UTTERANCE PARTITIONING FOR SUPERVECTOR AND I-VECTOR SPEAKER VERIFICATION

WEI RAO

Ph.D

The Hong Kong Polytechnic University

2015

The Hong Kong Polytechnic University

Department of Electronic and Information Engineering

Utterance Partitioning for Supervector and I-Vector Speaker Verification

Wei RAO

A thesis submitted in partial fulfillment of
the requirements for the degree of

Doctor of Philosophy

November 2014

CERTIFICATE OF ORIGINALITY

I hereby declare that this thesis is my own work and that, to the best of my knowledge and belief, it reproduces no material previously published or written, nor material that has been accepted for the award of any other degree or diploma, except where due acknowledgement has been made in the text.

________________________________(Signed)

_____________**Wei RAO**_____________(Name of student)

To my parents

Abstract

In recent years, GMM–SVM and i-vectors with probabilistic linear discriminant analysis (PLDA) have become prominent approaches to text-independent speaker verification. The idea of GMM–SVM is to derive a GMM-supervector by stacking the mean vectors of a target-speaker dependent, MAP-adapted GMM. The supervector is then presented to a speaker-dependent support vector machine (SVM) for scoring. However, a problematic issue of this approach is the severe imbalance between the numbers of speaker-class and impostor-class utterances available for training the speaker-dependent SVMs.

Different from high dimension GMM-supervectors, the major advantage of i-vectors is that they can represent speaker-dependent information in a low-dimension space, which opens up opportunity for using statistical techniques such as linear discriminant analysis (LDA), within-class covariance normalization (WCCN), and PLDA to suppress the channel- and session-variability. While these techniques have achieved state-of-the-art performance in recent NIST Speaker Recognition Evaluations (SREs), they require multiple training speakers each providing sufficient numbers of sessions to train the transformation matrices or loading matrices. However, collecting such a corpus is expensive and inconvenient. In a typical training dataset, the number of speakers could be fairly large, but the number of speakers who can provide many sessions is quite limited. The lack of multiple sessions per speaker could cause numerical problems in the within speaker scatter matrix, a problematic issue known as the small sample-size problem in the literature.

Although the above-mentioned data imbalance problem and small sample-size problem are caused by different reasons, both of them can be overcome by an utterance partitioning and resampling technique proposed in this thesis. Specifically, the sequence order of acoustic vectors in an enrollment utterance is first randomized; then the randomized sequence is partitioned into a number of segments. Each of these segments is then used to compute a GMM-supervector or an i-vector. A desirable number of supervectors/i-vectors can be produced by repeating this randomization and partitioning process a number of times. This method is referred to as utterance partitioning with acoustic vector resampling (UP–AVR). Experiments on the NIST 2002, 2004 and 2010 SREs show that UP–AVR can help the SVM training algorithm to find better decision boundaries so that SVM scoring outperforms other speaker comparison methods such as cosine distance scoring. Furthermore, results demonstrate that UP–AVR can enhance the capability of LDA and WCCN in suppressing session variability, especially when the number of conversations per training speaker is limited.

This thesis also proposes a new channel compensation method called multi-way LDA that uses not only the speaker labels but also microphone labels in the training i-vectors for estimating the LDA projection matrix. It was found that the method can strengthen the discriminative capability of LDA and overcome the small sample-size problem.

To overcome the implicit use of background information in the conventional PLDA scoring in i-vector speaker verification, this thesis proposes a method called PLDA-SVM scoring that uses empirical kernel maps to create a PLDA score space for each target speaker and train an SVM that operates in the score space to produce verification scores. Given a test i-vector and the identity of the target speaker under test, a score vector is constructed by computing the PLDA scores of the test i-vector with respect to the target-speaker's i-vectors and a set of nontarget-speakers' i-vectors. As a result, the bases of the score space are divided into two parts: one defined by the target-speaker's i-vectors and another defined by the nontarget-speakers' i-vectors. To ensure a proper balance between the two parts, utterance partitioning is applied to create multiple target-speaker's i-vectors from a single or a small number of utterances. With the new protocol brought by NIST SRE, this thesis shows that PLDA-SVM scoring not only performs significantly better than the conventional PLDA scoring and utilizes the multiple enrollment utterances of target speakers effectively, but also opens up opportunity for adopting sparse kernel machines for PLDA-based speaker verification systems. Specifically, this thesis shows that it is possible to take the advantages of the empirical kernel maps by incorporating them into a more advanced kernel machine called relevance vector machine (RVM). Experiments on NIST 2012 SRE suggest that the performance of PLDA-RVM regression is slightly better than that of PLDA-SVM after performing UP-AVR.

AUTHOR'S PUBLICATIONS

Journal Papers

1. M.W. Mak and **W. Rao**, "Utterance Partitioning with Acoustic Vector Resampling for GMM-SVM Speaker Verification", *Speech Communication*, vol. 53, no. 1, Jan. 2011, pp. 119–130.

2. **W. Rao** and M.W. Mak, "Boosting the Performance of I-Vector Based Speaker Verification via Utterance Partitioning", *IEEE Trans. on Audio, Speech and Language Processing*, vol. 21, no. 5, May 2013, pp. 1012–1022.

Conference Papers

1. W. Jiang, M. W. Mak, **W. Rao**, and H. Meng, "The HKCUPU System for the NIST 2010 Speaker Recognition Evaluation", *IEEE Int. Conf. on Acoustic Speech and Signal Processing (ICASSP 2011)*, Prague, Czech Republic, May 2011, pp. 5288–5291.

2. M.W. Mak and **W. Rao**, "Acoustic Vector Resampling for GMMSVM-Based Speaker Verification", *11th Annual Conference of the International Speech Communication Association (Interspeech 2010)*, Makuhari, Japan, Sept. 2010, pp. 1449–1452.

3. M.W. Mak and **W. Rao**, "Likelihood-Ratio Empirical Kernels for I-Vector Based PLDA-SVM Scoring", *IEEE Int. Conf. on Acoustic Speech and Signal Processing (ICASSP 2013)*, Vancouver, Canada, May 2013, pp. 7702–7706.

4. **W. Rao** and M.W. Mak, "Addressing the Data-Imbalance Problem in Kernel-based Speaker Verification via Utterance Partitioning and Speaker Comparison", *12th Annual Conference of the International Speech Communication Association (Interspeech 2011)*, Florence, Italy, Aug. 2011, pp. 2717–2720.

5. **W. Rao** and M.W. Mak, "Utterance Partitioning with Acoustic Vector Resampling for I-Vector Based Speaker Verification", *Odyssey 2012: The Speaker and Language Recognition Workshop*, Singapore, Jun 2012, pp. 165–171.

6. **W. Rao** and M.W. Mak, "Alleviating the Small Sample-Size Problem in I-Vector Based Speaker Verification", *Int. Sym. on Chinese Spoken Language Processing (ISCSLP 2012)*, Hong Kong, Dec. 2012, pp. 335–339.

7. **W. Rao** and M.W. Mak, "Construction of Discriminative Kernels from Known and Unknown Non-targets for PLDA-SVM Scoring", *IEEE Int. Conf. on Acoustic Speech and Signal Processing (ICASSP 2014)*, Florence, Italy, May 2014, pp. 4040–4044.

8. **W. Rao** and M.W. Mak, "Relevance Vector Machines with Empirical Likelihood-Ratio Kernels for PLDA Speaker Verification", *Int. Sym. on Chinese Spoken Language Processing (ISCSLP 2014)*, Singapore, Sep. 2014.

9. **W. Rao**, M. W. Mak, and K. A. Lee, ”Normalization of Total Variability Matrix for I-vector/PLDA speaker verification”, *IEEE Int. Conf. on Acoustic Speech and Signal Processing (ICASSP 2015)*, Brisbane Australia, Apr. 2015. (Accepted)

10. J.H. Zhong, W. Jiang, **W. Rao**, M.W. Mak, and H. Meng, ”PLDA Modeling in the Fishervoice Subspace for Speaker Verification”, *15th Annual Conference of the International Speech Communication Association (Interspeech 2014)*, Singapore, Sep. 2014.

ACKNOWLEDGMENTS

First and foremost, I would like to express my special appreciation and thanks to my PhD supervisor Dr. Man-Wai Mak for his guidance, encouragement, and patience. Throughout my time at The Hong Kong Polytechnic University, his immense and extensive knowledge amazed me and his enthusiasm on scientific research, rigorous scholarship, and genial personality deeply infected and inspired me. I do appreciate that he has offered countless helpful suggestions for my research work and instructed and helped me on writing papers and this thesis. I am extremely fortune and grateful to have Dr. Man-Wai Mak as my supervisor.

Furthermore, my sincere gratitude goes to our university, our department and Research Office for their generous and consistent support during my PhD study. My deep thanks also go to the secretarial support staff including Suki, Cora, Shirley and other members in the General Office of our department, whose help on the administrative field are much appreciated.

I also thanks to Dr. Kong-Aik Lee, Dr. Bin Ma and Prof. Hai-Zhou Li from Institute for Infocomm Research (I2R), for offering me the exchange opportunities in their groups and leading me working on the exciting research topic.

