58.08	21.35	83.87
SME (λ=100)	67.67	94.17	94.38	91.99	82.76	57.69	21.05	84.20
SME (λ=200)	65.92	94.40	94.69	92.33	83.09	57.58	21.09	84.42
SME (λ=500)	63.80	94.13	94.63	92.50	83.41	57.93	21.57	84.52
SME (λ=700)	63.80	94.12	94.64	92.51	83.42	57.91	21.58	84.52

Table 5.17. Relative WER reductions for SME from MLE baseline using 5db SNR training data.

Rel. WER red.	clean	20db	15db	10db	5db	0db	-5db	Avg.
SME (λ=50)	-0.14%	7.08%	27.45%	40.64%	37.68%	25.28%	13.41%	29.41%
SME (λ=100)	-14.60%	14.01%	34.35%	45.29%	38.78%	24.58%	13.08%	30.85%
SME (λ=200)	-20.81%	17.40%	37.97%	47.61%	39.95%	24.39%	13.12%	31.82%
SME (λ=500)	-28.32%	13.42%	37.27%	48.77%	41.09%	25.01%	13.65%	32.25%
SME (λ=700)	-28.32%	13.27%	37.38%	48.84%	41.12%	24.97%	13.66%	32.25%

5.3.4 5db SNR Training Condition

Tables 5.16, 5.17, and 5.18 give detailed test accuracies, relative WER reductions, and detailed accuracies on test set a, b, and c using the 5db SNR training data. SME with λ=50, 100, 200, 500, and 700 are tested. All SME options get 29%-32% relative WER reductions, with around 84% word accuracies. Although the average 84% word accuracy still looks good (compared to 72% word accuracy got by SME with the clean-trained model), the accuracy on the clean testing condition is already unacceptable. In the first time, SME decreased the word accuracy in the clean testing condition. This shows the generalization for mismatched conditions is not symmetric. Although clean-trained model can be generalized well to the -5db SNR testing case with SME, it is not true for the 5db-SNR-trained model to be generalized to the clean testing case with SME. A possible reason is that the utterance with 5db SNR has already been distorted a lot. Even with matched testing case, it can only get around 72% word accuracy with MLE model. This low quality model cannot be generated well with SME.

Table 5.18. Detailed accuracies on testing set a, b, and c for SME with different balance coefficient λ using 5db SNR training data.

Word Acc	set a	set b	set c	Avg.
SME (λ=50)	88.20	80.54	81.85	83.87
SME (λ=100)	88.44	81.30	81.51	84.20
SME (λ=200)	88.50	82.00	81.09	84.42
SME (λ=500)	88.49	82.54	80.54	84.52
SME (λ=700)	88.48	82.55	80.54	84.52

Table 5.19. Detailed test accuracies for MLE and SME using 0db SNR training data.

Word Acc	clean	20db	15db	10db	5db	0db	-5db	Avg.
MLE	35.08	79.67	81.07	76.46	63.54	39.52	6.86	68.05
SME (λ=50)	27.85	79.40	85.76	83.96	74.05	51.05	17.67	74.84
SME (λ=100)	27.77	78.32	85.52	84.59	75.09	51.98	18.38	75.10
SME (λ=200)	27.63	78.08	85.38	84.83	75.47	54.24	18.49	75.20
SME (λ=300)	27.63	78.06	85.38	84.82	75.47	52.25	18.53	75.19
SME (λ=500)	27.60	78.01	85.35	84.79	75.48	52.23	18.55	75.17

5.3.5 0db SNR Training Condition

Tables 5.19, 5.20, and 5.21 give detailed test accuracies, relative WER reductions, and detailed accuracies on test set a, b, and c using the 0db SNR training data. SME with λ=50, 100, 200, 300, and 500 are tested. All SME options get about 22% relative WER reductions, with around 75% word accuracies. In this case, the original model quality is bad, with only 40% word accuracy for MLE model on the matched testing case (0db testing). This bad quality model severely affect the performance of SME. In fact, with such a bad performance, the major issue of training is how to improve the accuracy for the matched testing case. Generalization is only a minor issue now, although SME still gets more than 30% relative WER reductions for the 10db and 5db SNR testing case, and more than 20% relative WER reduction for the 15db SNR testing scenario.

5.4 Conclusion

We have evaluated the generalization issues of SME and MCE in this study. Multi-condition testing with both clean and multi-condition training is investigated on the Aurora2 task. In the clean training case, SME achieves an overall average of 29% relative WER reductions

Table 5.20. Relative WER reductions for SME from MLE baseline using 0db SNR training data.

Rel. WER red.	clean	20db	15db	10db	5db	0db	-5db	Avg.
SME (λ=50)	-11.14%	-1.33%	24.78%	31.86%	28.83%	19.06%	11.61%	21.25%
SME (λ=100)	-11.26%	-6.64%	23.51%	34.54%	31.68%	20.60%	12.37%	22.07%
SME (λ=200)	-11.48%	-7.82%	22.77%	35.56%	32.72%	24.34%	12.49%	22.38%
SME (λ=300)	-11.48%	-7.92%	22.77%	35.51%	32.72%	21.05%	12.53%	22.35%
SME (λ=500)	-11.52%	-8.17%	22.61%	35.39%	32.75%	21.02%	12.55%	22.28%

Table 5.21. Detailed accuracies on testing set a, b, and c for SME with different balance coefficient λ using 0db SNR training data.

Word Acc	set a	set b	set c	Avg.
SME (λ=50)	79.95	69.27	75.77	74.84
SME (λ=100)	80.17	69.86	75.44	75.10
SME (λ=200)	80.26	70.11	75.26	75.20
SME (λ=300)	80.26	70.08	75.28	75.19
SME (λ=500)	80.23	70.10	75.21	75.17

while MCE gets less than 1% relative WER reductions. Although both methods perform similarly when testing with clean utterances, SME outperforms MCE significantly in the testing utterances with SNRs ranging from -5db to 20db. In those mismatched conditions, the margin in SME contributes to classifier generalization and results in great performance improvements for SME. In multi-condition training, SME is slightly better than MCE since in this case the focus of classifier learning is more on minimizing the empirical risk instead of maximizing the margin for generalization. We hope the observations in this study can further deepen the research of generalization property of margin-based classification methods.

In this chapter, we also comprehensively worked on the single SNR training case. The training sets with 20db, 15db, 10db, 5db, and 0db SNRs were created. Only with single SNR training level, some SME options still can get 30% relative WER reductions, with 84% word accuracy on the test set. Since the test is still on mismatched conditions, the performance clearly demonstrates SME's nice property of generalization. The word accuracy of 84% is close to 87% accuracy in the multi-condition training case. Therefore, single SNR level training may be an option to improve the accuracy in robust ASR applications. We also need to pay attention with the trained model qualities from the distorted training set. Because the training signal was distorted by the additive noise, we cannot expect the matched test case can get a 99% word accuracy as what have been obtained in the clean-trained clean-tested case. Therefore, SME has to focus on both the empirical risk minimization and the generalization.

This chapter only presents our initial study, we are now working on a number of related research issues. First, current evaluation of these DT methods is on Aurora2, which is a connected-digit task. We may extend the evaluation to a larger task, such as Aurora4 [95]. Second, SME may be combined with other robust ASR methods as in [123] to further improve ASR performance.

CHAPTER 6

THE RELATIONSHIP BETWEEN MARGIN AND HMM PARAMETERS

From our study it is clear that the margin parameter is related to the discrimination power of the classification models. All the margin-based automatic speech recognition (ASR) methods implicitly address the generalization issue by claiming that they are using a margin. This quantity is often specified as a numeric variable and determined empirically. large margin estimation (LME) sets the margin as a variable, and tries to optimize it together with the hidden Markov model (HMM) parameters. large margin hidden Markov models (LM-HMMs) directly set it to 1 and use the summation of the traces of all the precision matrices as a regularization term. soft margin estimation (SME) also treats the margin as a variable. The issues of how the margin is related to the HMM parameters and how it directly characterizes the generalization ability of HMM-based classifiers have not been addressed so far in the literature. This is in clear contrast with the case of support vector machines (SVMs) in which the margin is clearly expressed as a function of model parameters.

This study investigates the above-mentioned issue. By making a one-to-one mapping between the objective functions of SVMs and SME, we show that the margin can be expressed as a function of HMM parameters. A divergence-based margin is then proposed to characterize the generalization ability of HMM-based classifiers. It can then be plugged into the SME-based objective functions for simultaneous optimization of the margin and HMM parameters. Tested on the TIDIGITS task, the proposed SME method with model-based margin performs similarly to the original SME method, and may be with better theoretic justifications.

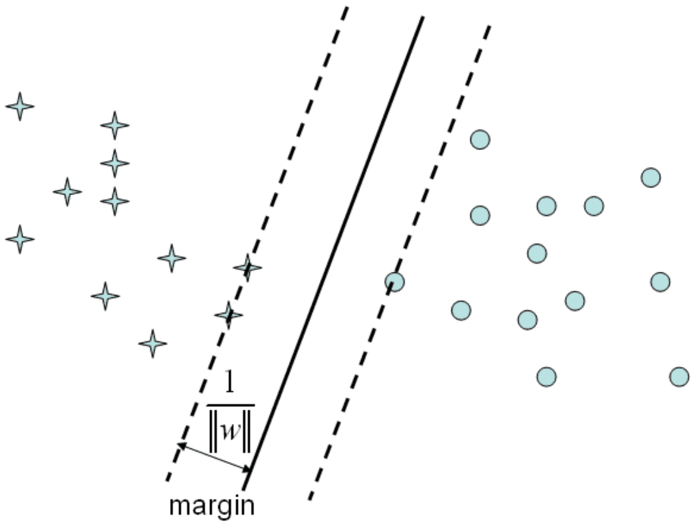

Figure 6.1. A binary separable case of SVMs.

6.1 Mapping between SME and SVMs

Figure 6.1 shows the separable case of binary SVMs where there exist a projection w and an offset b such that $y_i(wxi + b) > 0$ for all training samples (x_1, y_1),, (x_n, y_n). In this situation, SVMs solve the following optimization problem:

$$\max_{w} \frac{1}{||w||}, \text{ subject to } y_i(x_i \frac{w}{||w||} + \frac{b}{||w||}) - \frac{1}{||w||} > 0, \tag{6.1}$$

where $||w||$ is the Euclidean norm of w, and $1/||w||$ is referred as the margin. With this optimization objective, every mapped sample is at least a distance of $1/||w||$ from the decision boundary. If the mismatch between the training and testing sets only causes a shift less than this margin in the projected space, a correct decision can still be made. The margin, $1/||w||$, can be considered as a measure to characterize the generalization property for the SVMs.

In Figure 6.1, for the positive and negative classes, the corresponding hyper-planes (dashed lines) can be viewed as the models with parameters w and b ($\Lambda = (w, b)$), satisfying $(wx + b) = +1$. $2/||w||$ (the margin times 2) is considered as a separation distance, $D(\Lambda)$, for these two models. This view can be extended to defining the optimization target of non-separable SVMs as a combination of empirical risk minimization and model separation (or margin) maximization by minimizing:

$$\lambda||w|| + \frac{1}{N}\sum_i \left[\frac{1}{||w||} - y_i(x_i\frac{w}{||w||} + \frac{b}{||w||})\right]_+ \tag{6.2}$$

where λ is a balance coefficient. Note that in most literatures of SVMs (e.g., [108]), there is a scale of $||w||$ different from Eq. (6.2). That is to make its objective function as a second-order function to optimize. With margin $\rho = 1/||w||$, Eq. (6.2) can be rewritten as

$$\frac{\lambda}{\rho(\Lambda)} + \frac{1}{N}\sum_i \left[\rho(\Lambda) - y_i(x_i\frac{w}{||w||} + \frac{b}{||w||})\right]_+ . \tag{6.3}$$

If we denote

$$d(O_i, \Lambda) = y_i(x_i\frac{w}{||w||} + \frac{b}{||w||}), \tag{6.4}$$

then Eq. (6.3) of SVMs has an almost one-to-one mapping with Eq. (6.5) of SME:

$$L^{SME}(\rho, \Lambda) = \frac{\lambda}{\rho} + \frac{1}{N}\sum_{i=1}^{N}(\rho - d(O_i, \Lambda))_+. \tag{6.5}$$

6.2 SME with Further Generalization

By carefully comparing Eq. (6.5) with Eq. (6.3), we can see there is one difference between them. In Eq. (6.5), the margin ρ is a numeric variable, independent with the system parameter, Λ. In contrast, in Eq. (6.3) the margin is a function of the system parameter, Λ, denoted as $\rho(\Lambda)$. As discussed in Section 6.1, the margin of SVMs can be considered as a half of the system distance:

$$\rho(\Lambda) = D(\Lambda)/2. \tag{6.6}$$

After determined by the optimization process of SVMs, this margin only depends on parameter, $\Lambda = (w, b)$.

The margin in margin-based ASR methods is in different case. Given system parameters, Λ, the margin cannot be computed out because it is a single variable got from data. For better generalization, it is desirable to express the margin as a function of the HMM parameters. As shown above, the margin, ρ, can be expressed as a function of the model distance in the SVM system. In the following, we wish to express ρ in SME as a function of the system model distance of HMMs.

For an HMM system, every state is modeled by a Gaussian mixture model (GMM). Hence, the first step is to get the model distance of GMMs. The symmetric Kullback-Leibler divergence is a well-known measure for comparing two densities [23]:

$$D(k,l) = E\left\{-log\frac{p_k(x)}{p_l(x)}|w_l\right\} - E\left\{-log\frac{p_k(x)}{p_l(x)}|w_k\right\} \tag{6.7}$$

where $p_k(x)$ and $p_l(x)$ are the probability density functions of the two models, w_k and w_l. Eq. (6.7) has a closed form expression for Gaussian densities [23]:

$$D_G(k,l) = \frac{1}{2}tr\left\{(\Sigma_k^{-1} + \Sigma_l^{-1})(\mu_k - \mu_l)(\mu_k - \mu_l)^T\right\} + \frac{1}{2}tr\left\{\Sigma_k^{-1}\Sigma_l + \Sigma_k\Sigma_l^{-1} - 2I\right\} \tag{6.8}$$

where (μ_k, Σ_k) and (μ_l, Σ_l) are the mean vectors and covariance matrices of Gaussian densities, k and l. I is an identity matrix. For GMMs, the following approximation [40] is made for the divergence of the ith and jth GMMs:

$$D_{GMM}(i,j) = \sum_{k=1}^{M_i}\sum_{l=1}^{M_j} c_{ik}c_{jl}D_G(ik, jl), \tag{6.9}$$

where ik and jl indicate the kth and lth Gaussians of the ith and jth GMMs. This approximation weights all the pairwise Gaussian components from GMMs with the corresponding mixture weights (c_{ik} and c_{jl}) and sums them together.

Suppose there are a total of NG GMMs in the system, Eq. (6.9) is used to compute all the pairwise divergences of GMMs in the system: $\{D_{GMM}(i,j), i = 1...NG, j = 1...NG, i \neq j\}$. Next we need to define the system model distance for the whole set of GMMs. Keeping in mind that the system model distance is used for generalization, the definition of system model distance of GMMs should be an indicator for better generalization. The bigger the system model distance, the less confusion the system will have.

Next we define the system divergence as the model distance for multiple GMMs as:

$$D(\Lambda) = \frac{1}{NG}\sum_i D_{GMM}(i, nearest(i)). \tag{6.10}$$

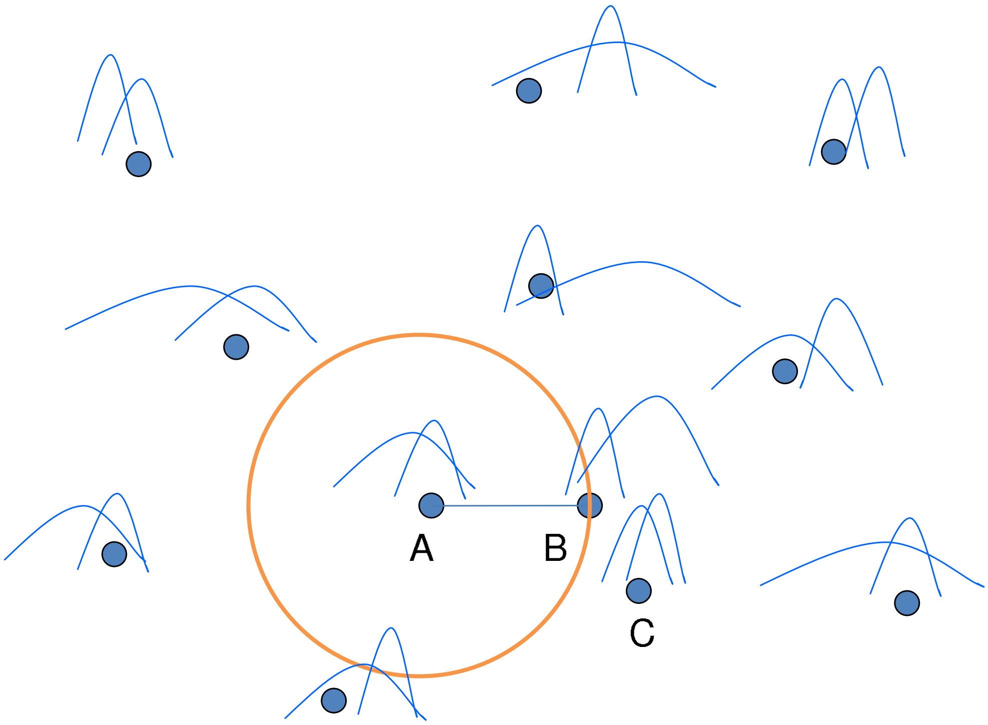

Figure 6.2. Divergence computation for HMM/GMM systems.