In my time at PolyU, I met a lot of friends and I am grateful to their help and support. Particular thanks to Lin-Lin Cao, Shi-Biao Wan, Li-Li Wang, Xue Dong, Jin Xu, Cheng-Xing Meng, Wen Wang, Hon-Bill Yu, Wei Wang ,Shi-Xiong Zhang and Jia-Jie Fan. My gratitude also goes to my friend Jie Qiu for her encouragement and concern. Besides, I would like to thank Sven, Pablo, Peng Yang, Hai-Bo Liu, Lei Wang, Jeremy, Yvonne, Gary, Poh-Keng Teo and her family. Their help on both my academic research and my life in Singapore has been the most unforgettable and pleasant time of my exchange.

Special thanks to my boyfriend for his love, support, and patience during my PhD study. He is an invaluable source of moral support and joy in my life. Last but not the least, I would like to express my deepest gratitude to my parents Chao-Mei Rao and Cui-E Yang for their unconditional and unselfish love, constant concern, and persistent support. I do appreciate that my parents always offer me helpful advices, positive energy, and courage to overcome obstacles. This thesis is dedicated to my parents.

TABLE OF CONTENTS

Table of Contents i

List of Figures vii

List of Tables xvi

List of Abbreviations xxiii

Chapter 1: Introduction 1

1.1 Speaker Verfication 1

1.2 Motivation of the Thesis 3

1.3 Contribution of the Thesis 5

1.4 Thesis Organization 7

Chapter 2: Speaker Verification 8

2.1 Overview of Speaker Verification Systems 8

2.2 Acoustic Features for Speaker Verification 10

2.3 Speaker Modeling 12

2.3.1 Gaussian Mixture Models 13

2.3.2 GMM–UBM 15

2.4 Score Normalization 17

2.4.1 Zero Normalization (Z-Norm) 17

2.4.2 Test Normalization (T-Norm) 18
2.4.3 ZT-Norm and TZ-Norm 19
2.4.4 Symmetric Normalization (S-Norm) 19
2.5 Performance Evaluation Metrics 20
2.5.1 Equal Error Rate 21
2.5.2 Detection Cost Function 22
2.5.3 Detection Error Tradeoff Curve 24

Chapter 3: GMM-supervector Based Speaker Verification 25

3.1 GMM-Supervectors 25
3.2 GMM–SVM 27
3.3 GMM–UBM Scoring vs. GMM–SVM Scoring 29
3.4 Nuisance Attribute Projection 30
3.5 Data Imbalance Problem in GMM–SVM Systems 31
3.6 Speaker Comparison 33

Chapter 4: I-Vector Based Speaker Verification 35

4.1 I-Vector Extraction 35
4.2 Channel Compensation 38
4.2.1 Linear Discriminant Analysis 39
4.2.2 Within-Class Covariance Normalization 40
4.2.3 Multi-way Linear Discriminant Analysis 40
4.3 Scoring Methods 42
4.3.1 Cosine Distance Scoring 43
4.3.2 SVM Scoring 43

4.3.3 PLDA Scoring 44
4.4 Small Sample-Size Problem in LDA and WCCN 45

Chapter 5: Utterance Partitioning with Acoustic Vector Resampling 46

5.1 Approaches to Alleviate Data Imbalance 47
5.1.1 Data Processing Approaches 47
5.1.2 Algorithmic Approaches 49
5.2 Approaches to Solve Small Sample-Size Problem 50
5.3 UP-AVR for GMM–SVM Speaker Verification 52
5.3.1 Motivation of UP-AVR 52
5.3.2 Procedure of UP-AVR 53
5.3.3 Advantages of UP-AVR 54
5.4 UP-AVR for I-Vector Speaker Verification 57
5.4.1 Effect of Utterance Length on I-Vectors 59
5.4.2 Properties of I-Vectors Produced by UP-AVR 62
5.4.3 Alleviating Small Sample-Size Problem 64
5.4.4 Alleviating Data Imbalance Problem 65

Chapter 6: Gaussian-PLDA with Sparse Kernel Machines 66

6.1 Gaussian–PLDA 68
6.1.1 Effect of Length Normalization on I-vectors 69
6.1.2 Estimation of PLDA Parameters 71
6.1.3 PLDA Scoring 73
6.2 Multi-Utterance Scoring for PLDA Speaker Verification 75

6.3 Empirical Kernel Maps for PLDA-SVM Scoring 78

6.3.1 Empirical Kernel Maps 81

6.3.2 PLDA-SVM 83

6.4 Empirical Kernel Maps for PLDA-RVM Scoring 86

6.4.1 Relevance Vector Machine 86

6.4.2 PLDA-RVM 89

6.5 Known Non-targets for PLDA-SVM/PLDA-RVM Scoring 90

Chapter 7: Experimental Setup 92

7.1 Speech Data 92

7.2 Acoustic Front-End Processing 98

7.3 GMM-supervector Based Speaker Verification 99

7.3.1 Speaker Models 99

7.3.2 Session and Channel Variability Modelling 100

7.3.3 Score Normalization 102

7.4 I-Vector Based Speaker Verification 102

7.4.1 Total Variability Modeling and Channel Compensation 102

7.4.2 Scoring Method and Score Normalization 102

7.4.3 Utterance Length after Utterance partitioning 103

7.5 Gaussian-PLDA with Sparse Kernel Machines 103

7.5.1 Enrollment Utterances and Non-Target Speaker Utterances . 103

7.5.2 Total Variability Modeling and PLDA 105

Chapter 8: Results and Discussion 106

8.1 GMM-supervector Based Speaker Verification 106

8.1.1 Statistical Properties of GMM-Supervectors 106
8.1.2 GMM–UBM vs. GMM–SVM 107
8.1.3 Comparing Kernels in GMM–SVM 107
8.1.4 Comparing Kernels in Speaker Comparison 111
8.1.5 GMM–SVM vs. Speaker Comparison 113
8.1.6 Effectiveness of UP–AVR 113
8.1.7 Comparing with Other Systems 121
8.2 I-Vector Based Speaker Verification 121
8.2.1 Effects of Training-set Size on I-Vectors 121
8.2.2 Small Sample-Size Problem in LDA and WCCN 123
8.2.3 Effectiveness of Multi-way LDA 126
8.2.4 UP-AVR for LDA and WCCN 128
8.2.5 UP-AVR for SVM Scoring 131
8.3 Gaussian PLDA with Sparse Kernel Machines 134
8.3.1 Choice of I-Vectors for Empirical Kernel Maps 134
8.3.2 Choice of Non-Targets for SVM Training 135
8.3.3 Use of Noisy Speech for PLDA Training 137
8.3.4 Comparison of Scoring Methods 141
8.3.5 Importance of UP-AVR for SVM and LR Scoring 143
8.3.6 Property of Empirical Kernel Maps in SVM and RVM 144
8.3.7 Optimization of RBF Parameter Gamma 146
8.3.8 PLDA-SVM vs. PLDA-RVM 148

Chapter 9: Conclusions and Future Work **150**

9.1 Conclusions 150

9.2 Future Work 152

Appendix A: Notation **154**

Bibliography **156**

LIST OF FIGURES

1.1 The task of speaker verification. 2

2.1 The enrollment and verification phases of a speaker verification system. 9

2.2 The relationship between the Mel-frequencies (f^{Mel}) and linear frequencies (f^{Lin}). 11

2.3 The flow chart of extracting MFCCs. 12

2.4 The enrollment and verification phases of GMM–UBM based speaker verification. Refer to Eq. 2.8 for GMM–UBM scoring function. 16

2.5 The procedure of Z-Norm in GMM–UBM scoring. 17

2.6 The procedure of T-Norm in GMM–UBM scoring. 18

2.7 FRR and FAR with respective to the decision threshold. An EER of 2.62% is obtained when the decision threshold is equal to -27.50. . . 21

2.8 An example of DET curves. 24

3.1 The flow chart of GMM-supervector extraction. $\mathbf{O}^{(s)}$ is the sequence of acoustic vectors extracted from utt$^{(s)}$ of speaker s, $\mathbf{\Lambda}^{(s)}$ is the GMM model of speaker s, w_i is the i-th mixture weight of GMM model, and μ_i is the mean of i-th Gaussian component. 26

3.2 The enrollment and verification phases of GMM–SVM based speaker verification. Refer to Eq. 3.1 for the SVM scoring function. 28

3.3 A two-class problem in 2-dimensional space illustrating the imbalance between the number of positive-class samples and negative-class samples. (a) The orientation (slope) of the decision boundary depends largely on the negative-class data. (b) Adding more positive-class data can enhance the influence of the positive-class data on the decision boundary (slope changes from -1.0 to -1.12). There is a region in the feature space where the positive-class's support vector (encircled $\square$) can move around without affecting the orientation of the decision boundary, but a small change in the negative-class' support vectors (encircled $*$) can tilt the decision boundary. 32

3.4 A 3-dimensional two-class problem illustrating the imbalance between the number of minority-class samples (red pentagram) and majority-class samples (pink triangles) in a linear SVM. The decision plane is defined by 3 support vectors (enclosed by black circles). The green region beneath the decision plane represents the region where the minority-class sample can be located without changing the orientation of the decision plane. 33

4.1 The flow chart of i-vector based speaker verification system. $\mathbf{N}_s$ and $\mathbf{N}_c$ are the zero-order Baum-Welch statistics for target speaker s and test c, respectively. $\mathbf{F}_s$ and $\mathbf{F}_c$ are the first-order Baum-Welch statistics for target speaker s and test c, respectively. $\mathbf{x}_s$ and $\mathbf{x}_c$ are i-vectors for target speaker s and test c, respectively. $\mathbf{A}$ is the LDA projection matrix and $\mathbf{B}$ is the WCCN projection matrix. $\mathbf{B}^\mathsf{T}\mathbf{A}^\mathsf{T}\mathbf{x}_s$ and $\mathbf{B}^\mathsf{T}\mathbf{A}^\mathsf{T}\mathbf{x}_c$ are the LDA and WCCN projected i-vectors for target speaker s and test c, respectively. 36

4.2 The grid for arranging the i-vectors of training speakers. 41

5.1 The procedure of utterance partitioning with acoustic vector resampling (UP-AVR). Note that randomization and creation of target-speaker's supervectors can be repeated several times to obtain a sufficient number of target-speaker's supervectors. 55

5.2 Scores produced by SVMs that use one or more speaker-class supervectors (SVs) and 250 background SVs for training. The horizontal axis represents the training/testing SVs. Specifically, SV-1 to SV-15 were obtained from 15 utterances of the same target-speaker, and SV-16 to SV-65 were obtained from the utterances of 50 impostors. For (B) to (F) in the legend, SV-1 was used for training and SV-2 to SV-15 were used as speaker-class test vectors, i.e., the score of SV-1 is the training-speaker score, whereas the scores of SV-2 to SV-15 are test speaker scores. For (A), SV-1 to SV-5 were used for training and SV-6 to SV-15 were used for testing. In (C), 10 speaker-class SVs were generated by a Gaussian generator using SV-1 as the mean vector and component-wise intra-speaker variances as the diagonal covariance matrix; whereas in (D)–(F), 1, 5, and 10 speaker-class UP-AVR SVs were obtained from the utterance corresponding to SV-1 by UP-AVR. $N = 5, R = 1$ is set for generating 5 speaker-class UP-AVR SVs and $N = 5, R = 2$ is set for generating 10 speaker-class UP-AVR SVs. Values inside the squared brackets are the mean difference between speaker scores and impostor scores. NAP [1] had been applied to the SVs. 56

5.3 The procedure of utterance partitioning with acoustic vector resampling (UP-AVR) for i-vector system. Note that index randomization, utterance partitioning, and i-vector extraction can be repeated several times to obtain a sufficient number of i-vectors. 58

5.4 Intra-speaker and inter-speaker cosine-distance scores versus utterance length for (a) 8-min interview conversation, (b) 3-min interview conversation, and (c) 5-min telephone conversation. 60

5.5 Minimum decision costs (old MinDCF) achieved by i-vectors derived from utterances of various lengths. The costs were based on the intra- and inter-speaker cosine-distances shown in Figure. 5.4. For computation efficiency, no score normalization was applied. 61

5.6 Average cosine distance scores with error bars versus the number of partitions per full-length utterance. In all cases, the number of resampling in UP-AVR was set to 1, i.e. $R = 1$. 63

5.7 Average cosine distance scores with error bars versus the number of random resampling R. In all cases, the number of partitions in UP-AVR was set to 4, i.e. $N = 4$. 64

6.1 The flow chart of i-vector/PLDA speaker verification system. $\mathbf{T}$ is the total variability matrix. Refer to the caption in Figure 4.1 for the definitions of $\mathbf{x}_s$, $\mathbf{x}_t$, $\mathbf{A}$, and $\mathbf{B}$ and Eq. 6.13 for $\mathbf{P}$ and $\mathbf{Q}$. 67

6.2 The effect of length normalization on the distribution of i-vectors. (a) The distribution of i-vectors without length normalization. (b)The distribution of i-vectors with length normalization. The normal probability plots help to check whether the data follow a Gaussian distribution. If the data are Gaussian, the plots will be linear. 70

6.3 Construction of empirical kernel maps for i-vector/PLDA speaker verification. 84

6.4 The procedure of PLDA–SVM training with empirical kernel maps. . 84

7.1 Numbers of enrollment i-vectors per target speaker with and without performing UP-AVR. Each speaker index represents one target speaker. 104

8.1 The variance of features in the GMM-supervectors with different numbers of mixture components. To facilitate comparison, the maximum variance has been normalized to 1.0. The horizontal line is the variance threshold ($= 0.05$) above which the features are deemed relevant. The numbers along the horizontal line are the number of relevant features in the supervectors. All results are based on NIST'02. 108

8.2 Histograms of mixture weights with increasing relevance factor r for the means in the K_{GM} kernel. The bottom panel shows the histogram of UBM's mixture weights. 111

8.3 (a) EER and (b) minimum DCF versus number of speaker-class supervectors used for training the speaker-dependent SVMs in NIST'02. The supervectors were obtained by utterance partitioning with acoustic vector resampling (UP-AVR, $N = 4$). (c) p-values of McNemar's tests [2] on the pairwise differences between the EERs in (a). For each entry, $p < 0.005$ means that the difference between the EERs is statistically significant at a confidence level of 99.5%. 116

8.4 Performance of UP-AVR for different numbers of partitions (N) and resampling (R) in NIST'02. When $R = 0$, UP-AVR is reduced to UP. 117

8.5 Minimum DCF versus EER demonstrating the performance improvement obtained by the utterance partitioning approach in (a) NIST'02 and (b) NIST'04. 119

8.6 The DET performance of GMM–UBM, GMM–SVM, and GMM–SVM with utterance partitioning in (a) NIST'02 and (b) NIST'04. T-Norm was applied in all cases. 120

8.7 Minimum decision cost (MinDCF) versus the number of speakers used for estimating the LDA and WCCN projection matrices. Each speaker has either 6, 8, or 10 utterances for estimating the i-vectors. . 122

8.8 EER versus the dimension after PCA projection. $M = x$ means each speaker only has x recordings for training the LDA and WCCN matrices.126

8.9 EER versus the dimension of the projected i-vectors in the first stage of PCA+LDA and multi-way LDA. The number of recordings per speaker is 8 ($M = 8$). Refer to Section 8.2.3 for the explanation of "1st stage". 127

8.10 (a) EER and (b) minimum DCF versus number of i-vectors per recording session for different numbers of recording session per speaker for training the LDA and WCCN matrices. The i-vectors were obtained by utterance partitioning with acoustic vector resampling (UP-AVR, $N = 4$; $R = 1, 2, 4$). M is the number of recordings per speaker used for training the matrices, $M = 0$ means without LDA and WCCN, and $M \geq 8$ means at least 8 utterances per speaker were used for training. 129

8.11 The effect of varying the number of partitions (N) and the number of resampling (R) on the performance of UP-AVR. $R = 0$ means without applying UP-AVR to the utterances. 130

8.12 Equal Error Rate versus the number of enrollment i-vectors per target speaker (H_s in Eq. 6.21) for constructing the empirical kernel maps. *Config 1*: Each target speaker has at least 15 enrollment utterances and the i-vectors were obtained from these utterances without applying UP-AVR. *Config 2*: Each target speaker has one enrollment utterance only and all enrollment i-vectors (for SVM training and empirical kernel map construction) were generated by applying UP-AVR on this utterance. 136

8.13 Minimum normalized DCF versus the number of enrollment i-vectors per target speaker (H_s in Eq. 6.21) for constructing the empirical kernel maps. See the caption in Figure 4 for *Config 1* and *Config 2* in the legend. 137

8.14 SNR distribution of test utterances in CC2, CC4, and CC5 of NIST'12.141

8.15 The DET performance of PLDA (I-vector Avg.), PLDA (Score Avg.), and PLDA–SVM scoring under CC2 in NIST'12. See Chapter 8.3.4 for the nomenclature of methods in the legend. 143

8.16 Minimum Norm DCF (2012) versus EER demonstrating the performance improvement obtained by the utterance partitioning approach in CC2, CC4, and CC5. 145

8.17 The property of empirical kernel maps in SVMs and RVM regressions. Gamma is the RBF parameter γ. 146

8.18 Histograms of the RBF parameter γ in target-speakers' RVMs (top panel) and SVMs (bottom panel). 147

LIST OF TABLES

7.1 The number of male target-speaker trials and impostor trials and the speech types for training and testing under the common conditions that involve microphone recordings in the core set of NIST 2010 SRE. *Mic*: Combining the trials of all common conditions that involve microphone recordings. *phn-mic*: Telephone conversation recorded by microphones. *int-mic*: Interview sessions recorded by microphones. . . 94

7.2 The number of male target-speaker trials and impostor trials and the speech types for training and testing under the common conditions that involve microphone recordings in the extended core set of NIST 2010 SRE. *Mic*: Combining the trials of all common conditions that involve microphone recordings. *phn-mic*: Telephone conversation recorded by microphones. *int-mic*: Interview sessions recorded by microphones. 94

7.3 The number of male target-speaker trials and impostor trials under the common conditions that involve telephone recordings in the core set of NIST 2012 SRE. 95

7.4 The roles played by different corpora in the performance evaluations. #Only speakers with 8 or more utterances were used for estimating the NAP matrices. "System 1" represents GMM-supervector based speaker verification system (Chapter 3). "System 2" represents i-vector based speaker verification system (Chapter 4). 96

8.1 The effect of varying the number of Gaussian components in GMM-supervectors on the verification performance of the GMM–SVM baseline system (with NAP and T-norm) in NIST'02. 107