As Figure 6.2, for the ith GMM, only its nearest GMM ($nearest(i)$) is considered to significantly contribute the value of the system divergence. The divergence of two GMMs far apart from each other provides little information to quantify the system confusion, or system generalization. For HMM systems, every state is modeled by a GMM. Hence, the system divergence of HMMs can also be defined in a similar way to Eq. (6.10) by considering all the state GMMs. The only difference is that GMMs belonging to the same speech unit cannot be included in defining $nearest(i)$. Special attention should be paid if the idea is applied to a triphone- (or even quinphone-) ASR system. Because the triphone states are very similar for triphones sharing the same center phone, the candidates in $nearest(i)$ only considers the triphone states with different center phone.

To overcome the weakness of original SME, margin ρ may be replaced with system divergence, which is a function of HMM parameters and characterizes the system generalization. Also, margin ρ is compared with average log likelihood ratio (LLR) in Eq. (6.5) while divergence is the expectation of LLR. However, D in Eq. (6.10) is not with the same scale as the margin in our original SME. The reason is stated in the following. In Eq. (6.5), margin ρ is compared with separation measure d, which is the average LLR. This LLR is computed by using the correct and most competing models in an utterance. For different frames, the most competing models differ, resulting relatively small average LLR. In contrast, the divergence of two models uses all the samples in space for computation. This may result in a bigger value for the expectation of LLR. Until now, we cannot find in theory the exact relationship between system divergence in Eq. (6.10) and the margin. Instead, we observed that the square root of the divergence in Eq. (6.10) is similar to the margin used in our original SME work, which will be shown in Table 6.2. Therefore, we set:

$$D(\Lambda) = \left\{ \frac{1}{NG} \sum_i D_{GMM}(i, nearest(i)) \right\}^{1/2}. \tag{6.11}$$

In Eq. (6.11), D is summed over $D_{GMM}(i, nearest(i))$, which is not symmetric because the nearest neighbor is not symmetric. For example, in Figure 6.2, GMM A's nearest neighbor is GMM B. But GMM B's nearest neighbor is GMM C instead of GMM A.

Embed the margin in Eq. (6.11) into the SME framework and get:

$$\begin{aligned} L^{SME}(\Lambda) &= \frac{\lambda}{\rho(\Lambda)} + \frac{1}{N} \sum_{i=1}^{N} (\rho(\Lambda) - d(O_i, \Lambda))_+ \\ &= \frac{\lambda}{\rho(\Lambda)} + \frac{1}{N} \sum_{i=1}^{N} (\rho(\Lambda) - d(O_i, \Lambda)) I(\rho(\Lambda) - d(O_i, \Lambda) > 0) \\ &= \frac{\lambda}{\rho(\Lambda)} + \frac{1}{N} \sum_{i=1}^{N} (\rho(\Lambda) - d(O_i, \Lambda)) \frac{1}{1 + exp(-\gamma(\rho(\Lambda) - d(O_i, \Lambda)))}. \end{aligned} \tag{6.12}$$

The last equation of Eq. (6.12) is got by smoothing the indicator function with a sigmoid function. Eq. (6.12) is a smoothing function of HMM parameters and can be solved by using generalized probabilistic descent (GPD) [50] algorithm.

It should be noted that although the empirical approximation of margin is not precise, it can still work well under the framework of SME. As opposed to the separable case, there is no unique soft margin value for the cases of inseparable classification. The final soft margin value is affected by the choice of the balance coefficient, λ. In essence, Eq. (6.12) works well for generalization in two parts. The first is to pull the samples away from the decision boundary with a distance of margin by reducing the empirical risk. The second is to make this margin as a function of the system model distance, and then maximize it by minimizing the objective function in Eq. (6.12).

6.2.1 Derivative Computation

In the following, we get the derivative of Eq. (6.12) w.r.t. parameter Λ for the use in the GPD algorithm.

$$\begin{aligned}\frac{\partial L^{SME}(\Lambda)}{\partial\Lambda} &= -\frac{\lambda}{\rho^2(\Lambda)}\frac{\partial(\rho(\Lambda))}{\partial\Lambda} + \frac{1}{N}\sum_{i=1}^{N}\frac{\partial\left\{(\rho(\Lambda)-d(O_i,\Lambda))\frac{1}{1+exp(-\gamma(\rho(\Lambda)-d(O_i,\Lambda)))}\right\}}{\partial\Lambda} \\ &= -\frac{\lambda}{\rho^2(\Lambda)}\frac{\partial(\rho(\Lambda))}{\partial\Lambda} + \frac{1}{N}\sum_{i=1}^{N}\left\{\begin{array}{c}\frac{\partial(\rho(\Lambda)-d(O_i,\Lambda))}{\partial\Lambda}\frac{1}{1+exp(-\gamma(\rho(\Lambda)-d(O_i,\Lambda)))} \\ +(\rho(\Lambda)-d(O_i,\Lambda))\frac{\partial\frac{1}{1+exp(-\gamma(\rho(\Lambda)-d(O_i,\Lambda)))}}{\partial\Lambda}\end{array}\right\}. \end{aligned} \tag{6.13}$$

Since

$$\begin{aligned}&\frac{\partial 1/[1+exp(-\gamma(\rho-d(O_i,\Lambda)))]}{\partial\Lambda} \\ &= -\left\{\frac{1}{1+exp(-\gamma(\rho-d(O_i,\Lambda)))}\right\}^2 exp[-\gamma(\rho-d(O_i,\Lambda))](-\gamma)\frac{\partial(\rho-d(O_i,\Lambda))}{\partial\Lambda} \\ &= \gamma\left\{1-\frac{1}{1+exp(-\gamma(\rho-d(O_i,\Lambda)))}\right\}\frac{1}{1+exp(-\gamma(\rho-d(O_i,\Lambda)))}\frac{\partial(\rho-d(O_i,\Lambda))}{\partial\Lambda},\end{aligned} \tag{6.14}$$

we can rewrite Eq. (6.13) as

$$\frac{\partial L^{SME}(\Lambda)}{\partial\Lambda} = -\frac{\lambda}{\rho^2(\Lambda)}\frac{\partial(\rho(\Lambda))}{\partial\Lambda} + \frac{1}{N}\sum_{i=1}^{N}\left\{\begin{array}{c}\frac{\partial(\rho(\Lambda)-d(O_i,\Lambda))}{\partial\Lambda}\frac{1}{1+exp(-\gamma(\rho(\Lambda)-d(O_i,\Lambda)))} \\ \cdot\left(1+\gamma(\rho(\Lambda)-d(O_i,\Lambda))\left\{1-\frac{1}{1+exp(-\gamma(\rho(\Lambda)-d(O_i,\Lambda)))}\right\}\right)\end{array}\right\}. \tag{6.15}$$

Let

$$l = 1/[1+exp(-\gamma(\rho-d(O_i,\Lambda)))], \tag{6.16}$$

then Eq. (6.15) can be rewritten as

$$\begin{aligned}\frac{\partial L^{SME}(\Lambda)}{\partial\Lambda} &= -\frac{\lambda}{\rho^2(\Lambda)}\frac{\partial(\rho(\Lambda))}{\partial\Lambda} + \frac{1}{N}\sum_{i=1}^{N}\left\{\frac{\partial(\rho(\Lambda)-d(O_i,\Lambda))}{\partial\Lambda}l(1+\gamma(\rho(\Lambda)-d(O_i,\Lambda))(1-l))\right\} \\ &= \left\{\begin{array}{c}\left[-\frac{\lambda}{\rho^2(\Lambda)}+\frac{1}{N}\sum_{i=1}^{N}\{l(1+\gamma(\rho(\Lambda)-d(O_i,\Lambda))(1-l))\}\right]\frac{\partial(\rho(\Lambda))}{\partial\Lambda} \\ +\frac{1}{N}\sum_{i=1}^{N}\{l(1+\gamma(\rho(\Lambda)-d(O_i,\Lambda))(1-l))\}\frac{\partial(-d(O_i,\Lambda))}{\partial\Lambda}\end{array}\right\}.\end{aligned} \tag{6.17}$$

Since $d(O_i, \Lambda)$ is a normalized LLR, $\frac{\partial(-d(O_i,\Lambda))}{\partial\Lambda}$ can be computed similarly to what has been done in MCE training. Please refer [48] for the detailed formulations of those derivatives.

In the following, we just give the formulation of $\frac{\partial(\rho(\Lambda))}{\partial\Lambda}$. From Eqs. (6.8), (6.9), and (6.11), we have

$$\begin{aligned}\frac{\partial(\rho(\Lambda))}{\partial\Lambda} &= 0.5(\frac{1}{NG}\sum D_{GMM})^{-1/2}\frac{\partial(\frac{1}{NG}\sum D_{GMM})}{\partial\Lambda} \\ &= \frac{0.5}{\rho(\Lambda)}\frac{\partial(\frac{1}{NG}\sum D_{GMM})}{\partial\Lambda},\end{aligned} \tag{6.18}$$

and

$$\frac{\partial D_{GMM}}{\partial\Lambda} = \frac{\partial\sum_i\sum_j c_i c_j D_G(i,j)}{\partial\Lambda} \tag{6.19}$$

Since $D_G(i, j)$ is symmetric for Gaussian components i and j, the following content only gives the derivative of $D_G(i, j)$ w.r.t. μ_j and σ_j. For the lth dimension of μ_j and σ_j, the preprocessing in MCE is also applied.

$$\mu_{jl} \rightarrow \tilde{\mu}_{jl} = \frac{\mu_{jl}}{\sigma_{jl}} \tag{6.20}$$

$$\sigma_{jl} \rightarrow \tilde{\sigma}_{jl} = log\sigma_{jl} \tag{6.21}$$

Finally, we have

$$\frac{\partial D_G(i,j)}{\partial\tilde{\mu}_{jl}} = (\mu_{jl} - \mu_{il})\left(\frac{1}{\sigma_{jl}^2} + \frac{1}{\sigma_{il}^2}\right)\sigma_{jl} \tag{6.22}$$

$$\frac{\partial D_G(i,j)}{\partial\tilde{\sigma}_{jl}} = -\frac{(\mu_{jl} - \mu_{il})^2}{\sigma_{jl}^2} + \frac{\sigma_{jl}^2}{\sigma_{il}^2} - \frac{\sigma_{il}^2}{\sigma_{jl}^2} \tag{6.23}$$

After getting all the derivatives, GPD is used to update all HMM parameters.

6.3 Comparison with Minimum Divergence Training

Recently, divergence was used to design a new DT method, minimum divergence training (MDT) [20]. The difference between divergence-based SME and MDT is discussed in the following aspects.

Table 6.1. Testing set string accuracy comparison with different methods.

	MLE	MCE	SME	SME_C
1-mix	95.20%	96.94%	98.76%	98.76%
2-mix	96.20%	97.40%	98.91%	98.95%
4-mix	97.80%	98.24%	99.15%	99.20%
8-mix	98.03%	98.66%	99.29%	99.29%
16-mix	98.36%	98.87%	99.29%	99.30%

- Objective: SME jointly maximizes the system divergence and minimize the empirical risk. MDT minimizes the divergence between the recognized model sequence and the correct model sequence in utterances.

- Computation: The divergence used in SME is symmetric while it is asymmetric in MDT. MDT uses the unscented transform mechanism for the computation and has high precision. In contrast, the divergence in SME for GMMs is simply computed by weighting combination of the divergence of Gaussians. The symmetric and simple approximation makes the divergence be a much concise form of GMM parameters.

- Usage: SME uses divergence globally, as a measure of system generalization. MDT uses divergence utterance by utterance, as a local measure of the dissimilarity between the competing models and correct models in utterance.

6.4 Experiments

The modified SME framework was evaluated on the TIDIGITS connected digit recognition task. The experiment setup is the same as that in previous sections.

Table 6.1 compares string accuracies of different training methods with various numbers of mixture components in each HMM state. The column labeled SME shows the accuracy of the SME method with divergence-based margin. SME outperforms MLE and MCE significantly. The last column of Table 6.1 with label SME_C lists the performance of the original SME method, which is similar to those of SME.

In Table 6.2 we list the square root values of the system divergence (Eq. (6.11)) of all the

Table 6.2. Square root of system divergence (Eq. (6.11)) with different methods.

	MLE	MCE	SME	SME_C
1-mix	3.7	5.1	5.6	5.0
2-mix	5.5	6.3	6.5	6.0
4-mix	6.8	7.0	7.2	7.5
8-mix	8.0	8.2	8.6	8.5
16-mix	9.0	9.1	10.0	11.0

models. The divergence trend of the three training methods is clear within the same model configuration. The divergence of MLE is the smallest and that of SME is the largest in all model configurations. System divergence is however not the only indicator of accuracy. For 1-mixture SME models, the string accuracy is 98.76%, which is better than that of the 16-mixture MLE models. But the system divergence of the 1-mixture SME models is far less than that of the 16-mixture MLE models. Hence, generalization is not only the factor that determines the recognition performance. The rightmost column of Table 6.2 with label SME_C lists the empirical margins used in the original SME method. It is easy to see that the divergence-based margins are similar in value to these empirical margins.

6.5 Conclusion

In this study, we extend the margin from a constant value in our previous work to a function of the system model distance. This distance-based margin is a function of all the model parameters and well characterizes the system generalization. A system divergence is defined as the model distance of GMMs or HMMs. A divergence-based margin is plugged into the objective function of SME. The modified SME achieved similar performance to that obtained with the previously proposed method, and with a better theoretic foundation. This facilitates rapid design without the need to try different constants for systems with different complexities. From the experiment, for each mixture model setting, the divergence follows the same trend as the accuracy. Meanwhile for different model configurations, a greater divergence does not result in a greater accuracy. This demonstrates better generalization results in better accuracy, but is not the only factor to determine the performance.

The work in this study goes beyond the margin definition in SVMs. In binary SVMs, the margin is easily viewed as the separation distance of the two competing classifiers. For multiple classifiers, there is no obvious extension. We directly relate the margin with model parameters to characterize the generalization issue. Although this work uses HMMs as classifiers, we believe this model-based margin can be generalized to benefit other multi-class margin-based methods.

As stated in this study, a square root operation is adopted to map the system divergence to the soft margin. Although satisfactory results were obtained, many efforts are needed to investigate what is the underlying theory and whether there are other functions to associate the system divergence with margin in SME. In [36], some precise realizations of divergence with heavy computation cost are discussed. In our study, the computation of divergence is simple and easy for optimization, but with some potential precision loss. We will investigate whether better precision modeling of divergence will bring out further improvements. In [67] we have shown that SME also works well on a large vocabulary continuous speech recognition (LVCSR) task. We will demonstrate the effectiveness of this divergence-based SME on a LVCSR task in future works.

CHAPTER 7

SME FOR LARGE VOCABULARY CONTINUOUS SPEECH RECOGNITION

Although margin-based methods have shown their superiority on small tasks, they have not well demonstrated the same power on large vocabulary continuous speech recognition (LVCSR) tasks. To be widely used, it is necessary to show convincing successes on LVCSR tasks. The research presented here will make a comprehensive study of SME for LVCSR tasks.

7.1 SME for LVCSR with the Generalized Probabilistic Descent Algorithm

The key issue for using SME in LVCSR tasks is to define appropriate model separation measures. One method is to use $d^{SME\_utter}(O_i, \Lambda)$ in Table 7.1, and solve HMM parameters by minimizing the quantity in Eq. (7.1).

$$\begin{aligned} L^{SME}(\rho, \Lambda) &= \frac{\lambda}{\rho} + \frac{1}{N}\sum_{i=1}^{N}(\rho - d(O_i, \Lambda))_+ \\ &= \frac{\lambda}{\rho} + \frac{1}{N}\sum_{i=1}^{N}(\rho - d(O_i, \Lambda))I(O_i \in U), \end{aligned} \tag{7.1}$$

where $d(O_i, \Lambda)$ is normalized log likelihood ratio with frame selection used in previous chapters. This separation measure only considers one competing string for an utterance. However, most successful discriminative training (DT) methods on LVCSR use lattices to get a rich set of competing candidates information. The advantage can also be explained by the test risk bound in Eq. (3.1) since lattices provide more confusion patterns (i.e., more data). This will result in a reduced generalization term, which makes the test risk bound tighter. In the following, two solutions are provided for lattice-based separation measure definitions for LVCSR.