8.2 Performance of GMM–UBM and GMM–SVM in NIST'02. 108

8.3 Performance of GMM–UBM and GMM–SVM in NIST'04 (core test, all trials). 109

8.4 The performance of GMM–SVM (SVM) using three different kernels under different common conditions (CC) in NIST'10. ZT-norm was applied in all cases. "MinNDCF" represents the minimum norm DCF. 110

8.5 The performance of the K_{GM} kernel in GMM-SVM systems under CC4, CC7, and CC9 of NIST'10 with increasing value of relevance factors for both means and variances. 110

8.6 The performance of speaker comparison (SC) using three different kernels under different common conditions (CC) in NIST 2010 SRE. Except for System D, ZT-norm was applied in all cases. 110

8.7 The performance of different systems on NIST'10 under different common conditions (CC). Results were divided into three groups: (1) SVM – GMM-SVM with different kernels, (2) SC – speaker comparison using different inner-product discriminant kernels, and (3) JFA – joint factor analysis models. The lowest EER and MinNDCF across all three groups were displayed in bold. The kernels being compared include the KL-divergence kernel K_{KL}, GMM-UBM mean interval kernel K_{GUMI}, and geometric-mean comparison kernel K_{GM}. Except for Systems E, G and K, ZT-norm was applied in all cases. UP-AVR, the method proposed in this thesis, was applied to two of the GMM-SVM systems for comparison. (E)+(L) denotes the linear score fusion of the best GMM-SVM system and the JFA system. 112

8.8 Effect of varying the number of speaker-class supervectors on speaker verification performance in NIST'02. The speaker-class supervectors were generated by utterance partitioning (UP) and utterance partitioning with acoustic vector resampling (UP-AVR). The first column shows the number of speaker-class GSVs, which include the GSVs created by UP or UP-AVR and the GSV produced by the full-length utterance. "None" in the 2nd column means a full-length utterance was used to produce a single supervector for training a speaker-dependent SVM. N and R are the number of partitions per full-length utterance and the number of times resampling were performed to obtain the speaker-class GSVs. When the number of GSVs generated by UP-AVR is larger than the number of required speaker's GSV (e.g., 2nd row with UP-AVR, $N = 4$ and $R = 1$), the speaker's GSVs were randomly selected from the pool. The same utterance partitioning procedure was also applied to the background speakers' utterances so that the length of partitioned background utterances matches with that of the speaker's partitioned utterances. The number of background GSVs is $B(N+1)$, where B is the number of background speakers (112 for male and 122 for female). The numbers inside the square brackets are the 90% confidence intervals of FAR (at equal error threshold) and minimum DCF found by bootstrapping techniques [3, 4]. 114

8.9 Performance of GMM–SVM and GMM–SVM with utterance partitioning in NIST'04 (core test, all trials). The numbers inside the parentheses indicate the number of speaker-class supervectors used for training a speaker-dependent SVM, which include the supervectors generated by UP-AVR ($N = 4$, $R = 1, 8, 15, 20, 23, 25, 50$) and the full-length utterance. 118

8.10 The performance of using different methods for solving the small sample-size problem in LDA and WCCN. $M = x$ means each speaker only has x recordings for training the LDA and WCCN matrices. $M \geq 8$ means each speaker provides at least 8 recordings, with an average of 31 recordings per speaker. "LDA": the conventional LDA; "PI-LDA": pseudo-inverse LDA; "PCA + LDA": perform PCA before LDA; "UP-AVR(N)": dividing each of the full-length training utterances into N partitions using UP-AVR, with the number of resampling R set to 4. 124

8.11 The performance of Multi-way LDA and other LDA methods. *MW-LDA + LDA*: Multi-way LDA. $M = x$ means each speaker only has x recordings for training the LDA and WCCN matrices. "–" denotes the situation where singularity occurs when estimating the projection matrices. 128

8.12 The performance of i-vector based speaker verification in NIST'10 using different scoring methods. C is the SVM's penalty factor. "CC" denotes common condition. "Mic" represents all common conditions involving interview-style speech or microphone speech. "CDS": cosine distance scoring. "CDS+UP-AVR(R)": computing the average cosine distance score between the claimant's LDA+WCCN projected i-vector and $NR+1$ target-speaker's i-vectors produced by UP-AVR with number of partitions $N=4$ and number of re-sampling $R=1,2$, or 4. "SVM+UP-AVR(R)": SVM scoring with each SVM trained by using $NR+1$ target-speaker's LDA+WCCN projected i-vectors and 633 background speakers' i-vectors, where the target-speaker i-vectors were produced by UP-AVR with number of partitions $N=4$ and number of re-sampling $R=1,2$, or 4. 132

8.13 Performance of varying the number of background speakers' i-vectors for constructing empirical kernel maps under CC2 of NIST'12. The first column represents the number of background speakers' i-vectors for constructing the empirical kernel maps. 138

8.14 Performance (CC2 in NIST'12) of PLDA-SVM scoring using different types of non-target speakers for training the SVMs. 139

8.15 Performance (core set, male speakers) of different PLDA training methods for different common conditions of NIST'12. 140

8.16 Performance of scoring methods for the multiple enrollment utterances in NIST'12 under the common conditions that involve telephone recordings. 142

8.17 Performance comparison between SVM and RVM in common condition 2 of NIST'12. *RVM-C* represents relevance vector machine classification. *RVM-R* represents relevance vector machine regression. *UP-AVR* represents utterance partitioning with acoustic vector resampling [5]. The methods are named by the processes applied to the i-vectors for computing the verification scores. For example, *PLDA+UP-AVR+SVM-II* means that UP-AVR has been applied to create target-speaker i-vectors for training PLDA-SVMs (Eq. 6.27) that use empirical kernel Map II (Eq. 6.24). 148

LIST OF ABBREVIATIONS

CC	Common evaluation condition
CDS	Cosine distance scoring
DCF	Detection cost function
DCT	Discrete cosine transform
DET	Detection error tradeoff
DFT	Discrete Fourier transform
EER	Equal error rate
EM	Expectation maximization
FAR	False accept rate
FRR	False reject rate
GMM	Gaussian mixture model
GSVs	GMM-supervectors
JFA	Joint factor analysis
LDA	Linear discriminant analysis
LR	Likelihood ratio
MAP	Maximum a posteriori
MFCC	Mel-frequency cepstral coefficient

NAP	Nuisance attribute projection
NIST	National institute of standards and technology
PCA	Principal component analysis
PLDA	Probabilistic linear discriminant analysis
RBF	Radial basis function
ROC	Receiver operating characteristic
RVM	Relevance vector machine
SC	Speaker comparison
S-Norm	Symmetric normalization
SNR	Signal-to-noise ratio
SRE	Speaker recognition evaluation
SVs	Supervectors
SVM	Support vector machine
T-Norm	Test normalization
UBM	Universal background model
UP-AVR	Utterance partitioning with acoustic vector resampling
VAD	Voice activity detector
WCCN	Within-class covariance normalization
Z-Norm	Zero normalization

Chapter 1

INTRODUCTION

Biometric authentication has recently attracted increasing attention from the public because of its simple, quick, and user-friendly features. Biometric authentication is an automatic process for verifying or identifying the identity of a person based on his/her physiological or behavioral characteristics. Typically, biometric systems extract physiological and behavioral characteristics from fingerprints, irises, retinas, hands, voice, faces, handwriting, keystroke, and gait. Among these characteristics, voice is the least intrusive and is the easiest to obtain. Therefore, voice biometrics is well accepted by users.

Voice biometrics, also known as speaker recognition, can be divided into speaker identification and speaker verification. The former is to identify a speaker from a known set of speakers and the latter is to verify the identity of a claimed speaker. This thesis focuses on speaker verification.

1.1 Speaker Verfication

Speaker verification [6] is to verify the identity of a claimant based on his/her own voices (Figure 1.1). It can be divided into two categories: text-dependent and text-independent. The former requires speakers to speak the same set of keywords for enrollment and verification. The latter is more flexible and does not have any restriction on the contents of the utterances. Therefore text-independent speaker verification has more applications. This thesis focuses on text-independent speaker verification.

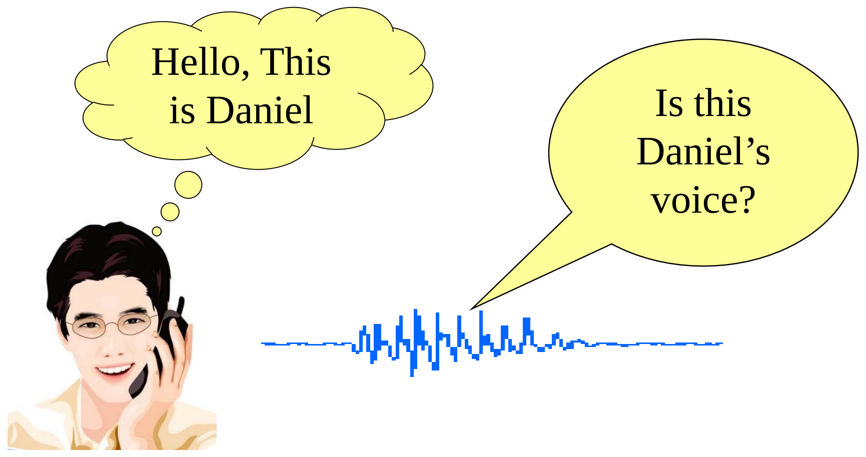


Figure 1.1: The task of speaker verification.

Speaker verification can also be divided into close-set speaker verification and open-set speaker verification. If it is a priori known that the claimant is from a group of registered speakers, we have close-set speaker verification; otherwise, the verification is an open-set one. Research has shown that the performance of close-set speaker verification is better than that of open-set speaker verification. However, open-set speaker verification is much more practical. Among all of the NIST[1] Speaker Recognition Evaluations (SREs), only the NIST 2012 SRE [7] involves open-

[1]NIST is the abbreviation of American National Institute of Standards and Technology.

set verification; all the evaluations prior to 2012 involve close-set verification.

Speaker verification systems can be used for (1) transaction authentications in credit-card purchases, stock trading and telephone banking, (2) access controls of physical premises and computer networks, and (3) forensic investigations in court cases.

With decades of research and development, the performance of speaker verification systems in laboratory settings is already very good. However, their performance degrades rapidly when they are deployed in real-world environments. This is because real-life environments are very complicated. There are a number of factors that affect performance, including [8, 9]: (1) duration of test segments; (2) noise effects, (3) channel effect and mismatch, and (4) emotion status of speakers. This thesis is related to the first three factors.

1.2 Motivation of the Thesis

Nowadays, state-of-the-art speaker verification systems are mainly based on Gaussian mixture model and support vector machines (GMM–SVM) [10] or i-vectors [11] with probabilistic linear discriminant analysis (PLDA) [12–14]. Previous experimental results have demonstrated the excellent performance of these techniques. However, there are some limitations in these speaker verification systems. The limitations are elaborated below.

1. *Data imbalance in SVM training.* For GMM–SVM systems, the training of the speaker-dependent SVMs is fairly unusual in that typically only a few or sometimes even only one enrollment utterance is available. While in the latest NIST SRE, multiple enrollment utterances per speaker are available for training, not all speakers have multiple utterances. Some speakers only have one enrollment

utterance. Given that the number of background-speaker utterances is typically several hundreds, the limited number of enrollment utterances leads to a severe data imbalance problem. An undesirable consequence of data imbalance is that the orientation of the SVM's decision boundaries in GMM–SVM systems are largely dictated by the data in the majority (background speakers) class [15].

2. *Small sample-size problems in i-vector systems.* The idea of i-vectors is to represent the characteristics of an utterance through the latent factors of a factor analyzer. Because the factor loading matrix of the factor analyzer defines the possible speaker- and channel-variability of i-vectors, it is important to suppress the unwanted channel variability. Linear discriminant analysis (LDA) [16], within-class covariance normalization (WCCN) [17], and PLDA are commonly used for such purpose. These methods, however, require training data comprising many speakers each providing sufficient recording sessions for good performance. Performance will suffer when the number of speakers and/or number of sessions per speaker are too small. In other words, When the number of training speakers and/or number of recording sessions per speaker are insufficient, numerical difficulty or error will occur in estimating the transformation matrices, resulting in inferior performance. In machine learning literatures, this is known as the small sample-size problem [18, 19].

3. *Implicitly utilization of background information in i-vector/PLDA framework.* Given the i-vector of a test utterance and the i-vectors of the claimed target-speaker, PLDA scoring [12–14] and cosine distance scoring [11] compute the scores without referring to any other speakers. Taking PLDA scoring as an

example, the verification decision is based on the likelihood ratio score derived from two hypotheses: (1) the test i-vector and the target-speaker i-vector are from the same speaker and (2) these two i-vectors are from two different speakers. Because the computation of the likelihood ratio does not involve other i-vectors, this scoring method *implicitly* uses background information through the universal background model (UBM) [20], the total variability matrix [11], and the PLDA's factor loading matrix. While this likelihood ratio scoring method is computationally efficient, the implicit use of background information is a drawback of this method.

1.3 Contribution of the Thesis

This thesis proposes a technique called utterance partitioning with acoustic vector resampling (UP-AVR) [5, 15, 21, 22] to alleviate the data imbalance problem and small sample-size problem mentioned in Section 1.2. The idea is to partition an enrollment utterance into a number of sub-utterances and to resample the acoustic vectors to produce more GMM-supervectors [10] or i-vectors for training the target-speakers' SVMs and the LDA models. Specifically, the silence regions of a long conversation are firstly removed. Then, the speech frame indexes of the conversation are randomized. Finally, the randomized frame-sequence is partitioned into equal-length segments, with each segment independently used for estimating a GMM-supervector or i-vector. This frame-index randomization and partitioning process can be repeated several times to produce a desirable number of GMM-supervectors or i-vectors for each conversation.

Another channel compensation method called multi-way LDA [22] is also proposed in this thesis to solve the small sample-size problems. In the classical i-vector approach, covariance analysis is only applied to the speaker domain for computing the within-speaker scatter matrix and between-speaker scatter matrix. The assumption is that each training i-vector has a speaker label and that each speaker provides a number of utterances (i-vectors) using a variety of microphones. This approach, however, ignores the fact that the same set of microphones are used in the recording sessions for all training speakers. Multi-way LDA is to exploit the extra information that can be found in the microphone labels to strengthen the discriminative capability of LDA and to solve small sample-size problem.

To address the limitation of PLDA scoring and cosine distance scoring, we have recently proposed to apply SVM scoring with empirical kernel maps for taking the background speaker information *explicitly* during the scoring process [23–25]. This method captures the discrimination between a target-speaker and non-target-speakers in the SVM weights as well as in the score vectors that live in an empirical score space. Specifically, for each target speaker, an empirical score space with dimension equal to the number of training i-vectors for this target speaker is defined by using the idea of empirical kernel maps [26–28]. Given an i-vector, a score vector living in this space is formed by computing the LR scores of this i-vector with respect to each of the training i-vectors. A speaker-dependent SVM – referred to as PLDA-SVM – can then be trained using the training score vectors. During verification, given a test i-vector and the target-speaker under test, the LR scores are mapped to a score vector, which is then fed to the target-speaker's SVM to obtain the final test score. The empirical kernel map proposed in this thesis is also a novel way of incorporating multiple enrollment i-vectors in the scoring process.

1.4 Thesis Organization

The thesis is organized as follows. Chapter 2 provides an overview of speaker verification and introduces four important modules in typical speaker verification systems: acoustic feature extraction, speaker modeling, score normalization, and performance evaluation metrics.

Chapter 3 reviews the theory of GMM-supervector based speaker verification. The drawbacks of GMM–SVM are also discussed.

Chapter 4 reviews the theory of i-vector based speaker verification including i-vector extraction, channel compensation, and scoring. The small sample-size problem in the i-vector framework will also be discussed.

Chapter 5 presents the traditional methods for solving the problems mentioned in Chapter 3 and Chapter 4 and introduces the proposed method – UP-AVR – to solve these problems.

Chapter 6 reviews the theory of Gaussian PLDA and its limitations and introduces our proposed methods to overcome the limitation: PLDA-SVM scoring and PLDA-RVM scoring.

Chapter 7 describes the arrangement of speech data in different NIST SREs and presents the details of experimental setup.

Chapter 8 presents and discusses the experimental results on the NIST SRE datasets.

Chapter 9 concludes the finding of the thesis and provides a brief outline of future research.

Chapter 2

SPEAKER VERIFICATION

Speaker verification is a binary classification task in which the objective is to determine whether a given utterance was spoken by the speaker whose identity is being claimed. This chapter present an overview of speaker verification including the theoretical background related to speaker verification, score normalization, and performance evaluation metrics.

2.1 Overview of Speaker Verification Systems

As illustrated in Figure 2.1, the speaker verification process comprises two phases: enrollment and verification. During enrollment, a user of the system provides one or more utterances. Then, feature extraction is performed to transform the speakers' utterances into speaker-specific feature vectors that are used for training a speaker model. This phase is always done offline. During verification, a speaker (called claimant) claims his/her identity and gives one or more utterances. The feature vectors extracted from these utterances are compared with the model of the claimed identity and background speaker model to produce a score. If the score is larger than a decision threshold, the claimant will be accepted; otherwise, the claimant will be rejected. Because this phase is performed online, the algorithm involved should be computationally efficient.