The first solution is similar to lattice-based MMIE [118], [122], MCE [81], and MPE [101]. We define distances, $d^{SME\_MMIE}(O_i, \Lambda)$, $d^{SME\_MCE}(O_i, \Lambda)$, and $d^{SME\_MPE}(O_i, \Lambda)$, as

Table 7.1. Separation measure for SME

$d^{SME\_utter}(O_i, \Lambda)$	$\frac{1}{n_i} \sum_j log \left[\frac{P_\Lambda(O_{ij} \vert S_i)}{P_\Lambda(O_{ij} \vert \hat{S}_i)} \right] I(O_{ij} \in F_i)$
$d^{SME\_MMIE}(O_i, \Lambda)$	$log \frac{P_\Lambda(O_i \vert S_i) P(S_i)}{\sum_{\hat{S}_i} P_\Lambda(O_i \vert \hat{S}_i) P(\hat{S}_i)}$
$d^{SME\_MCE}(O_i, \Lambda)$	$1 - \frac{1}{1+exp(-\gamma d(O_i,\Lambda)+\theta)}$
$d^{SME\_MPE}(O_i, \Lambda)$	$\frac{\sum_{\hat{S}_i} P_\Lambda(O_i \vert \hat{S}_i) P(\hat{S}_i) RawPhoneAccuracy(\hat{S}_i)}{\sum_{\hat{S}_i} P_\Lambda(O_i \vert \hat{S}_i) P(\hat{S}_i)}$

shown in Table 7.1. We can now take advantage of optimization algorithms adopted in lattice-based DT methods to obtain statistics at the utterance level and then use extended Baum-Welch algorithms [99] to optimize parameters. However, due to their focus on utterance-level competition, it is possible to lose the advantage of the frame-level discrimination power in the SME separation measures as discussed in previous chapters.

SME separations can also be defined at the word segment level [66]. The first step is to align the utterance with the correct transcription and get the timing information for every word. The second step is to find competing words for every word in the lattice. This is done by examining the lattice to get words falling into the time segment of current correctly transcribed words. A frame overlapping threshold is set not to consider words with too few overlapping frames as competing words. For example, for the lattice in Figure 7.1, the competing words are listed in Table 7.2. For the p-th overlapping word pair, we denote the number of overlapped frames as n_{op}, the j-th overlapping frame as O_{oj}, set of overlapping frames as F_{op}, and the target and competing words as W_{target} and W_{comp}. A word level separation can be defined as:

$$d_{op}^{SME\_word}(O_i, \Lambda) = \frac{1}{n_{op}} \sum_j log \left[\frac{P_\Lambda(O_{oj} \vert W_{target})}{P_\Lambda(O_{oj} \vert W_{comp})} \right] I(O_{oj} \in F_{op}). \tag{7.2}$$

where $P_\Lambda(O_{oj} \vert W_{target})$ and $P_\Lambda(O_{oj} \vert W_{comp})$ are the likelihood scores for W_{target} and W_{comp}.

For any word pair W_{target} and W_{comp}, we compute Eq. (7.2), and plug all of them into the following formula:

$$L^{SME}(\rho, \Lambda) = \frac{\lambda}{\rho} + \frac{1}{N} \sum_{i=1}^{N} \left[\frac{1}{np_i} \sum_{p=1}^{np_i} (\rho - d_{op}^{SME\_word}(O_i, \Lambda))_+ \right], \tag{7.3}$$

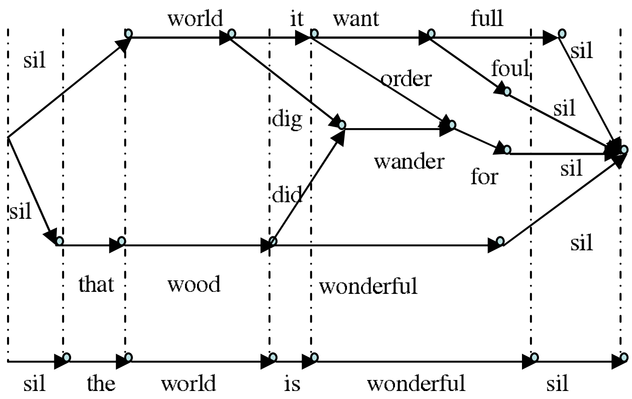

Figure 7.1. Lattice example: the top lattice is obtained in decoding, and the bottom is the corresponding utterance transcription.

where np_i denotes the number of overlapping word pairs in utterance O_i. Then, the GPD algorithm is used to update HMM parameters.

Table 7.2. Correct and competing words for lattice example

Correct word	Competing words
the	that
world	wood, it , dig
is	it, dig, did, wonderful
wonderful	want, full, foul, order, dig, did, wander, for

We found the word level separation ($d_{op}^{SME\_word}(O_i, \Lambda)$ with word pairs in lattices) to be better than the utterance-level measure ($d^{SME\_utter}(O_i, \Lambda)$ with only the correct and most competitive strings), because it uses more confusion patterns. For SME, $d_{op}^{SME\_word}(O_i, \Lambda)$ may also have an advantage over other separation measures defined above, which have only one value for each utterance. This is because in SME we will plug this separation value into Eq. (7.1), and the utterances with values greater than the value of the margin will not contribute to parameter optimization. However in some cases, there may be some word pairs in lattices that still have distances less than the value of the margin. The word level separation measure $d_{op}^{SME\_word}(O_i, \Lambda)$ makes use of those word pairs to get more confusion patterns.

7.1.1 Experiment of SME with GPD

We used the 5k-WSJ0 [97] task to evaluate the effectiveness of SME on LVCSR. The training set is the SI-84 set, with 7077 utterances from 84 speakers. All testing is conducted on the Nov92 evaluation set, with 330 utterances from 8 speakers. Baseline HMMs are trained with MLE using the HMM toolkit (HTK). The HMMs are cross-word triphone models. There were 2818 shared states obtained with a decision tree and each state observation density is modeled by an 8-mixture Gaussian mixture model. The input features were 12MFCCs + energy, and their first and second order time derivatives. A trigram LM within the 5k-WSJ0 corpus was used for decoding. The baseline WER was 5.06% for MLE models.

The bigram LM was used to obtain seed lattices for all of the training utterances. These lattices were generated only once. At each iteration, the recently updated HMMs were incorporated to generate new lattices by using seed lattices as decoding word graphs. Following this, SME was used to update HMM parameters. The method, denoted by SME_word, is based on the word level separation measure, $d_{op}^{SME\_word}(O_i, \Lambda)$, defined in Eq. (7.2).

MCE model was trained with the similar implementation as [81]. The bigram LM was used to generate lattices and the unigram LM was used to rescore them. The correct path was removed from the decoded word graph, and the smoothing constant was set to 0.04 as in [81]. The relative WER reduction of this MCE realization is similar to what has been reported in [81].

In Table 7.3, the WERs obtained with MLE, MCE and the SME method are compared. The SME method achieved lower WERs than those obtained with the MLE and MCE models. SME_word decreased WERs significantly from MLE, with 12% relative WER reductions.

Table 7.3. Performance on the 5k-WSJ0 task

	WER	Relative Improvement
MLE	5.06%	-
MCE	4.60%	9%
SME_word	4.43%	12%

7.2 SME for LVCSR with the Extended Baum-Welch Algorithm

In the last section, SME was shown to work well on the 5k-WSJ0 task. However, two potential areas for improvement need to be addressed. The first is that the generalized probabilistic descent algorithm was used for HMM parameter optimization. Although it is easy to work in a small task, we had a hard time getting suitable step sizes on LVCSR tasks. The second is that SME improves over models initialized with maximum likelihood estimation (MLE), but fails to demonstrate its advantage over conventional DT models in the same experimental configuration.

The first part of this study [67] addresses the two above-mentioned issues. For optimization, the extended Baum-Welch (EBW) [91] algorithm is adopted to update HMM parameters with statistics obtained from lattices. For comparison, MCE will be compared fairly with SME by sharing most of the implementation details. More conventional DT methods will be compared in later sections. As shown latter, the proposed SME modification, with utterance and frame selection using EBW optimization, performs better than MCE and MLE. Above all SME with frame selection works better than SME with utterance selection. Based on the framework of SME with frame selection, we unify the SME work with different separation (string, word, and phone) level. The best SME model achieves a relative word error rate (WER) reduction of 25% from our MLE baseline.

Let's revisit the formulation of SME. There are two targets for optimization: one is to minimize the empirical risk, and the other is to maximize the margin. These two targets are combined into a single SME objective function for minimization:

$$L^{SME}(\rho, \Lambda) = \frac{\lambda}{\rho} + R_{emp}(\Lambda) = \frac{\lambda}{\rho} + \frac{1}{N}\sum_{i=1}^{N} l(O_i, \Lambda), \tag{7.4}$$

where ρ is the soft margin, and λ is a coefficient to balance the soft margin maximization and the empirical risk minimization. $l(O_i, \Lambda)$ is a loss function for utterance O_i, Λ denotes the set of HMM parameters, and N is the number of training utterances.

As discussed in [69], there is a mapping relationship between λ and ρ. For a fixed λ, there is one corresponding ρ. Instead of choosing a fixed λ and trying to get the solution of Eq. (7.4), we can directly choose a ρ in advance. The key component of SME is a proper definition of the loss function, $l(O_i, \Lambda)$. This loss should be related to the margin, ρ.

In the original formulation of SME, margin is used for utterance selection with a hinge loss function. This usage will be discussed in Section 7.2.1. As an extension, margin-based frame selection will be discussed in Section 7.2.2. Margin-based utterance and frame selection allow the loss in Eq. (7.4) to focus on samples important to model separation instead of using all the samples.

7.2.1 SME with Utterance Selection

SME with utterance selection is formularized as:

$$l(O_i, \Lambda) = (\rho - d(O_i, \Lambda))I(O_i \in U). \tag{7.5}$$

$I(.)$ is an indicator function and selects the utterances that are in set U for the contribution of the empirical risk. $d(O_i, \Lambda)$ is the separation measure between the correct and competing candidates for O_i.

In the original formulation of SME, this utterance selection is realized via a hinge function as

$$\begin{aligned} l(O_i, \Lambda) &= (\rho - d(O_i, \Lambda))_+ \\ &= \begin{cases} \rho - d(O_i, \Lambda), & \text{if } \rho - d(O_i, \Lambda) > 0 \\ 0, & otherwise \end{cases} . \end{aligned} \tag{7.6}$$

This means that if the separation measure $d(O_i, \Lambda)$ is greater than the margin, this utterance is good enough and the parameters inside it need not update. Otherwise, this utterance causes some losses and contributes to the empirical risk computation of Eq. (7.4).

Outlier utterances may cause trouble for model training. Hence, another loss function for utterance selection is defined as:

$$l(O_i, \Lambda) = \begin{cases} \rho - d(O_i, \Lambda), & \text{if } \rho - d(O_i, \Lambda) > \tau \\ 0, & otherwise \end{cases} . \tag{7.7}$$

where the hinge loss function in Eq. (7.6) is modified to have an additional threshold. The utterance that has too small value of $d(O_i, \Lambda)$ will not contribute to the loss computation because it can be an outlier.

Now, it is critical to define the separation measure $d(O_i, \Lambda)$. To make a fair comparison between SME and MCE, the following separation measure is defined:

$$d(O_i, \Lambda) = log \frac{P_\Lambda(O_i|S_i)P(S_i)}{\sum_{\hat{S}_i \neq S_i \bigwedge \hat{S}_i \in G_i} P_\Lambda(O_i|\hat{S}_i)P(\hat{S}_i)} \tag{7.8}$$

where G_i is a decoded lattice, S_i is the correct transcription for utterance O_i, and $\hat{S}_i$ denotes the transcription of words in the decoded lattice, G_i. The quantity in Eq. (7.8) measures the separation between the correct path and competing paths. In MCE, Eq. (7.8) is further embedded into a sigmoid function as in [81]. That sigmoid function can also be considered as an utterance selection function. This is because the sentences with sigmoid values close to 0 or 1 have their derivatives near 0 and will not contribute to parameter update. SME does not use a sigmoid function because there is already an utterance selection item $I(O_i \in U)$ in Eq. (7.5).

By plugging Eq. (7.8) into Eq. (7.6) or Eq. (7.7) to compute the loss in Eq. (7.4), SME with utterance selection is realized.

7.2.2 SME with Frame Selection

Since the utterances that are not selected in Eq. (7.6) or Eq. (7.7) may still have key local discriminative information from individual frames, SME with frame selection is proposed as:

$$l(O_i, \Lambda) = \sum_j l(O_{ij}, \Lambda) = \sum_j (\rho - d(O_{ij}, \Lambda))I(O_{ij} \in F_i), \tag{7.9}$$

where O_{ij} is the jth frame for utterance O_i, and F_i is the frame set in which the frames contribute to the loss computation. SME now selects the frames that are critical to discriminative separation. We realize it with the frame posterior probability via computing the posterior probability for a word w (in the correct transcription S_i) with starting time t_{ws} and ending time t_{we}, which is got by summing the probabilities of all the lattice paths, R, in which w lies in:

$$p(w|t_{ws}, t_{we}, O_i) = \sum_{R \in G_i \bigwedge (w|t_{ws}, t_{we}) \in R \bigwedge (w|t_{ws}, t_{we}) \in S_i} \frac{P_\Lambda(O_i|R)P(R)}{\sum_{\hat{S}_i \in G_i} P_\Lambda(O_i|\hat{S}_i)P(\hat{S}_i)} \tag{7.10}$$

The frame posterior probability is then computed by summing the posterior probabilities of all the correct words that pass time j:

$$\begin{aligned} q(O_{ij}) &= \sum_{w|t_{ws} \leq j \leq t_{we}} p(w|t_{ws}, t_{we}, O_i) \\ &= \sum_{w|t_{ws} \leq j \leq t_{we}} \sum_{R \in G_i \bigwedge (w|t_{ws}, t_{we}) \in R \bigwedge (w|t_{ws}, t_{we}) \in S_i} \frac{P_\Lambda(O_i|R)P(R)}{\sum_{\hat{S}_i \in G_i} P_\Lambda(O_i|\hat{S}_i)P(\hat{S}_i)} \end{aligned} \tag{7.11}$$

Frame selection for SME is done by comparing the frame posterior probability with the margin ρ. Using similar selection styles as in Eqs. (7.6) and (7.7), the frame loss function has the following form:

$$l(O_{ij}, \Lambda) = \begin{cases} \rho - d(O_{ij}, \Lambda), & \text{if } \rho - q(O_{ij}) > 0 \\ 0, & otherwise \end{cases}. \tag{7.12}$$

or

$$l(O_{ij}, \Lambda) = \begin{cases} \rho - d(O_{ij}, \Lambda), & \text{if } \rho > q(O_{ij}) > \tau \\ 0, & otherwise \end{cases}. \tag{7.13}$$

Eqs. (7.12) and (7.13) select the frames that are critical for parameter updating. Eq. (7.12) focuses on confusion patterns and ignores good samples. Eq. (7.13) works on confusion patterns and removes the influence of noisy frames with too small posterior probabilities because they may be unreliable for parameter update due to wrong time alignment. As what will be demonstrated in the experiments section, Eq. (7.13) is critical for the success of frame-based SME.

The last step is to define frame level separation measure with similar computational steps as Eq. (7.11):

$$d(O_{ij}, \Lambda) = log \sum_{w|t_{ws} \leq j \leq t_{we}} \sum_{R \in G_i \bigwedge (w|t_{ws}, t_{we}) \in R \bigwedge (w|t_{ws}, t_{we}) \in S_i} \frac{P_\Lambda(O_i|R)P(R)}{\sum_{\hat{S}_i \neq S_i \bigwedge \hat{S}_i \in G_i} P_\Lambda(O_i|\hat{S}_i)P(\hat{S}_i)} \tag{7.14}$$

Similar to Eq. (7.8), the correct transcription is removed from the denominator in Eq. (7.14) because it is a measure of the correct versus the incorrect transcriptions. By plugging Eq. (7.14) into Eq. (7.12) or (7.13) to compute the loss functions in Eq. (7.9) and Eq. (7.4), SME with frame selection is implemented.

7.2.3 Implementation with EBW

The EBW formulation with the proposed separation measure in Eqs. (7.8) and (7.14) is implemented as follows. First, an MLE model and a bigram language model (LM) were used to decode all training utterances and generate corresponding word lattices. Then a unigram was used to rescore the decoded lattices. In all the DT methods experimented in this study, a factor of 1/15 was used to scale down the acoustical model likelihood as used in the other DT studies [101], [81],[99]. As noted in Eqs. (7.8) and (7.14), the probabilities of the correct transcriptions are subtracted in the denominators as in [81]. Updating statistics were obtained from the lattices with a forward backward algorithm. Then, EBW was used to update the HMM parameters as in [99]. Because SME directly works on generalization, no I-smoothing was used. SME and MCE share these steps, and only differ in the definition of objective functions.

7.2.4 Initial Experiments of SME with EBW

The MCE model was trained with the implementation in Section 7.2.3. The smoothing constant in the sigmoid function was set to 0.04 as in [81]. EBW was used for HMM parameters update. The WER of the MCE model was 4.60%, getting 9% relative WER reduction over the MLE baseline. This improvement percentage is similar to that reported in [81].

For the purpose of a fair comparison, all the proposed SME methods were modified on the basis of MCE implementation. This means that the implementations are similar, only the individual algorithm parts are different. SME_u and SME_uc indicate the SME models with utterance selections of Eqs. (7.6) and (7.7). The margin and cutting threshold for utterance-based SME are -2 and -60. SME_f and SME_fc are the SME models with the frame selections of Eqs. (7.12) and (7.13). The margin and cutting threshold for frame-based SME are 0.9 and 0.1. All SME models are initiated from MLE model.

Table 7.4 compares the resulting WERs and relative WER reductions of MCE and all the SME methods from MLE. All the proposed SME methods worked better than MCE, achieving about 12%-17% relative WER reduction from MLE baseline. The best was obtained by SME_fc, with the frame selection of Eq. (7.13).

Figures 7.2 and 7.3 compare the histograms of separation measure, $d(.)$, in Eq. (7.8) for MLE and SME_u models. No significant difference for the d values greater than 10 was observed. However, SME_u model moved the samples with d values less than -10 significantly to the right (resulting in bigger d values, which correspond to a better model separation), with 13% relative WER reduction. This demonstrates the optimization strategy of SME which focuses on confusion patterns. SME_uc achieved nearly the same result as SME_u. This shows that outlier is not a critical issue for SME with utterance selection because the utterance level information is stable. All the SME methods with utterance selection work better than MCE, showing that the utterance selection strategy in Eqs. (7.6) and (7.7) is more effective than the sigmoid function in this task.

A closer look at Table 7.4 shows that SME_f, the SME model with hinge loss as the frame selection function, works slightly worse than SME_u. The power of using more confusion patterns did not come out clearly in this case. The reason can be well illustrated in Figure 7.4 by observing the histogram of the frame posterior probabilities of the MLE

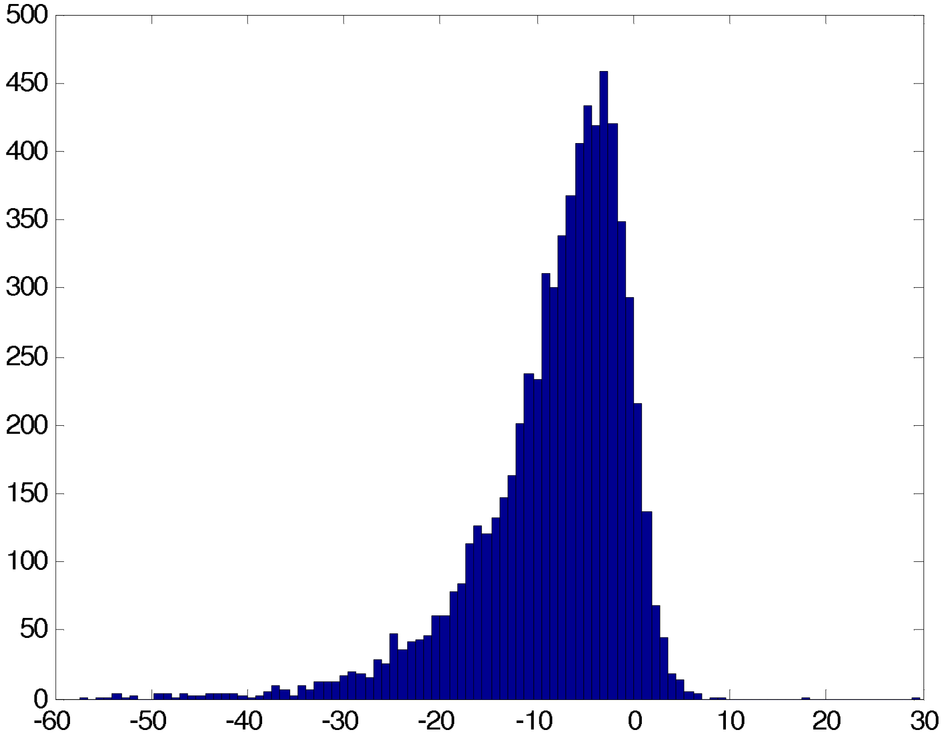

Figure 7.2. The histogram of the separation measure *d* in Eq. (7.8) of MLE model on training set.

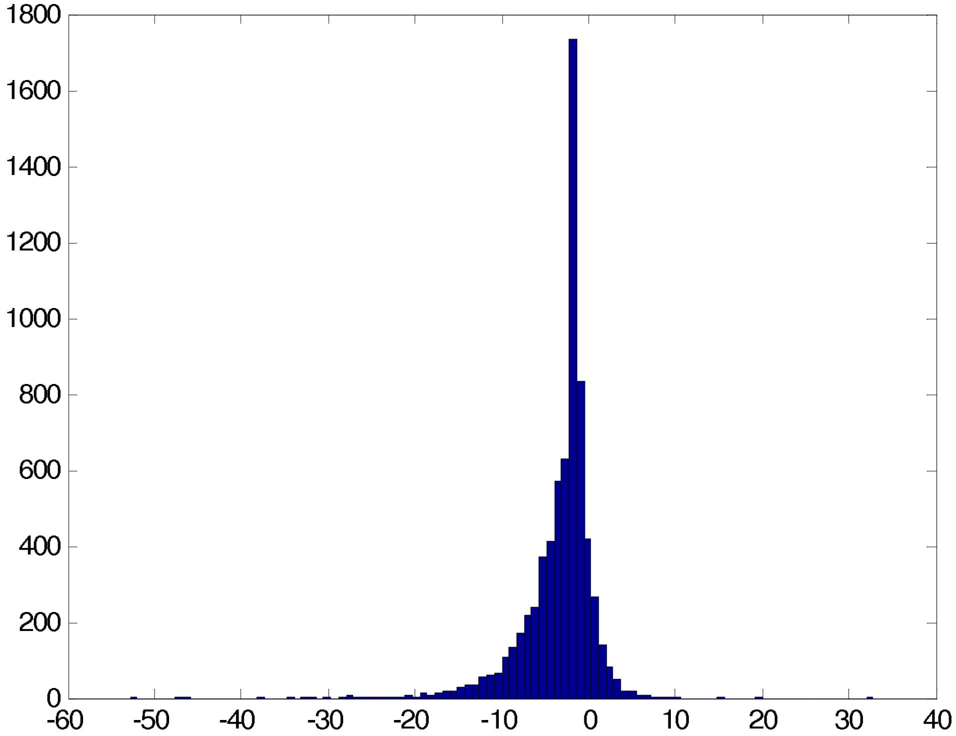

Figure 7.3. The histogram of the separation measure *d* in Eq. (7.8) of SME_u model on training set.

Table 7.4. Performance comparison for discriminative training methods with EBW on the 5k-WSJ0 task.

	WER	Relative Improvement
MLE	5.06%	-
MCE	4.60%	9%
SME_u	4.41%	13%
SME_uc	4.39%	13%
SME_f	4.46%	12%
SME_fc	4.22%	17%

model . Different from the utterance separation measures in Figure 7.2, the distribution of posterior probabilities has two strong modes: one is around 1, and the other is around 0. The reason that too many frames have zero posterior probability indicates that the time alignment of transcription is not precise. Therefore, given some misalignment information, the posterior probabilities are 0. These noisy frames degrade the power of using more confusion frame patterns.

As a solution, SME_fc removed the noisy frames that have too small posterior probabilities by using the loss function defined in Eq. (7.13). The margin and threshold of that loss function are set to be 0.9 and 0.1, respectively. Only a small amount of frames is in this range to be selected by SME_fc for parameter update. The effect is obvious: SME_fc achieved the best result, with a relative 17% WER reduction from the MLE baseline. It should be noted that the loss function in Eq. (7.13) is used to deal with the noisy frames for SME with frame selection because the statistics for frames is not stable. For SME with utterance selection (SME_uc), it makes little difference. As shown in Figure 7.5, the SME_fc histogram of frame posterior probabilities has very few samples lying between the range of [0.1, 0.9], which is the range defined for frame selection. Most the previous samples in that range of the initial MLE model were moved to the region, [0.9, 1]. This means that SME_fc increases the model separation distances of all these confusion patterns. For the frame samples that have the posterior probability less than 0.1, SME_fc did not work on them, which is consistent with what's defined in Eq. (7.13).

It is interesting to how many utterances and frames are selected to optimize HMM

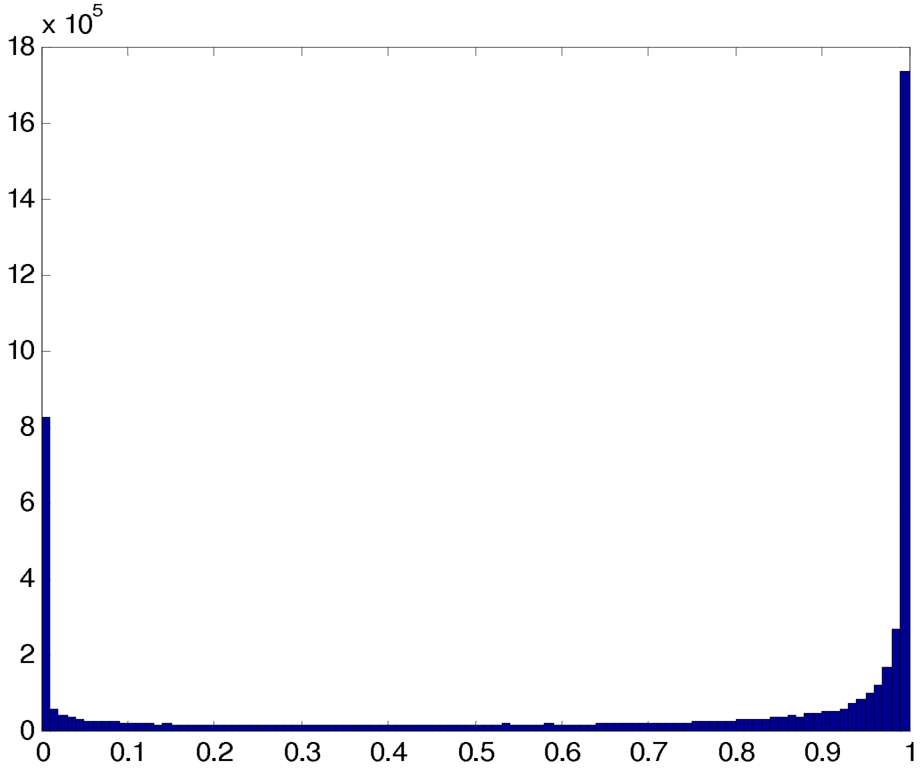

Figure 7.4. The histogram of the frame posterior probabilities of MLE model on training set.

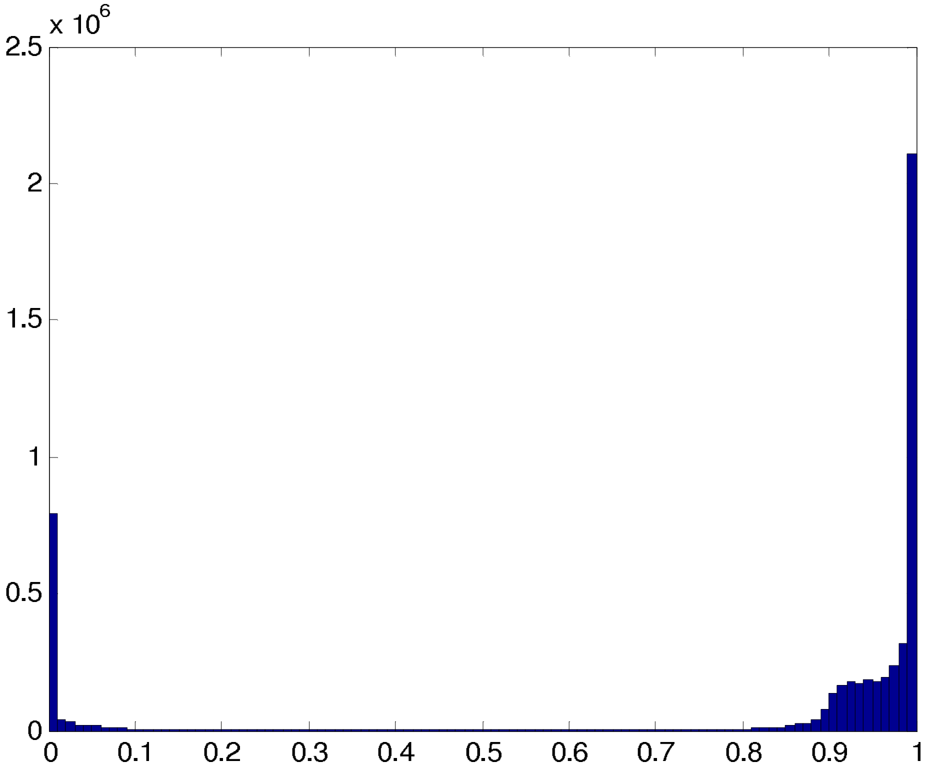

Figure 7.5. The histogram of the frame posterior probabilities of SME_fc model on training set.

parameters in the utterance-based SME and frame-based SME. For SME with utterance selection, about 86% utterances have a separation measure d less than -2 and are used for SME training. After SME training, around 52% utterances have utterance-level d less than -2. In contrast, SME with frame selection works in an aggressive way, with 27% frames falling in the effective posterior probability range [0.1 0.9]. After SME training, only 7% frames are in that range. This means that during the frame-based SME training, only 7%-27% frames are used.

7.2.5 SME with Various Separation Levels

Although the above study got satisfactory results, two potential issues need to be addressed. The first is that the criterion to judge correct and competing candidates was on the string level, although frame-based selection was performed. If word- or phone-level criterion can be used, performance may be boosted according to the success of MWE/MPE. The second is that SME only showed advantage over MCE and didn't compare with MMIE and MPE on the same task. The newest version of HMM toolkit (HTK) [125] provides an accurate tool to train MMIE and MWE/MPE models. We should test the performance of MMIE and MWE/MPE models trained by HTK.

This section addresses the two above-mentioned issues. For the unit separation, string, word, and phone levels are all considered. Different separation measures are defined based on the candidate definition. For comparison, MCE will be compared fairly with SME by sharing most of the implementation details. In addition, MMIE, MWE, and MPE trained by HTK will also be compared.

In the above sections, we have elaborated to design the loss function $l(O_i, \Lambda)$. Different loss functions for SME with frame or utterance selection have been discussed. Frame-based SME with a modified hinge loss function was shown to outperform other options. In this study, we will directly use that loss function. For SME, our target is to separate the correct candidates from the competing candidates with a distance greater than the value of the margin. How to define the candidates is an issue. In the above sections, the candidates are defined in the string level. However, this is different from the target of ASR, which wants to reduce the word error rate instead of the string error rate. In this paper, we further works on the word- and phone-level separation for SME. This turns out to be very effective to reduce the word error rate on LVCSR tasks.

For the string-level unit separation, we directly use the formulations in Section 7.2.2. In the following, that method is denoted as SME_String. It is the same as the method SME_fc reported in the above section.

For the word-level unit separation, we want to separate the correct words from the incorrect words in a decoded lattice. For the correct transcription S_i that consists of N_i correct words, its correct words set is

$$W_i = \left\{ w_i^1, w_i^2, ...w_i^{N_i} \right\} \tag{7.15}$$

Every word in set W_i has a unique label, a starting and an ending time. For any word w in set W_i with a time interval $[t_{ws}, t_{we}]$, we look at all the strings in the decoded lattice. If a segment has the same label with w and also spans $[t_{ws}, t_{we}]$, we consider the strings passing that segment as positive candidates and separate these positive strings from other strings in the duration $[t_{ws}, t_{we}]$. In this way, we can directly work on the local word level of the decoded lattice, and maximize the word accuracy.