According to Figure 2.1, there are three important steps in a speaker verification

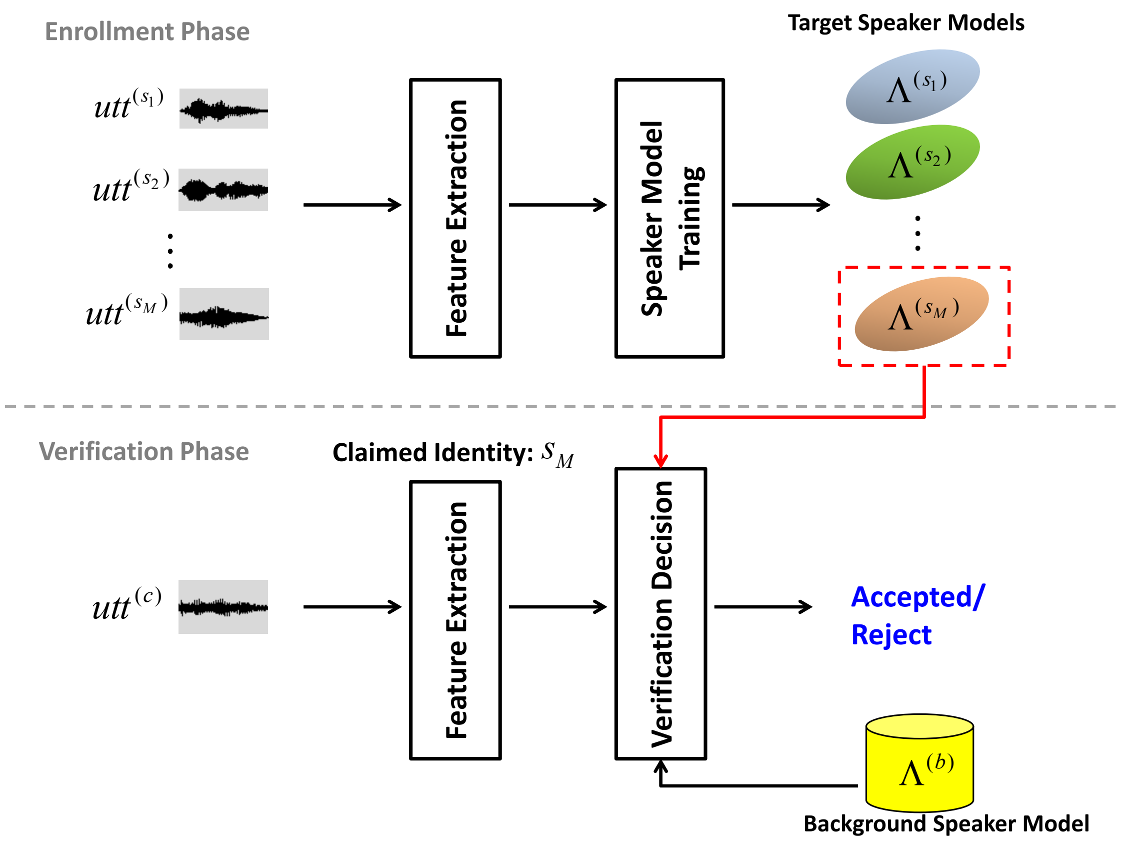


Figure 2.1: The enrollment and verification phases of a speaker verification system.

system.

- Feature extraction: It is the first step of speaker verification, which is also referred to as front-end processing. Its main goal is to extract speaker-specific feature vectors from speech signals. The criteria of ideal features [8, 29] are informativeness, invariance, uniqueness and ease of processing.

   1. **Informativeness**: informativeness means that the features should have high discriminative power among speakers. To have this property, the intra-speaker variability should be small and the inter-speaker variability

should be large.

2. **Invariance**: the features should be robust against noise and channel distortion.

3. **Uniqueness**: imitation of the features should be difficult.

4. **Ease of processing**: the features should be easy to obtain by using ordinary hardware, and software implementation should be simple.

Speech features can be categorized into short-term spectral features, voice source features, spectro-temporal features, prosodic features and high-level features [8]. In this thesis, only short-term spectral features are used. Section 2.2 will introduce this type of features in more detail.

- Speaker Model Training: It is also known as speaker modeling. The idea of speaker modeling is to train a speaker-dependent model given the feature vectors of a client speaker. Speaker models can be divided into generative models and discriminative models [8], which will be described in Section 2.3.

- Verification Decision: Decision making involves scoring computation, score normalization [30], and threshold determination. This step produces a score by comparing the claimant's speech features against the claimed speaker model and the background speaker model. The common scoring methods are likelihood-ratio scoring, SVM scoring, cosine distance scoring and PLDA scoring.

2.2 Acoustic Features for Speaker Verification

Feature extraction is an important part of a speaker verification system. Because unreliable speaker features will lead to poor speaker models and incorrect verification

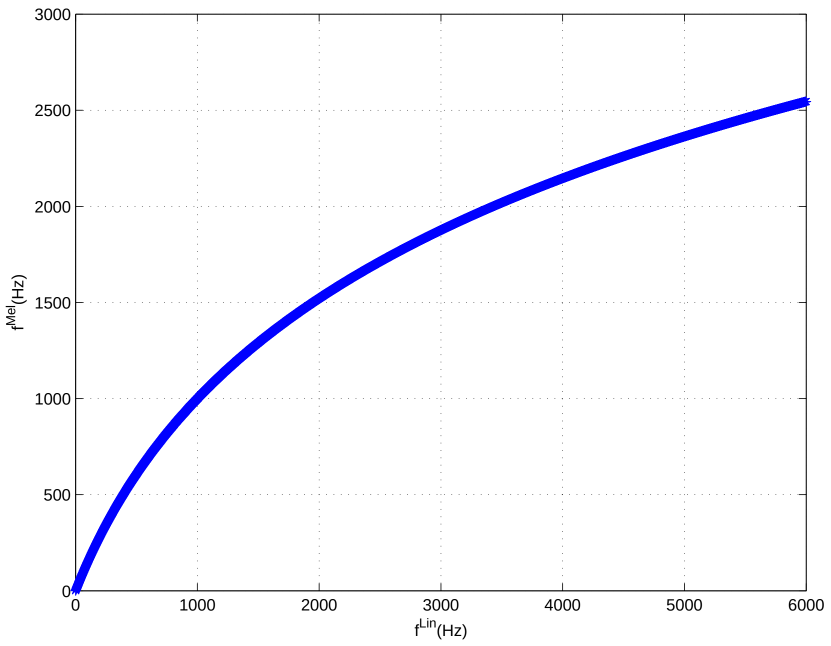


Figure 2.2: The relationship between the Mel-frequencies (f^{Mel}) and linear frequencies (f^{Lin}).

scores. Mel-frequency cepstral coefficients (MFCCs) [31] is one of the most popular short-term spectral features for speaker verification [32].

In 1940, Stevens and Volkman [33] found that the frequencies of sound are non-linearly related to the frequencies that human perceive from the sound. The relationship can be formulated as [34]:

$$f^{Mel} = 2595 \log_{10} \left(1 + \frac{f^{Lin}}{700} \right) \tag{2.1}$$

where f^{Mel} is the Mel-frequency and f^{Lin} is the linear frequency, both in the unit Hz. Figure 2.2 shows the relationship between f^{Lin} and f^{Mel}. Based on this finding, MFCCs was proposed in [31, 32]. Figure 2.3 illustrates the procedure for extracting MFCCs from speech signals.

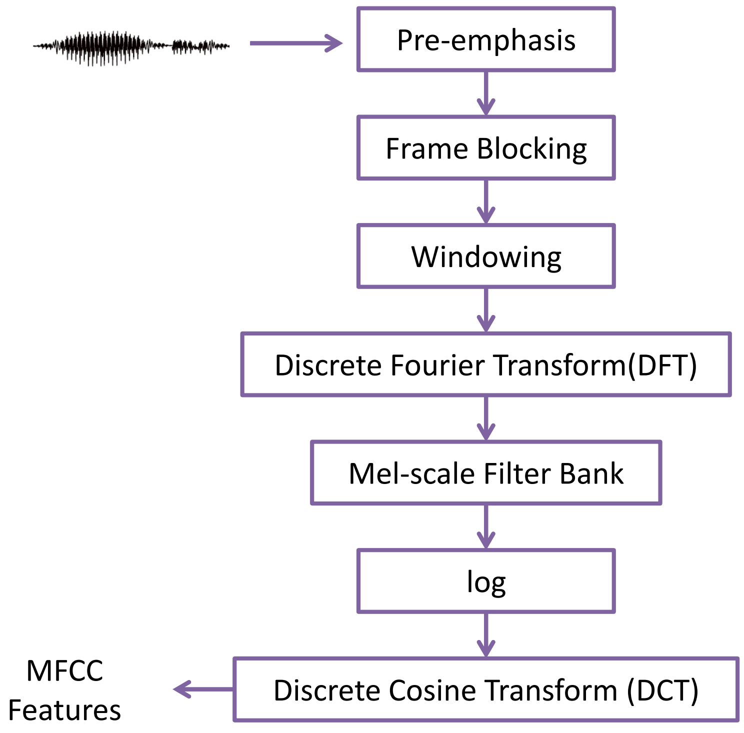


Figure 2.3: The flow chart of extracting MFCCs.

2.3 *Speaker Modeling*

Speaker modeling can be categorized into generative modeling and discriminative modeling [8]. Generative modeling is to use a statistical model to approximate or to fit the distribution of the feature vectors arising from a speaker. Gaussian mixture model [20, 35] is the most commonly used statistical model. In contrast, discriminative modeling is to model the decision boundary between speakers. The most popular example is support vector machines [10]. This chapter will introduce the classical speaker modeling approach: GMM–UBM. The more advanced approaches such as GMM-supervector and i-vectors will be covered in Chapter 3 and Chapter 4, respectively.