To formulate the counter parts of string-level separation with word-level unit, we need to define two string sets for each word w_i^n in set W_i. One set (Φ_i^n) contains all the strings passing w_i^n in the decoded lattice, G_i:

$$\forall R \in \Phi_i^n, \exists w \in R \wedge w = w_i^n \tag{7.16}$$

The other set ($\hat{\Phi}_i^n$) contains all the strings that do not pass w_i^n:

$$\forall \hat{S}_i \in \hat{\Phi}_i^n, \exists \hat{w} \in \hat{S}_i \wedge \hat{w} \neq w_i^n \tag{7.17}$$

The following example is used to differentiate the word-level separation from string-level separation. The arcs in Figure 7.6 can be words or phones. In the lattice in Figure

Table 7.5. Correct and competing paths for word-level separation in Figure 7.6.

time interval	correct paths	competing paths
t_1	“abc”, “abf”, “ae”, “agd’	“hbc”, “hbf”, “he”, “hgd”
t_2	“abc”, “hbc”, “abf”, “hbf”	“ae”, “agd”, “he”, “hgd”
t_3	“abc”, “hbc”	“abf”, “ae”, “agd”, “hbf”, “he”, “hgd”

7.6, the nominator lattice gives the correct paths as “abc” and “ad”. In the denominator lattice, the possible paths are “abc”, “abf”, “ae”, “agd”, “hbc”, “hbf”, “he”, and “hgd”. Among them, only “abc” is treated as a correct path for the string-level formulation. The word-level separation handles the paths in a different way. We first get all the arcs from the nominator lattice. They are “a”, “b”, “c”, and “d”. For the interval t_1 that “a” spans in the nominator lattice, we have one “a” in the denominator lattice. Therefore, the paths “abc”, “abf”, “ae”, and “agd” are considered as correct paths and will be separated from paths “hbc”, “hbf”, “he”, and “hgd” during time interval t_1. For the interval t_2 that “b” spans in the nominator lattice, there are two “b”s in the denominator lattice. Hence, paths “abc”, “hbc”, “abf”, “hbf” need to be separated from the remaining paths during time interval t_2. Similarly, for time interval t_3, paths “abc” and “hbc” will be separated from paths “abf”, “ae”, “agd”, “hbf”, “he”, and “hgd”. We summarize the above content in Table 7.5.

In contrast to the string-level operation, a word w in the decoded lattice is considered to be a correct word when it is in set Φ_i^n (i.e., it has the same word label, starting and ending time as w_i^n). The posterior probability of word w is now defined by summing the probabilities of all the paths in set Φ_i^n that cross word w:

$$p(w|t_{ws}, t_{we}, O_i) = \sum_{R \in \Phi_i^n \bigwedge (w|t_{ws}, t_{we}) \in R} \frac{P_\Lambda(O_i|R)P(R)}{\sum_{\hat{S}_i \in G_i} P_\Lambda(O_i|\hat{S}_i)P(\hat{S}_i)} \tag{7.18}$$

The frame posterior probability is again computed by summing the posterior probabilities of all the correct words that pass time j:

$$q(O_{ij}) = \sum_{w|t_{ws} \leq j \leq t_{we}} p(w|t_{ws}, t_{we}, O_i) \tag{7.19}$$

The frame level separation measure is defined to separate the strings cross time j in set

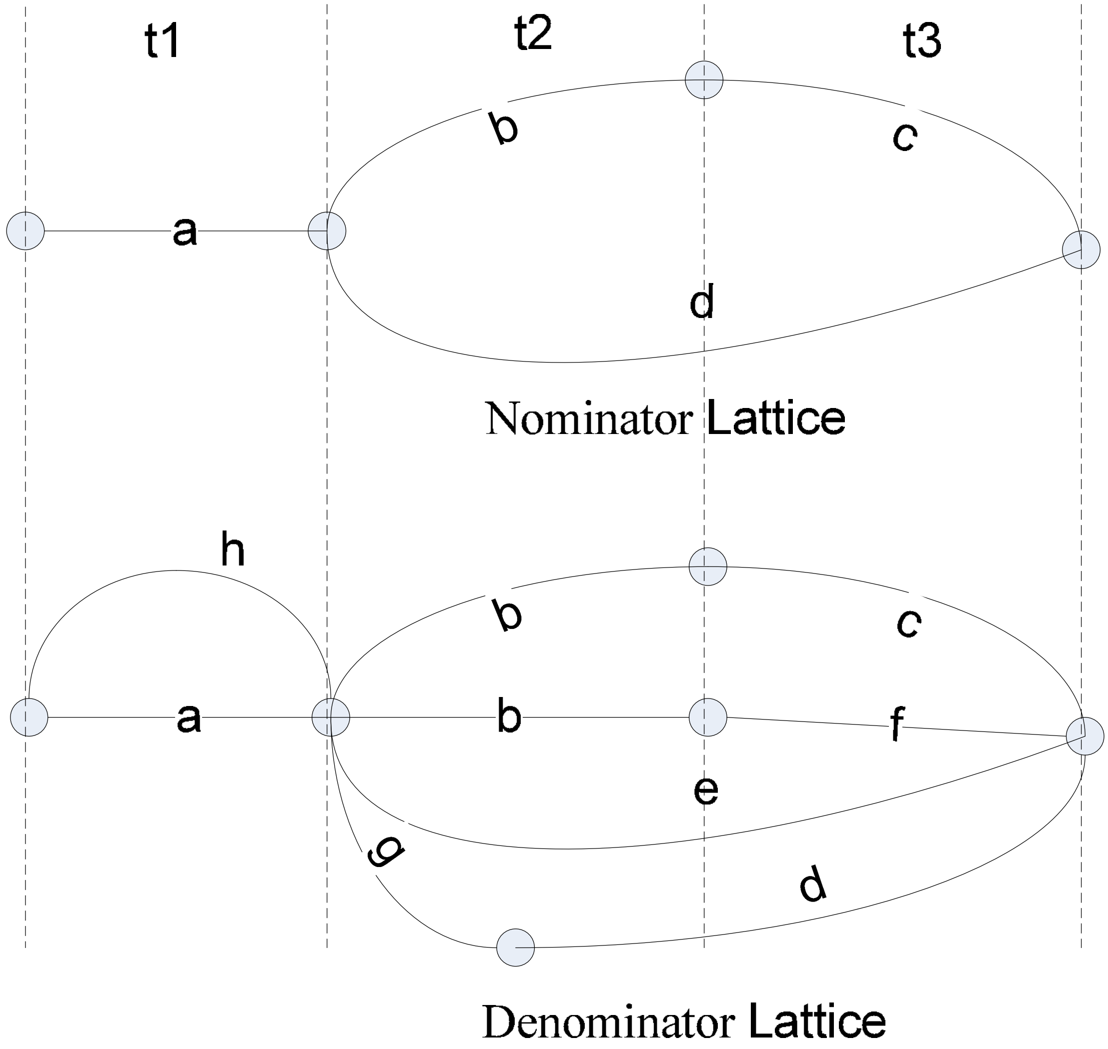

Figure 7.6. A lattice example to distinguish the string-level separation with the word-level separation

Φ_i^n from the strings in set $\hat{\Phi}_i^n$:

$$d(O_{ij}, \Lambda) = log \sum_{w|t_{ws} \leq j \leq t_{we}} \sum_{R \in \Phi_i^n \wedge (w|t_{ws}, t_{we}) \in R} \frac{P_\Lambda(O_i|R)P(R)}{\sum_{\hat{S}_i \in \hat{\Phi}_i^n} P_\Lambda(O_i|\hat{S}_i)P(\hat{S}_i)} \tag{7.20}$$

By plugging Eqs. (7.19) and (7.20) into Eq. (7.13) to compute the loss functions in Eq. (7.4), SME with word-level separation is realized.

An extension from word- to phone-level unit is straightforward. We replace all the word denotations from Eq. (7.15) to Eq. (7.20) with phone denotations.

7.2.6 Practical Implementation Issues

The word or phone posterior probability computation is a key to implement the EBW algorithm. For the denominator lattice in EBW, we need to remove all the correct candidates and rescale the remaining word or phone arcs to get the posterior probabilities. For SME with string-level candidate unit, any word in a correct string may be shared with an incorrect string. Therefore, we need to consider all the word arcs and follow the method proposed in [81] to compute the word posterior probability:

$$\gamma(w|t_{ws}, t_{we}, O_i) = \frac{\sum_{R \in G_i \wedge (w|t_{ws}, t_{we}) \in R} P_\Lambda(O_i|R)P(R) - \sum_{R \in G_i \wedge R = S_i \wedge (w|t_{ws}, t_{we}) \in R} P_\Lambda(O_i|R)P(R)}{\sum_{\hat{S}_i \in G_i} P_\Lambda(O_i|\hat{S}_i)P(\hat{S}_i) - \sum_{\hat{S}_i \in G_i \wedge \hat{S}_i = S_i} P_\Lambda(O_i|\hat{S}_i)P(\hat{S}_i)} \tag{7.21}$$

It is noted that the posterior probability in Eq. (7.21) is for the decoded lattice after the correct strings removed. This is different from the posterior probability in Eq. (7.10), which is for the original decoded lattice.

For SME with word- and phone-level units, it is much simpler. We just consider all the strings passing the incorrect words in a time interval $[t_{ws}, t_{we}]$ and compute the posterior probability as:

$$\gamma(w|t_{ws}, t_{we}, O_i) = \sum_{\hat{S}_i \in \hat{\Phi}_i^n \wedge (w|t_{ws}, t_{we}) \in \hat{S}_i} \frac{P_\Lambda(O_i|\hat{S}_i)P(\hat{S}_i)}{\sum_{\hat{S}_i \in \hat{\Phi}_i^n} P_\Lambda(O_i|\hat{S}_i)P(\hat{S}_i)} \tag{7.22}$$

However, the rescaling of Eq. (7.22) is aggressive since it will magnify the contribution from segments that have large correct word posterior probabilities. For example, consider a segment has two wrong words. Each word has an original posterior probability of 0.05. All other words in that segment are correct words. After rescaling, those two words will each have posterior probability of 0.5, 10 times as their original probabilities. This magnification will hurt the optimization. Therefore, we set a threshold β and only rescale the segment that has a correct word posterior probability less than β.

7.2.7 Experiments of SME with Various Separation Levels

The MMIE, MWE, and MPE models were trained with HTK. The i-smoothing factor was set 100 for MMIE, 25 for MWE, and 50 for MPE, as suggested in [99]. The word error rates (WER) of MMIE, MWE, and MPE on the Nov92 evaluation set are 4.60%, 4.37%, and 3.92%, which correspond to 9%, 14%, and 22% relative WER reductions, respectively.

The MCE model was also trained. The smoothing constant in the sigmoid function was set to 0.04 as in [81]. EBW was used for HMM parameters update. The WER of the MCE model was 4.60%, getting 9% relative WER reduction over the MLE baseline.

For the purpose of a fair comparison, all the proposed SME methods were modified on the basis of the MCE implementation. This means that the implementations are similar, only the individual algorithm parts are different. Three SME models were trained with the string-, word-, and phone-level units separation. They are denoted as SME_String, SME_Word, and SME_Phone, separately. All the SME models were initiated from MLE model. For all the SME models training, the margin ρ and the threshold τ were set as 0.9 and 0.1, individually. For SME_Word and SME_Phone, the threshold β was set as 0.4 for rescaling the posterior probabilities of incorrect arcs in the denominator lattice.

Table 7.6 compares the resulting WERs and relative WER reductions of the conventional DT methods (MCE, MMIE, MWE, and MPE) and all the SME methods (SME_String, SME_Word, and SME_Phone) from the MLE baseline. The conventional DT methods got 9%-22% relative WER reductions. Among them, MPE performed the best, obtaining 22% relative WER reductions. All the proposed SME methods worked better than most of the conventional DT methods, achieving about 17%-25% relative WER reduction from MLE

Table 7.6. Performance comparison for most discriminative training methods with EBW on the 5k-WSJ0 task.

	WER	Relative Improvement
MLE	5.06%	-
MCE	4.60%	9%
MMIE	4.60%	9%
MWE	4.37%	14%
MPE	3.92%	22%
SME_String	4.22%	17%
SME_Word	4.11%	19%
SME_Phone	3.81%	25%

baseline. SME_Phone is the best among all the evaluated methods, reaching 25% relative WER reductions. This shows the objective of phone-level separation is very effective on the 5k-WSJ0 task. Working on the string-level discrimination, SME_String is better than MCE and MMIE. Focusing on the word level, SME_Word also outperforms MWE. The relation is also true for SME_Phone and MPE.

The evolutions of WERs of the best two DT methods, MPE and SME_Phone, are plotted in Figure 7.7. The minimum WERs of MPE and SME_Phone were reached at iteration 12. It is almost in every iteration that the WER of SME_Phone was less than that of MPE.

Figure 7.7 also gives a WER evolution of a method named Phone_Sep. It is a middle step implementation of SME_Phone. Phone_Sep shares most implementation with SME_Phone and targets to separate correct phones from incorrect phones in the decoded lattice, without using margin-based loss function. It also directly rescales the posterior probabilities of the incorrect phone arcs in the denominator lattice, without using a threshold as SME_Phone. In the first iteration, its WER drops to 4.17%. This demonstrates phone-level optimization is very powerful. However, its WER increases quickly in the following iterations. Only three iterations are given in Figure 7.7. As discussed in the above Section, a threshold will make the whole process conservative without dramatically magnifying the contribution of some segments with very large posterior probability for correct arcs. SME formulation finally gives this phone level method great performance.

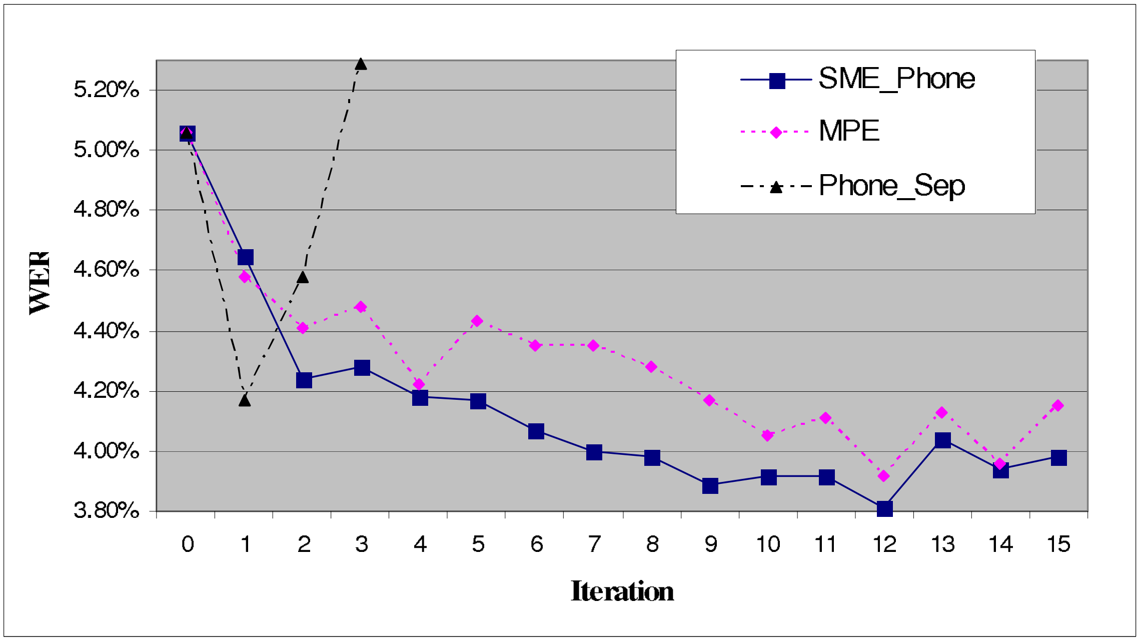

Figure 7.7. Evolutions of testing WER for MPE, SME_Phone, and Phone_Sep models on the 5k-WSJ0 task.

For completeness, the WER evolutions of word-level and string-level DT methods are plotted in Figure 7.8 and Figure 7.9. SME methods are better than their counter part DT methods almost in every iteration.

As Section 7.2.4, we can also plot the frame posterior probabilities for the SME methods with word-level and phone-level separation. Figures 7.10 and 7.11 compare the histograms of frame posterior probabilities, in Eq. (7.19) (computed with word units) for MLE and SME_Word models. Figures 7.10 and 7.11 plot the histogram of the frame posterior probabilities (Eq. (7.19) computed with phone units) of MLE and SME_Phone models on training set with phone-level separation.

It is interesting to look at the number of frames with posterior probabilities as 0. The number of 0-posterior frames in Figure 7.10 (word-level separation) is less than that in Figure 7.4 (string-level separation). And Figure 7.12 (phone-level separation) has the least number of 0-posterior frames. This is because for the string-level processing, a word is considered as a correct word only if it is in a correct path, while for word-level processing, a word is considered as a correct word as long as it is in the correct word set W_i (Eq. (7.15)).

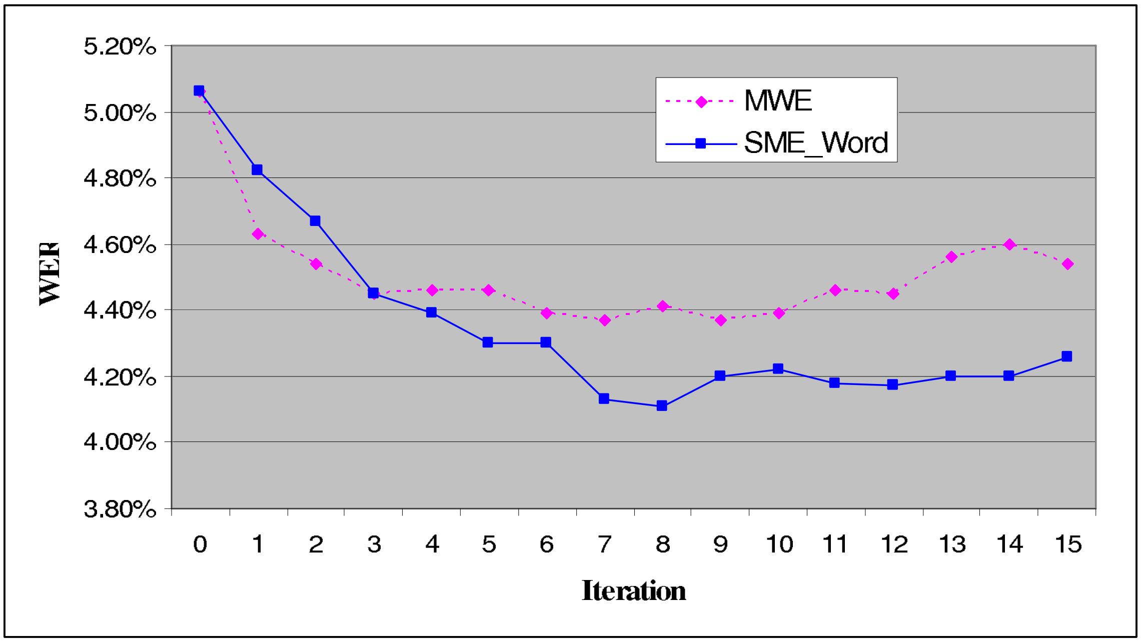

Figure 7.8. Evolutions of testing WER for MWE and SME_Word models on the 5k-WSJ0 task.

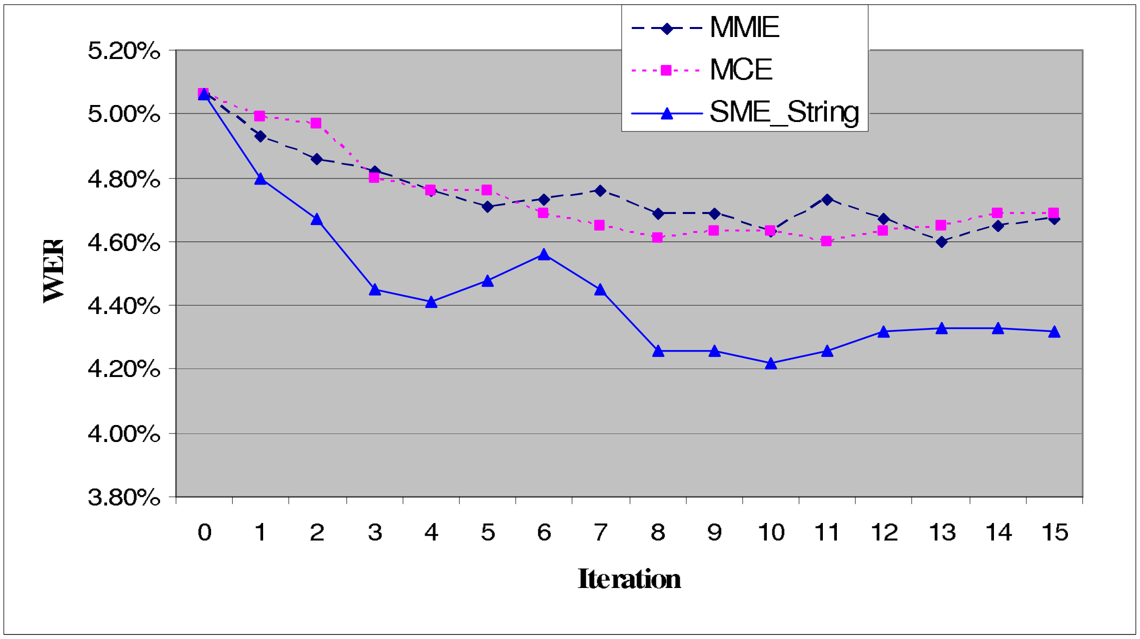

Figure 7.9. Evolutions of testing WER for MMIE, MCE, and SME_String models on the 5k-WSJ0 task.

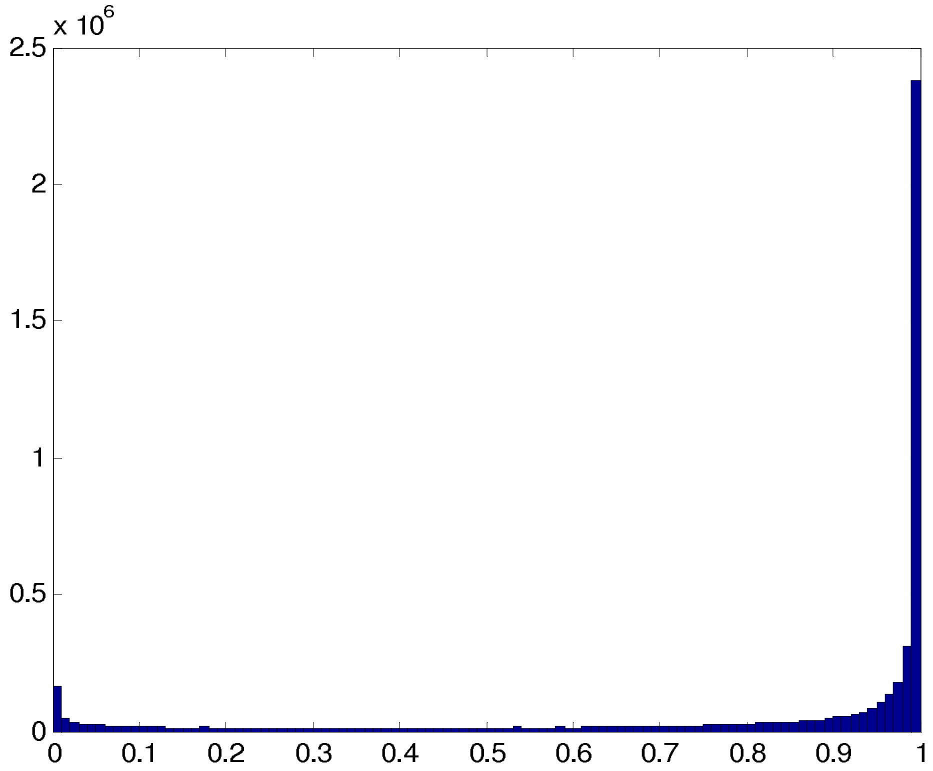

Figure 7.10. The histogram of the frame posterior probabilities of MLE model on training set with word-level separation.

Hence, for every frame, the posterior probability computed with word-level separation must be greater than or equal to that computed with string-level separation. Since a phone can be considered as a correct phone even if it lies in a wrong word, the posterior probability computed with phone-level separation must be greater than or equal to that computed with word-level separation. The above reasons result in different histograms of frame posterior probabilities in Figures 7.4, 7.10, and 7.12, although these three figures use the same MLE model for computation.

In Figure 7.10, about 26% frames have posterior probabilities in the range [0.1, 0.9]. After SME_Word training, about 12% frames are in that range, as shown in Figure 7.11. In Figure 7.12, only 13% frames have posterior probabilities in the range [0.1, 0.9]. After SME_Phone training, about 3% frames are in that range, as shown in Figure 7.13. This is a surprising observation and means only 3%-13% frames are effective in SME_Phone training. And this small amount of frames contribute the best performance, with 25%

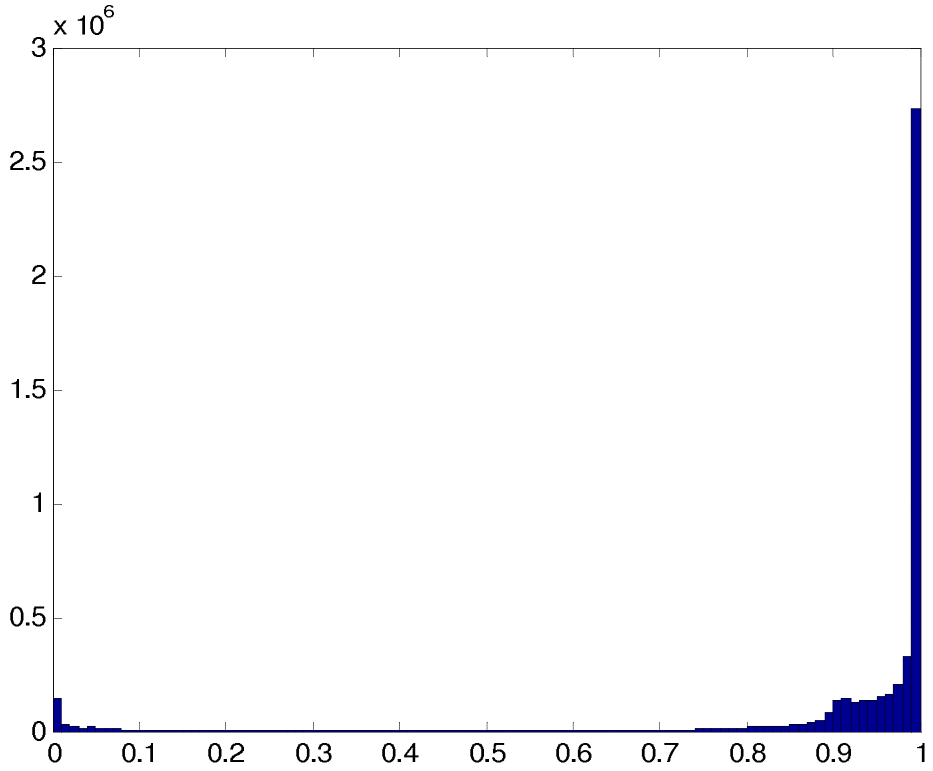

Figure 7.11. The histogram of the frame posterior probabilities of SME_Word model on training set with word-level separation.

relative WER reduction on the 5k-WSJ0 task.

7.3 Conclusion

In this chapter, we first investigate SME with the GPD optimization. The choice of various loss functions is illuminated and different kinds of separation measures are defined under a unified SME framework. Then, SME is applied to LVCSR by defining separation measures at word levels and optimized with the GPD algorithm. Tested on the 5k-WSJ0 task, this SME implementation achieves about 12% relative WER reduction over our best MLE baselines.

The focus of this chapter is to optimize different SME methods with the EBW algorithm. We first demonstrate SME with frame selection works better than SME with utterance selection. Then, SME methods for string-level (SME_String), word-level (SME_Word), and phone-level (SME_Phone) separation are proposed. All these methods focus on how

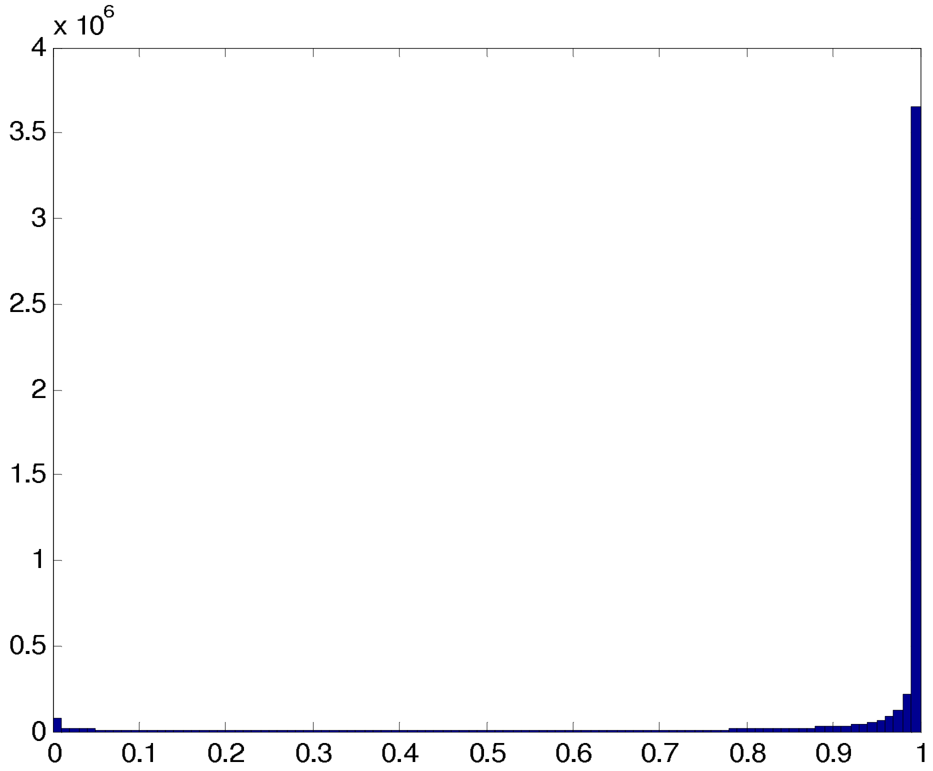

Figure 7.12. The histogram of the frame posterior probabilities of MLE model on training set with phone-level separation.

to make the correct units separate from the competing units with a distance greater than the value of the margin. The most popular DT methods (MMIE, MCE, MWE, and MPE) are used for comparison on the 5k-WSJ0 task. In each separation level, SME outperforms its counter part of conventional DT methods. In string level, SME_String gets 17% relative WER reductions while MCE and MMIE all get 9% relative WER reductions. In word level, SME_Word obtains 19% relative WER reductions while MWE has 14%. In phone level, SME_Phone achieves 25% relative WER reductions, compared to 22% for MPE. We can see that phone-level separation optimization is the most effective way to reduce WER on the 5k-WSJ0 task. SME works the best in this study and demonstrates that the margin-based methods can be a good option for LVCSR tasks.

It is also interesting to observe that SME methods with utterance selection and frame selection only take very small amount of samples to optimize the model parameters. This is especially in the case of SME with frame selection. In some extreme case, only less

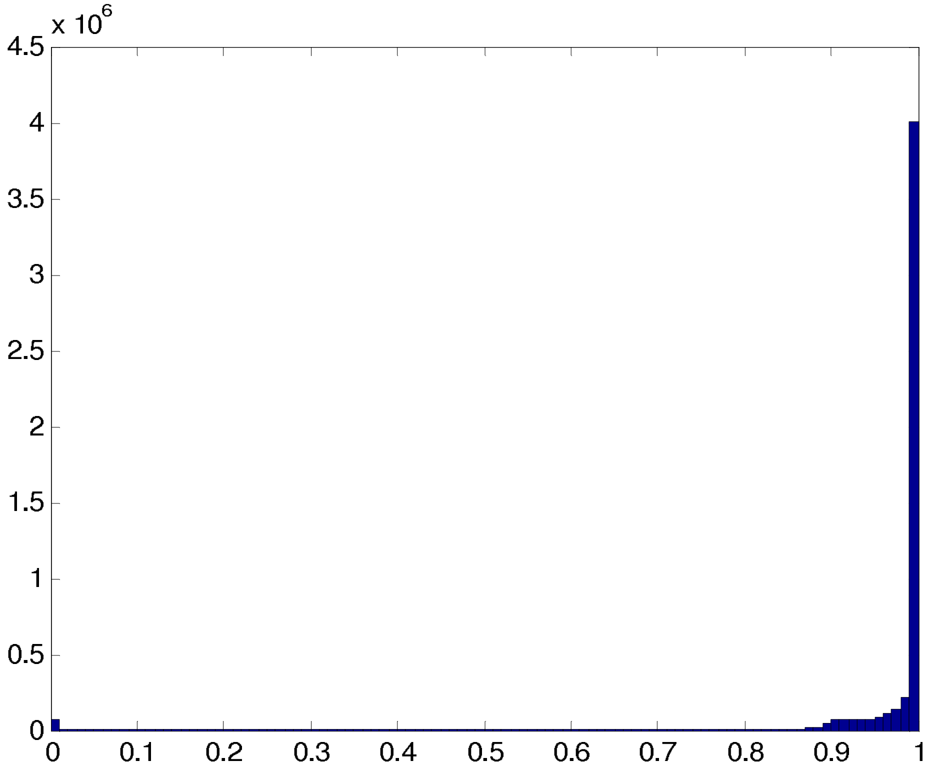

Figure 7.13. The histogram of the frame posterior probabilities of SME_Phone model on training set with phone-level separation.

than 10% frames are used. This phenomenon demonstrates the frames in an utterance have different weights for the contribution of discriminative training.