2.3.1 Gaussian Mixture Models

Gaussian distributions, also known as normal distributions, are one of the most popular continuous probability distributions for modeling speech features. Given a D-dimensional feature vector $\mathbf{o}$, the likelihood is given by

$$p(\mathbf{o}) = \frac{1}{(2\pi)^{D/2}\left|\mathbf{\Sigma}\right|^{1/2}} \exp\left\{-\frac{1}{2}(\mathbf{o}-\boldsymbol{\mu})^{\mathsf{T}}\mathbf{\Sigma}^{-1}(\mathbf{o}-\boldsymbol{\mu})\right\} \tag{2.2}$$

where $\boldsymbol{\mu}$ is a $D \times 1$ mean vector and $\mathbf{\Sigma}$ is a $D \times D$ covariance matrix. The advantages of Gaussian distributions are differentiable and symmetric. However, most real speech data can not be adequately modeled by a single Gaussian distribution [16]. Therefore, Gaussian mixture models (GMM) [35–37] are proposed to overcome the limitation. Specifically, an M-mixture GMM comprises M Gaussian components with mean $\{\boldsymbol{\mu}_i\}_{i=1}^{M}$ and covariance $\{\mathbf{\Sigma}_i\}_{i=1}^{M}$. Given sufficient number of Gaussians (M is large enough), speech data can be modeled accurately.

The density function of an M-mixture GMM is given by

$$p(\mathbf{o}|\mathbf{\Lambda}) = \sum_{i=1}^{M} w_i p_i(\mathbf{o}) \tag{2.3}$$

where w_i is the prior probability of the i-th Gaussian component and $p_i(\mathbf{o})$ is the component likelihood (refer to Eq. 2.2). They are also known as the mixture weights, which satisfy the constraint $\sum_{i=1}^{M} w_i = 1$. The parameters of a GMM is represented by $\mathbf{\Lambda} = \{w_i, \boldsymbol{\mu}_i, \mathbf{\Sigma}_i; i = 1, \ldots, M\}$. Because MFCCs are used as the speech features and the elements of MFCC vectors are largely uncorrelated (because of the DCT), diagonal covariance matrices are usually adequate. The use of diagonal covariance greatly simplifies computation. Given a sequence of acoustic vectors

$\mathbf{O} = \{\mathbf{o}_1, \ldots, \mathbf{o}_T\}$, the log-likelihood of $\mathbf{O}$ with respect to $\boldsymbol{\Lambda}$ can be computed as:

$$\log p(\mathbf{O}|\boldsymbol{\Lambda}) = \sum_{t=1}^{T} \log p(\mathbf{o}_t|\boldsymbol{\Lambda}). \tag{2.4}$$

The parameter $\boldsymbol{\Lambda}$ of a GMM can be estimated by maximum-likelihood using the expectation-maximization (EM) algorithm [29, 38]. Specifically, given a sequence of acoustic vectors $\{\mathbf{o}_1, \ldots, \mathbf{o}_T\}$, the model parameter $\boldsymbol{\Lambda} = \{w_i, \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i\}$ are updated iteratively as follows:

- E-step: the posterior probability of mixture components are computed:

$$\Pr(i|\mathbf{o}_t) = \frac{w_i p_i(\mathbf{o}_t)}{\sum_{j=1}^{M} w_j p_j(\mathbf{o}_t)} \qquad i = 1, \ldots, M. \tag{2.5}$$

- M-step: the new mixture weights, means, and covariance matrices of the mixture components are calculated as follows:

$$\begin{aligned} w_i &= \frac{1}{T} \sum_{t=1}^{T} \Pr(i|\mathbf{o}_t) \\ \boldsymbol{\mu}_i &= \frac{\sum_{t=1}^{T} \Pr(i|\mathbf{o}_t)\mathbf{o}_t}{\sum_{t=1}^{T} \Pr(i|\mathbf{o}_t)} \\ \boldsymbol{\Sigma}_i &= \frac{\sum_{t=1}^{T} \Pr(i|\mathbf{o}_t)\left(\mathbf{o}_t - \boldsymbol{\mu}_i\right)\left(\mathbf{o}_t - \boldsymbol{\mu}_i\right)^{\mathsf{T}}}{\sum_{t=1}^{T} \Pr(i|\mathbf{o}_t)} \end{aligned} \tag{2.6}$$

The E- and M-steps are repeated until converge. Usually, the K-means clustering algorithm [16] is used to initialize the model parameters.

2.3.2 GMM–UBM

The idea of GMM–UBM [20], which is the classical speaker verification framework, is to create a target-speaker's GMM via maximum *a posteriori* (MAP) adaptation of a universal background model (UBM). Figure 2.4 illustrates the enrollment and verification phases of a typical GMM–UBM system. Specifically, given an enrollment utterance with acoustic vector sequence $\mathbf{O} = \{\mathbf{o}_1, \ldots, \mathbf{o}_T\}$, the following formulae are applied to the mean vectors $\boldsymbol{\mu}_i$ of the UBM to obtain the adapted mean vectors $\boldsymbol{\mu}_i^{(s)}$:

$$\begin{aligned}
\boldsymbol{\mu}_i^{(s)} &= \alpha_i E_i(\mathbf{O}) + (1-\alpha_i)\boldsymbol{\mu}_i, \quad i = 1, \ldots, M, \\
\alpha_i &= \frac{n_i(\mathbf{O})}{n_i(\mathbf{O}) + r} \\
n_i(\mathbf{O}) &= \sum_{t=1}^{T} \Pr(i|\mathbf{o}_t) \\
E_i(\mathbf{O}) &= \frac{1}{n_i(\mathbf{O})} \sum_{t=1}^{T} \Pr(i|\mathbf{o}_t)\mathbf{o}_t \\
\Pr(i|\mathbf{o}_t) &= \frac{w_i p_i(\mathbf{o}_t)}{\sum_{j=1}^{M} w_j p_j(\mathbf{o}_t)}
\end{aligned} \tag{2.7}$$

where w_i and $p_i(\mathbf{o}_t)$ are the mixture weight and density function of the i-th mixture, respectively, and r is a relevance factor controlling the degree of adaptation.

During verification, the verification score of a claimant utterance, $\text{utt}^{(c)}$, is obtained by computing the ratio between the target-speaker likelihood and background-speaker likelihood, i.e.,

$$S_{\text{GMM–UBM}}(\text{utt}^{(c)}) = \log p(\mathbf{O}^{(c)}|\boldsymbol{\Lambda}^{(s)}) - \log p(\mathbf{O}^{(c)}|\boldsymbol{\Lambda}^{(b)}), \tag{2.8}$$

where $\mathbf{O}^{(c)}$ is the sequence of acoustic vectors (typically MFCCs [31] and their derivatives) derived from $\text{utt}^{(c)}$, and $\boldsymbol{\Lambda}^{(s)}$ and $\boldsymbol{\Lambda}^{(b)}$ are the target-speaker's GMM

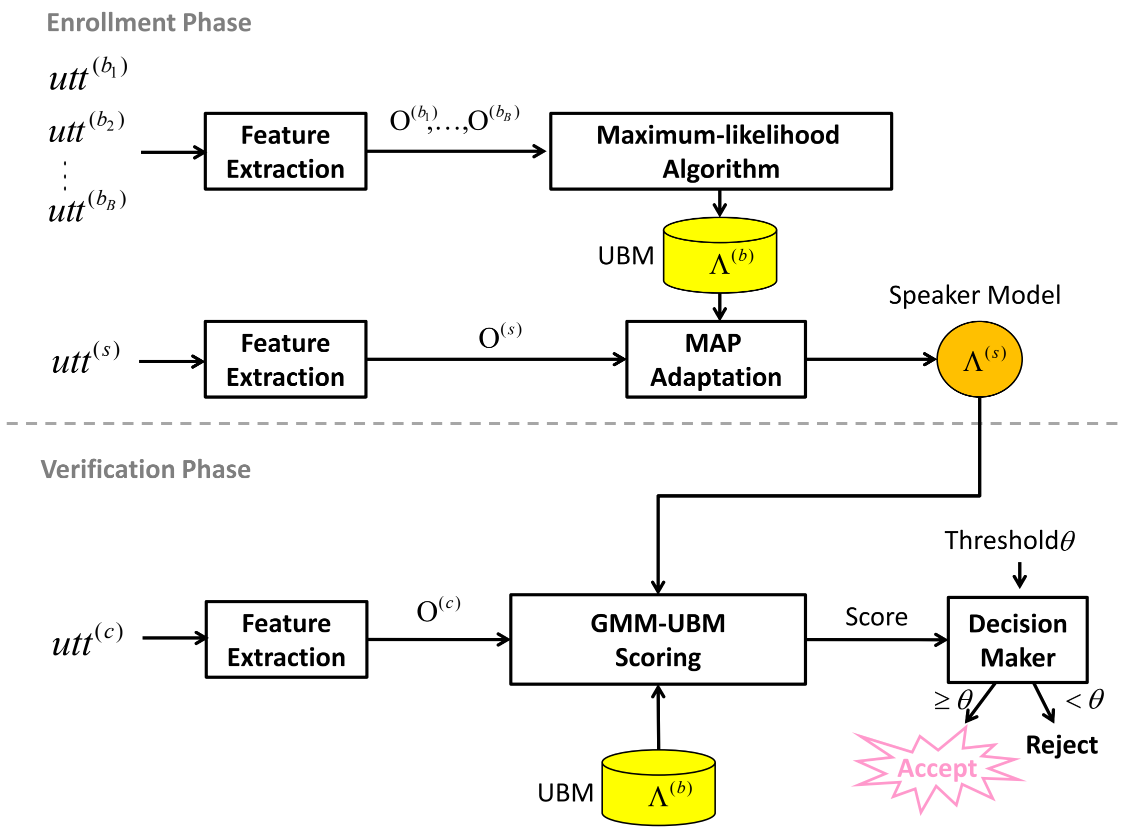


Figure 2.4: The enrollment and verification phases of GMM–UBM based speaker verification. Refer to Eq. 2.8 for GMM–UBM scoring function.

and the universal background model, respectively. Because the parameters of the two likelihood functions are estimated separately, the scoring function in Eq. 2.8 does not make full use of the discriminative information in the training data [39]. This is because when estimating the parameters of a target-speaker model in GMM–UBM systems, we do not strike to maximize the discrimination between the target-speaker's speech and impostors' speech as in discriminative training. Although a target-speaker model is adapted from the UBM, they are not computed "jointly". This concept has been explained in [39].