Two research issues need to be addressed in the future. The first is to extend this study to feature extraction on an LVCSR task. Feature-space minimum phone error (fMPE) [100] has already demonstrated its great power in LVCSR tasks. SME only showed its success in jointly optimization of features and HMM parameters on the TIDIGITS task. We will work SME on the feature extraction part and compare with fMPE on an LVCSR task. The second is to apply SME to even larger LVCSR tasks than the 5k-WSJ0 task.

CHAPTER 8
CONCLUSION

We have proposed a novel discriminative training method, called soft margin estimation (SME), to achieve both high accuracy and good model generalization. This proposed method utilizes the successful ideas of soft margin in support vector machines (SVMs) to improve generalization capability. It directly maximizes the separation of competing models to enhance the testing samples to approach a correct decision if the deviation from training models is within a safe margin. Frame and utterance selections are integrated into a unified framework to select the training utterances and frames critical for discriminating competing models. The choice of various loss functions is illustrated and different kinds of separation measures are defined under a unified SME framework.

Tested on the TIDIGITS database, even 1-mixture SME models can better separate different words and produce better string accuracy (98.76%) than 16-mixture maximum likelihood estimation (MLE) models (98.36%). The performance of SME is consistently better than that of large margin estimation (LME), and significantly better than those of MLE and minimum classification error (MCE). The 16-mixture SME model can achieve 99.30% string accuracy with the generalized probabilistic descent (GPD) optimization. The experiment coincides with the inequality of the test risk bound, showing that even though all the models have nearly the same training errors, the test string accuracies differ because of different margin values associated with various models.

By extending SME, we proposed another discriminative training method, called soft margin feature extraction (SMFE), to achieve even higher accuracy and better model generalization. By jointly optimizing the acoustic feature and HMM parameters under the framework of SME, SMFE performs much better than SME, and significantly better than MLE. Tested on the TIDIGITS database, even 1-mixture model can get string accuracy of 99.13%. And 99.61% string accuracy is got with 16-mixture SMFE model. This is a great improvement comparing to our original SME work that uses MFCCs as acoustic feature. This SMFE work again demonstrates the success of soft margin based method, which directly makes usage of the successful ideas of soft margin in SVMs to improve generalization capability, and of decision feedback learning in minimum classification error training to enhance model separation in classifier design.

It should be noted that there is no exact margin for the inseparable classification task, since different balance coefficients will result in different margin values. Also the test risk bound in statistical learning theory is not tight. The study here is to bridge the research in machine learning and the research in ASR. Although this study has shown the margin-based method performs well in various ASR tasks, more research works need to be done in order to get an explicit theoretic link between the test risk in ASR and the margin in statical learning theory.

To discover whether the margin in SME really has some generalization property, multi-condition testing of MCE and SME with both clean and multi-condition training is investigated on the Aurora2 task. In the clean training case, SME achieves an overall average of 29% relative WER reductions while MCE gets less than 1% relative WER reductions. Although both methods perform similarly when testing with clean utterances, SME outperforms MCE significantly in the testing utterances with SNRs ranging from -5db to 20db. In those mismatched conditions, the margin in SME contributes to classifier generalization and results in great performance improvements for SME. In multi-condition training, SME is slightly better than MCE since in this case the focus of classifier learning is more on minimizing the empirical risk instead of maximizing the margin for generalization. We hope the observations in this study can further deepen the research of generalization property of margin-based classification methods. We also comprehensively worked on the single SNR training case. Only with single SNR training level, some SME options still can get 30% relative WER reductions, with 84% word accuracy, which is close to 87% accuracy in the multi-condition training case. Therefore, single SNR level training may be an option to improve the accuracy in robust ASR applications.

As stated above, there is no explicit theory that explicitly links the test risk in ASR with the margin theory in statistical learning theory. To discover the relationship between the margin and system parameters, we extend the margin from a constant value in our previous work to a function of the system model distance. This distance-based margin is a function of all the model parameters and well characterizes the system generalization. A system divergence is defined as the model distance, and a divergence-based margin is plugged into the objective function of SME. The modified SME achieved similar performance to that obtained with the previously proposed method, and with a possible better theoretic foundation. However, this is only a very preliminary research, and we need to examine whether this is a good attempt in the future.

A focus of SME is to work on large vocabulary continuous speech recognition (LVCSR) tasks. In this study, we first investigate SME with the GPD optimization. SME is applied to LVCSR by defining separation measures at word levels and optimized with the GPD algorithm. Tested on the 5k-WSJ0 task, this SME implementation achieves about 12% relative word error rate (WER) reduction over our MLE baselines. Then, we propose different SME methods optimized with the extended Baum-Welch (EBW) algorithm. We first demonstrate SME with frame selection works better than SME with utterance selection. Then, SME methods for string-level (SME_String), word-level (SME_Word), and phone-level (SME_Phone) separation are proposed. All these methods focus on how to make the correct units separate from the competing units with a distance greater than the value of the margin. The most popular discriminative training methods (maximum mutual information estimation (MMIE), MCE, minimum word error (MWE), and minimum phone error (MPE)) are used for comparison on the 5k-WSJ0 task. In each separation level, SME outperforms its counter part of conventional DT methods. In string level, SME_String gets 17% relative WER reductions while MCE and MMIE all get 9% relative WER reductions. In word level, SME_Word obtains 19% relative WER reductions while MWE has 14%.

In phone level, SME_Phone achieves 25% relative WER reductions, compared to 22% for MPE. We can see that phone-level separation optimization is the most effective way to reduce WER on the 5k-WSJ0 task. SME works the best in this study and demonstrates that the margin-based methods can be a good option for LVCSR tasks. To show the advantage of SME on LVCSR tasks, it is critical to apply SME to even larger LVCSR tasks than the 5k-WSJ0 task.

REFERENCES

[1] A. Acero, L. Deng, T. Kristjansson, and J. Zhang, “HMM adaptation using vector Taylor series for noisy speech recognition,” *Proc. ICSLP,* vol.3, pp. 869-872, 2000.

[2] F. Alleva, X. Huang, and M. Y. Hwang, “Improvements on the pronunciation prefix tree search organization,” *Proc. ICASSP,* pp. 133-136, 1996.

[3] Y. Altun, I. Tsochantaridis, and T. Hofmann, “Hidden Markov support vector machines,” *Proc. ICML,* pp. 3-10, 2003.

[4] T. Anastasakos, J. McDonough, R. Schwartz, and J. Makhou, “A compact model for speaker adaptive training,” *Proc. ICSLP,* pp. 1137-1140, 1996.

[5] B. Atal, “Effectiveness of linear prediction characteristics of the speech wave for automatic speaker identifition and verification,” *J. Acoust. Soc. Amer.,* vol. 55, pp. 1304-1312, 1974.

[6] L. R. Bahl, P. F. Brown, P.V. de Souza, and R. L. Mercer, “Maximum mutual information estimation of hidden Markov model parameters for speech recognition,” *Proc. ICASSP,* vol. 1, pp. 49-52, 1986.

[7] S. J. Benson, Y. Ye and X. Zhang, “Solving large-scale sparse semidefinite programs for combinatorial optimization,” *SIAM Journal on Optimization,* vol. 10, no. 2, pp. 443-461, 2000.

[8] A. Biem, S. Katagiri, and B. -H. Juang, “Pattern recognition using discriminative feature extraction,” *IEEE Trans. Signal Processing,* vol. 45, no. 2, pp. 500-504, 1997.

[9] E. Bocchieri, “Vector quantization for efficient computation of continuous density likelihoods,” *Proc ICASSP,* vol. II, pp692-695, 1993.

[10] S. F. Boll, “Suppression of acoustic noise in speech using spectral subtraction,” *IEEE Trans. Acoust., Speech, Signal Process.,* vol. 27, no. 2, pp. 113-120, 1979.

[11] C. Burges, “A tutorial on support vector machines for pattern recognition,” *Data Mining and Knowledge Discovery,* vol. 2, no. 2, pp. 121-167, 1998.

[12] W. Campbell, “Generalized linear discriminant sequence kernels for speaker recognition,” *Proc. ICASSP,* pp. 161-164, 2002.

[13] W. M. Campbell, E. Singer, P. A. Torres-Carrasquillo, and D. A. Reynolds, “Language recognition with support vector machines,” *Proc. Odyssey: The Speaker and Language Recognition Workshop,* pp. 41-44, 2004.

[14] S. F. Chen and J. Goodman, “An empirical study of smoothing techniques for language modeling,” *Technical Report TR-10-98,* Harvard University, Aug. 1998.

[15] W. Chou, “Discriminant-function-based minimum recognition error rate pattern-recognition approach to speech recognition,” *Proceedings of the IEEE,* vol. 88, no. 8, pp. 1201-1223, 2000.

[16] S. Davis and P. Mermelstein, “Comparison of parametric representations for monosyllable word recognition in continuously spoken sentences,” *IEEE Trans. on Acoustics, Speech and Signal Processing,* vol. 28, no.4, pp. 357-366, 1980.

[17] L. Deng, A. Acero, M. Plumpe, and X. Huang. “Large vocabulary speech recognition under adverse acoustic environments,” *Proc. Interspeech,* vol.3, pp. 806-809, 2000.

[18] L. Deng, J. Wu, J. Droppo, and A. Acero, “Analysis and comparison of two speech feature extraction/compensation algorithms,” vol.12, no.6, pp.477- 480, 2005.

[19] A. P. Dempster, N. M. Laird, D. B. Rubin. “Maximum likelihood from incomplete data via the EM algorithm,” *Journal Roy. Stat. Soc.,* vol. 39, no. 1, pp. 1-38, 1977.

[20] J. Du, P. Liu, F. K. Soong, J. -L. Zhou, and R. -H. Wang, “Minimum divergence based discriminative training,” *Proc. Interspeech,* pp. 2410-2413, 2006.

[21] R. O. Duda, P. E. Hart, and D. G. Stork, *Pattern Classification, 2nd ed.* John Wiley Sons, Inc., New York, 2001.

[22] Q. Fu, A. M. Daniel, B. -H. Juang, J. L. Zhou, and F. K. Soong, “Generalization of the minimum classification error (MCE) training based on maximizing generalized posterior probability (GPP),” *Proc. Interspeech,* pp. 681-684, 2006.

[23] K. Fukunaga, *Introduction to Statistical Pattern Recognition,* 2nd ed. New York: Academic, 1990.

[24] M. J. F. Gales, “Maximum likelihood linear transformations for HMM-based speech recognition,” *Computer Speech and Language,* vol. 12, pp. 75-98, 1998.

[25] M. J. F. Gales and S. Young, “An improved approach to the hidden Markov model decomposition of speech and noise,” *Proc. ICASSP,* vol. I, pp. 233-236, 1992.

[26] A. Ganapathisraju, J. Hamaker, and J. Picone, “Hybrid SVM/HMM architecture for speech recognition,” *Proc. Interspeech,* pp. 504-507, 2000.

[27] S. Gao, W. Wu, C. -H. Lee, and T. -S. Chua, “A maximal figure-of-merit (MFoM)-learning approach to robust classifier design for text categorization,” *ACM Trans. Inf. Syst,* vol. 24, no.2, pp.190-218, 2006.

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

[29] 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. 291-298, 1994.

[30] Y. Gong, “Speech recognition in noisy environments: a survey,” *Speech Communication,* vol. 16, no. 3, pp. 261-291, 1995.

[31] Y. Gong, “A method of joint compensation of additive and convolutive distortions for speaker-independent speech recognition,” *IEEE Trans. Speech and Audio Proc.,* vol. 13, no. 5, pp. 975-983, 2005.

[32] P. S. Gopalakrishnan, D. Kanevsky, A. Nadas, and D. Nahamoo, “An inequality for rational functions with applications to some statistical estimation problems,” *IEEE Trans. on Information Theory,* vol. 37, no.1, pp. 107-113, 1991.

[33] R. Haeb-Umbach, D. Geller, and H. Ney, “Improvement in connected digit recognition using linear discriminant analysis and mixture densities,” *Proc. ICASSP,* pp. 239-242, 1993.

[34] H. Hermansky. “Perceptual linear predictive (PLP) analysis of speech,” *Journal Acoustical Society of America,* vol. 87, no. 4, pp. 1738-1752, 1990.

[35] H. Hermansky and N. Morgan, “RASTA processing of speech,” *IEEE Trans. Speech and Audio Proc.,* vol. 2, no. 4, pp. 578-589, 1994.

[36] J. Hershey and P. Olsen, “Approximating the Kullback Leibler divergence between Gaussian mixture models,” *Proc. ICASSP,* pp. IV 317- IV 320, 2007.

[37] H. G. Hirsch and D. Pearce, “The Aurora experimental framework for the performance evaluation of speech recognition systems under noisy conditions,” *Proc. ISCA ITRW ASR*, 2000.

[38] X. Huang, A. Acero, H. Hon, *Spoken Language Processing: A Guide to Theory, Algorithm and System Development,* Prentice Hall, 2001.

[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] 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 Proc.,* vol. 11, no. 6, pp. 581-589, 2003.

[41] M. J. Hunt and C. Lefebvre, “A comparison of several acoustic representations for speech recognition with degraded and undegraded speech,” *Proc. ICASSP,* pp. 262-265, 1989.

[42] F. Jelinek, *Statistical Methods for Speech Recognition.Cambridge,* MA: MIT Press, 1997.

[43] F. Jelinek and R. L. Mercer, “Interpolated estimation of Markov source param- eters from sparse data,” *Proc. of the Workshop on Pattern Recognition in Practice*, 1980.

[44] H. Jiang and X. Li , “Incorporating Training Errors For Large Margin HMMs Under Semi-definite Programming Framework,”*Proc. ICASSP,* 2007.

[45] H. Jiang, X. Li, and C. Liu, “Large margin hidden Markov models for speech recognition,” *IEEE Trans. On Audio, Speech, and Language Proc.,* vol. 14, no. 5, pp. 1584-1595, 2006.

[46] M. H. Johnson, J. Baker, S. Borys, K. Chen, E. Coogan, S. Greenberg, A. Juneja, K. Kirchhoff, K. Livescu, S. Mohan, J. Muller, K. Sonmez, and T. Wang, “Landmark-based speech recognition: report of the 2004 Johns Hopkins summer workshop,” *Proc. ICASSP,* 2005.

[47] B. H. Juang, “Maximum likelihood estimation for mixture multivariate stochastic observations of Markov chains,” *AT&T Tech. J.,* vol. 64, 1985.

[48] B. -H. Juang, W. Chou, and C. -H. Lee, “Minimum classification error rate methods for speech recognition,” *IEEE Trans. on Speech and Audio Proc.,* vol. 5, no. 3, pp. 257-265, 1997.

[49] B. H. Juang and S. Furui, “Automatic recognition and understanding of spoken language - a first step toward natural human-machine communication,” *Proceedings of the IEEE,* vol. 88, no. 8, pp. 1142-1165, 2000.

[50] S. Katagiri, B. -H. Juang, and C.-H. Lee, “Pattern recognition using a family of design algorithms based upon the generalized probabilistic descent method,” *Proc. IEEE,* vol. 86, no. 11, pp. 2345-2373, 1998.

[51] S. M. Katz, “Estimation of probabilities from sparse data for the language model component of a speech recognizer,” *IEEE Trans. Acoustics, Speech and Signal Processing,* vol.35, no.3, pp. 400-401, 1987.

[52] D. Y. Kim, H. Y. Chan, G. Evermann, M. J. F. Gales, D. Mrva, K. C. Sim, and P. C. Woodland, “Development of the CU-HTK 2004 broadcast news transcription systems,” *Proc. ICASSP,* pp. I861-I864, 2005.

[53] R. Kneser and H. Ney, “Improved backing-off for m-gram language modeling” *Proc. ICASSP,* pp. 181-184, 1995.

[54] N. Kumar and A. G. Andreou, “Heteroscedastic discriminant analysis and reduced rank HMMs for improved speech recognition,” *Speech Communcation,* vol. 26, pp. 283-297, 1998.

[55] H.-K. J. Kuo, E. Fosle-Lussier, H. Jiang, and C. -H. Lee, “ Discriminative training of language models for speech recognition,” *Proc. ICASSP,* vol. I, pp. 325-328, 2002.

[56] H.-K. J. Kuo and C. -H. Lee, “Discriminative training of natural language call routers,” *IEEE Trans. on Speech and Audio Processing,* vol. 11, no. 1, pp.24-35, 2003.

[57] C. -H. Lee, “On stochastic feature and model compensation approaches to robust speech recognition,” *Speech Communication,* vol. 25, pp. 29-47, 1998.

[58] C. H. Lee, F. K. Soong, and K. K. Paliwal, *Automatic Speech and Speaker Recognition: Advanced Topics,* Kluwer Academic, 1996.

[59] C. J. Leggetter and P. C. Woodland, "Maximum likelihood linear regression for speaker adaptation of continuous density HMMs," *Comput., Speech, Lang,* vol. 9, no. 2, pp. 171-185, 1995.

[60] E. L. Lehmann, *Testing Statistical Hypothesis (2nd edition),* Wiley, 1986.

[61] R. G. Leonard, "A database for speaker-independent digit recognition," *Proc. ICASSP,* pp.328-331, 1984.

[62] S. E. Levinson, L. R. Rabiner, and M. M. Sondhi, "An introduction to the application of the theory of probabilistic functions of a Markov process to automatic speech recognition," *Bell System Tech. Journal,* vol. 62, no. 4, pp. 1035-1074, 1983.

[63] Haizhou Li, Bin Ma, and Chin-Hui Lee, "A vector space modeling approach to spoken language identification," *IEEE Transactions on Audio, Speech and Language Processing,* vol. 15, no. 1, pp. 271-284, 2007.

[64] J. Li, L. Deng, D. Yu, Y. Gong, and A. Acero, "HMM adaptation using a phase-sensitive acoustic distortion model for environment-robust speech recognition," *Proc. IEEE ICASSP,* 2008.

[65] J. Li and C. -H. Lee, "Soft margin feature extraction for automatic speech recognition," *Proc. Interspeech,* pp. 30-33, 2007.

[66] J. Li , S. M. Siniscalchi, and C. -H. Lee, "Approximate test risk minimization through soft margin estimation," *Proc. ICASSP,* pp. IV 653-IV 656, 2007.

[67] J. Li, Z. Yan, C. -H. Lee, and R. -H. Wang, "A study on soft margin estimation for LVCSR," *Proc. IEEE ASRU,* 2007.

[68] J. Li, M. Yuan, and C. -H. Lee, “Soft margin estimation of hidden Markov model parameters,” *Proc. Interspeech,* pp. 2422-2425, 2006.

[69] J. Li, M. Yuan, and C. -H. Lee, “Approximate test risk bound minimization through soft margin estimation,” *IEEE Trans. on Audio, Speech, and Language Proc.,* vol. 15, no. 8, pp. 2393-2404, 2007.

[70] J. Li, S. Yaman, C. -H. Lee, B. Ma, R. Tong, D. Zhu, and H. Li, “Language recognition based on score distribution feature vectors and discriminative classifier fusion,” *Proc. IEEE Odyssey,* 2006.

[71] X. Li and H. Jiang. “A constrained joint optimization method for large margin HMM estimation,” *Proc. IEEE ASRU,* pp. 151-156, 2005.

[72] X. Li and H. Jiang. “Solving large-margin hidden Markov model estimation via semidefinite programming,” *IEEE Trans. on Audio, Speech, and Language Proc.,* vol. 15, no. 8, pp. 2383-2392, 2007.

[73] X. Li, H. Jiang, and C. Liu, “Large margin HMMs for speech recognition,” *Proc. ICASSP,* pp. V513-V516, 2005.

[74] X. Li and H. Jiang, “Solving large margin estimation of HMMs via semidefinite programming,” *Proc. Interspeech,* pp. 2414-2417, 2006.

[75] Linguistic Data Consortium (LDC), *The CallFriend Corpora*.

[76] L. A. Liporace, ”Maximum likelihood estimation for multivariate observations of markov sources,” *IEEE Trans. Information Theory,* vol.28, no.5, pp. 729-734, 1982.

[77] C. Liu, H. Jiang, and X. Li, “Discriminative training of CDHMMS for maximum relative separation margin,” *Proc. ICASSP,* pp. I101-I104, 2005.

[78] C. Liu, H. Jiang, and L. Rigazio, “Recent improvement on maximum relative margin estimation of HMMs for speech recognition,” *Proc. ICASSP,* pp. I269-I272, 2006.

[79] J. Louradour, K. Daoudi, and F. Bach, "SVM speaker verification using an incomplete cholesky decomposition sequence kernel," *Proc. IEEE Odyssey,* 2006.

[80] B. Ma, H. Li and C.-H. Lee, "An acoustic segment modeling approach to automatic language identification," *Proc. Interspeech,* Lisbon, Portugal, September 2005.

[81] W. Macherey, L. Haferkamp, R. Schluter, and H. Ney "Investigations on error minimizing training criteria for discriminative training in automatic speech recognition," *Proc. Interspeech,* pp. 2133-2136, 2005.

[82] D. Macho and L. Mauuary, B. Noe, Y. M. Cheng, D. Ealey, D. Jouvet, H. Kelleher, D. Pearce, and F. Saadoun, "Evaluation of a noise-robust DSR front-end on Aurora databases," *Proc. ICSLP,* pp. 17-20, 2002.

[83] E. McDermott, T. J. Hazen, J. L. Roux, A. Nakamura, and S. Katagiri, "Discriminative training for large vocabulary speech," *IEEE Trans. On Audio, Speech, and Language Proc.,* vol. 15, no. 1, pp. 203-223, 2007.

[84] E. McDermott and S. Katagiri, "A new formalization of minimum classification error using a Parzen estimate of classification chance," *Proc. ICASSP,* vol. II, pp. 713-716, 2003.

[85] E. McDermott and S. Katagiri, "A derivation of minimum classification error from the theoretical classification risk using Parzen estimation," *Computer Speech and Language,* vol. 18, pp. 107-122, 2004.

[86] S. Molau, F. Hilger, and H. Ney, "Feature space normalization in adverse acoustic conditions," *Proc. ICASSP,* pp. 656-659, 2003.

[87] H. Ney. "The use of a one-stage dynamic programming algorithm for connected word recognition," *IEEE Trans. on Acoustics, Speech and Signal Processing,* vol. 32, no. 2, pp. 263-271, 1984.

[88] H. Ney, R. Haeb-Umbach, B.-H. Tran, and M. Oerder, “Improvements in beam search for 10000-word continuous speech recognition,” *Proc. ICASSP,* vol. 1, pp. 9-12, 1992.

[89] 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,* pp. 833-836, 1987.

[90] J. Nocedal and S. J. Wright, *Numerical Optimization,* Springer, 2000.

[91] Y. Normandin, *Hidden Markov models, maximum mutual information estimation, and the speech recognition problem.* Ph.D. thesis, McGill University, 1991.

[92] Y. Normandin, “Maximum mutual information estimation of hidden Markov models,” In *Automatic Speech and Speaker Recognition,* C. -H. Lee, F. K. Soong and K. K. Paliwal, Eds. Kluwer Academic Publishers, 1996.

[93] S. Ortmanns, H. Ney, and A. Eiden, “Language-model look-ahead for large vocabulary speech recognition,” *Proc. ICSLP,* pp. 2095-2098, 1996.

[94] M. Padmanabhan and S. Dharanipragada, “Maximum likelihood non-linear transformation for environment adaptation in speech recognition systems,” *Proc. Eurospeech,* pp. 2359– 2362, 2001.

[95] N. Parihar and J. Picone, “Aurora working group: DSR front end LVCSR evaluation AU/384/02,” Institute for Signal and Infomation Processing, Mississippi State Univ, 2002.

[96] D. B. Paul. “Algorithms for an optimal A^* search and linearizing the search in the stack decoder,” *Proc. ICASSP,* vol. 1, pp. 693-696, 1991.

[97] D. B. Paul and J. M. Baker, “The design for the wall street journal-based CSR corpus,” *Proceedings of the workshop on Speech and Natural Language,* 1992.

[98] A. Peinado and J. Segura, *Speech Recognition over Digital Channels— Robustness and Standards.* John Wiley and Sons Ltd, 2006.

[99] D. Povey, *Discriminative Training for Large Vocabulary Speech Recognition,* Ph.D. thesis, Cambridge University Engineering Dept., 2003.

[100] D. Povey, B. Kingsbury, L. Mangu, G. Saon, H. Soltau, and G. Zweig, "FMPE: discriminatively trained features for speech recognition," *Proc. ICASSP,* pp. I961-I964, 2005.

[101] D. Povey and P. Woodland, "Minimum phone error and I-smoothing for improved discriminative training," *Proc. ICASSP,* vol. 1, pp. 105-108, 2002.

[102] L. R. Rabiner, "A tutorial on hidden Markov models and selected applications in speech recognition," *Proceedings of the IEEE,* vol. 77, no. 2, pp. 257-286, 1989.

[103] L. Rabiner and B. -H. Juang. "An introduction to hidden Markov models," *IEEE Transactions on Acoustics, Speech, and Signal Processing,* vol. 3, no. 1, pp. 4-16, 1986.

[104] L. R. Rabiner and B. H. Juang, *Fundamentals of Speech Recognition,* Prentice-Hall, 1993.

[105] M. G. Rahim and C. -H. Lee, "String based minimum verification error (SB-MVE) training for flexible speech recognition," *Comput. Speech Lang.,* 1997.

[106] D. A. Reynolds, T. F. Quatieri, and R. B. Dunn, "Speaker verification using adapted Gaussian mixture models," *Digital Signal Processing,* vol.10, no. 1-3, pp. 19-41, 2000.

[107] R. Schlueter, W. Macherey, B. Muller, and H. Ney, "Comparison of discriminative training criteria and optimization methods for speech recognition," *Speech Communication,* vol. 34, no. 3, pp. 287-310, 2001.

[108] B. Schlkopf and A. J. Smola, *Learning with Kernels.* Cambridge, MA, MIT Press, 2001.

[109] R. Schwartz, L. Nguyen, and J. Makhoul, “Multiple-pass search strategies,” in *Automatic Speech and Speaker Recognition,* C. H. Lee, F. K. Soong, and K. K. Paliwal eds., pp. 57-81, Klewer Academic Publishers, 1996.

[110] F. Sha, *Large margin training of acoustic models for speech recognition,* Ph.D disseration, University of Pennsylvania, 2007.

[111] F. Sha and L. K. Saul, “Large margin Gaussian mixture modeling for phonetic classification and recognition,” *Proc. ICASSP,* pp. I265-I268, 2006.

[112] F. Sha and L. K. Saul, “Large margin hidden Markov models for automatic speech recognition,” in *Advances in Neural Information Processing Systems 19,* B. Scholkopf, J.C. Platt, and T. Hofmann, Eds., MIT Press, 2007.

[113] K. Shinoda and C. -H. Lee, “Structural MAP speaker adaptation using hierarchical priors,” *Proc. IEEE Workshop Automatic Speech Recognition and Understanding,* pp. 381388, 1997.

[114] E. Singer, P. A. Torres-Carrasquillo, T. P. Gleason, W. M. Campbell, and D. A. Reynolds, “Acoustic, phonetic, and discriminative approaches to automatic language recognition,” *Proc. Interspeech,* pp. 1345-1348, 2003.

[115] O. Siohan, C. Chesta, and C. -H. Lee, “Joint maximum a posteriori estimation of transformation and hidden Markov model parameters,” *Proc. ICASSP,* pp. 965-968, 2000.

[116] H. Soltau, B. Kingsbury, L. Mangu, D. Povey, G. Saon, and G. Zweig, “The IBM 2004 conversational telephony system for rich transcription,” *Proc. ICASSP,* pp. I205-I208, 2005.

[117] J. Stadermann and G. Rigoll, “A hybrid SVM/HMM acoustic modeling approach to automatic speech recognition,” *Proc. Interspeech,* pp. 661-664, 2004.

[118] V. Valtchev, J. Odell, P. Woodland, and S. Young, “MMIE training of large vocabulary recognition systems,” *Speech Communication,* vol. 22, no. 4, pp. 303-314, 1997.

[119] V. Vapnik, *The Nature of Statistical Learning Theory,* Springer-Verlag, New York, 1995.

[120] I. H. Witten and T. C. Bell, “The zero-frequency problem: estimating the probabilities of novel events in adaptive text compression,” *IEEE Transactions on Information Theory,* vol. 37, no.4, 1085-1094, 1991.

[121] P. Woodland, J. Odell, V. Valtchev, and S. Young, “Large vocabulary continuous speech recognition using HTK,” *Proc. ICASSP,* pp. II125-II128, 1994.

[122] P. C. Woodland and D. Povey, “Large scale discriminative training of hidden Markov models for speech recognition,” *Computer Speech and Language,* vol. 16, no. 1, pp. 25-47, 2002.

[123] J. Wu and Q. Huo, “An environment-compensated minimum classification error training approach based on stochastic vector mapping,” *IEEE Trans. On Audio, Speech, and Language Proc.*, vol. 14, no. 6, pp. 2147 - 2155, 2006.

[124] Y. Yin and H. Jiang, “A fast optimization method for large margin estimation of HMMs based on second order cone programming,” *Proc. Interspeech,* 2007.

[125] S. Young, G. Evermann, M. Gales, T. Hain, D. Kershaw, X. Liu, G. Moore, J. Odell, D. Ollason, D. Povey, V. Valtchev, and P. C. Woodland, *The HTK Book (for HTK Version 3.4),* Cambridge University, 2006.

[126] D. Yu, L. Deng, X. He, and A. Acero, “Use of incrementally regulated discriminative margins in MCE training for speech recognition,” *Proc. Interspeech,* pp. 2418-2421, 2006.

[127] J. Zheng and A. Stolcke, “Improved discriminative training using phone lattices,” *Proc. Interspeech,* 2005.

[128] M. A. Zissman, “Comparison of four approaches to automatic language identification of telephone speech,” *IEEE Trans. on Speech and Audio Processing,* vol. 4, no.1, pp. 31-44, 1996.

VITA

Jinyu Li received the B. Eng and M. Eng degrees in electrical engineering and information system from University of Science and Technology of China (with the highest honor, Guo Moruo Award), in 1997 and 2000, respectively. After working as a researcher in Intel China research center, he started the speech recognition research group in Anhui USTC iFlytek from 2001 to 2003, which is now the most successful speech company in China. Since 2004, he is a Ph.D. student under the supervision of Prof. Chin-Hui Lee at the school of electrical and computer engineering, Georgia Institute of Technology, Atlanta, U.S. He is the winner of Colonel Oscar P. Cleaver Award for the highest score in Ph.D. prelim test, 2004. He is also the winner of the best student paper of Interspeech, 2006.

In addition to the research of soft margin estimation presented in this dissertation, Jinyu Li is also active and has publications on several topics in speech recognition, including discriminative training, noise robustness, attribute detector design and spoken language recognition. The publications during his Ph.D. study in Georgia Institute of Technology are listed in the following.

1. **J. Li** and C. -H. Lee, "On a generalization of margin-based discriminative training to robust speech recognition," *Proc. Interspeech,* 2008.

2. **J. Li**, Z. Yan, C. -H. Lee, and R. -H. Wang, "Soft margin estimation with various separation levels for LVCSR," *Proc. Interspeech,* 2008.

3. **J. Li**, L. Deng, D. Yu, Y. Gong, and A. Acero, "HMM adaptation using a phase-sensitive acoustic distortion model for environment-robust speech recognition," *Proc. IEEE ICASSP,* 2008.

4. **J. Li**, L. Deng, D. Yu, J. Wu, Y. Gong, and A. Acero, "Adaptation of compressed

HMM parameters for resource-constrained speech recognition," *Proc. IEEE ICASSP,* 2008.

5. **J. Li**, M. Yuan, and C. -H. Lee, "Approximate test risk bound minimization through soft margin estimation," *IEEE Trans. on Audio, Speech, and Language Proc.,* vol. 15, no. 8, 2007.

6. **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," *Proc. IEEE ASRU,* 2007.

7. **J. Li**, Z. Yan, C. -H. Lee, and R. -H. Wang, "A study on soft margin estimation for LVCSR," *Proc. IEEE ASRU,* 2007.

8. **J. Li** and C. -H. Lee, "Soft margin feature extraction for automatic speech recognition," *Proc. Interspeech,* 2007.

9. I. Bromberg, Q. Fu, J. Hou, **J. Li**, C. Ma, B. Mattews, 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.

10. **J. Li**, S. M. Siniscalchi, and C. -H. Lee, "Approximate test risk minimization through soft margin estimation," *Proc. IEEE ICASSP,* 2007.

11. **J. Li**, M. Yuan, and C. -H. Lee, "Soft margin estimation of hidden Markov model parameters," *Proc. Interspeech,* 2006. (Best Student Paper)

12. M. Siniscalchi, **J. Li** and C. -H. Lee, "A study on lattice rescoring with knowledge scores for automatic speech recognition," *Proc. Interspeech,* 2006.

13. **J. Li**, S. Yaman, C. -H. Lee, B. Ma, R. Tong, D. Zhu, and H. Li, "Language recognition based on score distribution feature vectors and discriminative classifier fusion," *Proc. IEEE Odyssey,* 2006.

14. **J. Li** and C. -H. Lee, “On designing and evaluating speech event detectors,” *Proc. Interspeech,* 2005.

15. Y. Tsao, **J. Li** and C. -H. Lee, “A study on separation between acoustic models and its applications,” *Proc. Interspeech,* 2005.

16. **J. Li**, Y. Tsao, and C. -H. Lee, “A study on knowledge source integration for rescoring in automatic speech recognition,” *Proc. IEEE ICASSP,* 2005.