2.4 Score Normalization

The score in Eq. 2.8 needs to be normalized before a verification decision can be made. There are three advantages of score normalization: (1) it helps to find a global threshold for making verification decision, (2) it minimizes the effect of channel mismatches on the verification scores, and (3) it is an important step for score fusion. The most common score normalization methods [30] are zero normalization (Z-Norm), test normalization (T-Norm), ZT-Norm, TZ-Norm, and symmetric normalization (S-Norm) [40].

2.4.1 Zero Normalization (Z-Norm)

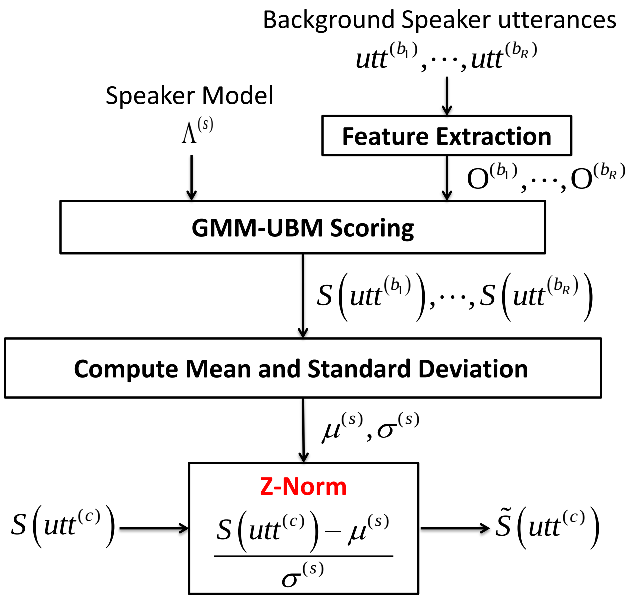

Background Speaker utterances
$utt^{(b_1)},\cdots,utt^{(b_R)}$
Speaker Model
$\Lambda^{(s)}$
Feature Extraction
$O^{(b_1)},\cdots,O^{(b_R)}$
GMM-UBM Scoring
$S(utt^{(b_1)}),\cdots,S(utt^{(b_R)})$
Compute Mean and Standard Deviation
$\mu^{(s)},\sigma^{(s)}$
Z-Norm
$S(utt^{(c)})$
$\frac{S(utt^{(c)})-\mu^{(s)}}{\sigma^{(s)}}$
$\tilde{S}(utt^{(c)})$

Figure 2.5: The procedure of Z-Norm in GMM–UBM scoring.

The idea of zero normalization (Z-Norm) [41] is to use the mean and variance of the scores that are obtained by comparing a speaker model with a set of background speaker utterances for score normalization. Specifically, given R background speaker utterances, speaker model $\mathbf{\Lambda}^{(s)}$ and a claimant utterance $\text{utt}^{(c)}$, the normalized score is:

$$\tilde{S}(\text{utt}^{(c)}) = \frac{S(\text{utt}^{(c)}) - \mu^{(s)}}{\sigma^{(s)}} \tag{2.9}$$

where $\mu^{(s)}$ and $\sigma^{(s)}$ are the mean and standard derivation of scores between speaker model $\mathbf{\Lambda}^{(s)}$ and the R background speaker utterances, $S(\text{utt}^{(c)})$ is the score of test utterance $\text{utt}^{(c)}$ against speaker model $\mathbf{\Lambda}^{(s)}$, which can be obtained by Eq. 2.8, Eq. 3.1, and the scoring methods in Chapter 4. Figure 2.5 illustrates the procedure of Z-Norm in GMM–UBM scoring.

2.4.2 Test Normalization (T-Norm)

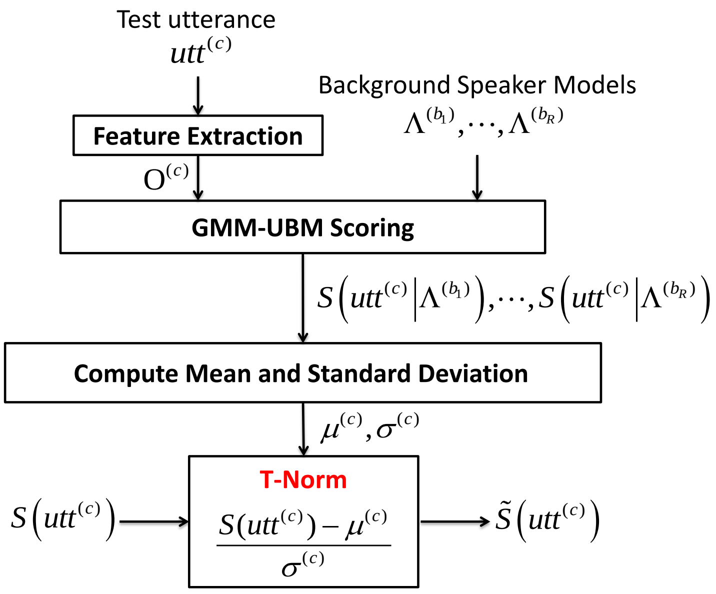

Test utterance
$utt^{(c)}$
Background Speaker Models
$\Lambda^{(b_1)},\cdots,\Lambda^{(b_R)}$
Feature Extraction
$O^{(c)}$
GMM-UBM Scoring
$S(utt^{(c)}|\Lambda^{(b_1)}),\cdots,S(utt^{(c)}|\Lambda^{(b_R)})$
Compute Mean and Standard Deviation
$\mu^{(c)},\sigma^{(c)}$
T-Norm
$\frac{S(utt^{(c)})-\mu^{(c)}}{\sigma^{(c)}}$
$S(utt^{(c)})$
$\tilde{S}(utt^{(c)})$

Figure 2.6: The procedure of T-Norm in GMM–UBM scoring.

Test normalization (T-Norm) [30] is similar to Z-Norm in that both require a set of background speakers for distribution shift and scaling. Assume that we are given R background speaker models and a test utterance $\mathrm{utt}^{(c)}$, the normalized score is:

$$\tilde{S}(\mathrm{utt}^{(c)}) = \frac{S(\mathrm{utt}^{(c)}) - \mu^{(c)}}{\sigma^{(c)}} \tag{2.10}$$

where $\mu^{(c)}$ and $\sigma^{(c)}$ are the mean and standard derivation of scores between the test utterance and R background speaker models. Figure 2.6 illustrates the procedure of T-Norm in GMM–UBM scoring.

2.4.3 ZT-Norm and TZ-Norm

Z-Norm and T-Norm can be combined for score normalization, which result in ZT-Norm or TZ-Norm [42]. ZT-Norm is to perform Z-Norm followed by T-Norm, and TZ-Norm is to perform T-Norm followed by Z-Norm. Research has found that applying ZT-Norm or TZ-Norm can achieve better performance than applying either Z-Norm or T-Norm alone.

2.4.4 Symmetric Normalization (S-Norm)

Symmetric normalization (S-Norm) [40] was proposed to harness the symmetry in cosine similarity scoring. Given a set of background speakers and two utterances utt_i and utt_j, the S-Norm score between utt_i and utt_j is:

$$\tilde{S}(utt_i, utt_j) = \frac{S(utt_i, utt_j) - \mu_i}{\sigma_i} + \frac{S(utt_i, utt_j) - \mu_j}{\sigma_j} \tag{2.11}$$

where $S(utt_i, utt_j)$ is the score between utt_i and utt_j, μ_i and σ_i are the mean and standard derivation of scores between utt_i and the set of background speakers, μ_j

and σ_j are the mean and standard derivation of scores between utt_j and the set of background speakers.

2.5 Performance Evaluation Metrics

To evaluate the performance of speaker verification systems, two types of error rates are typical used:

1. *False Rejection Rate (FRR)* $(P_{Miss|Target})$. The percentage of falsely rejecting the true speakers; this is also known as the miss probability:

$$P_{Miss|Target} = \frac{N_{Miss}}{N_{Target}} \tag{2.12}$$

   where N_{Miss} is the number of false rejections and N_{Target} is the number of target speakers.

2. *False Acceptance Rate (FAR)* $(P_{FalseAlarm|Nontarget})$. The percentage of falsely accepting the imposters as true speakers; this is also known as the false alarm probability:

$$P_{FalseAlarm|Nontarget} = \frac{N_{FalseAlarm}}{N_{Nontarget}} \tag{2.13}$$

   where $N_{FalseAlarm}$ is the number of false acceptances and $N_{Nontarget}$ is the number of non-target speakers (also called imposters).

A good speaker verification system should have low FAR and low FRR; however, it is difficult to keep both error rate low at the same time.

There are other performance measures for speaker verification systems: equal error rate [43], detection cost function [43], and detection error tradeoff curves [29]. They are elaborated below.

2.5.1 Equal Error Rate

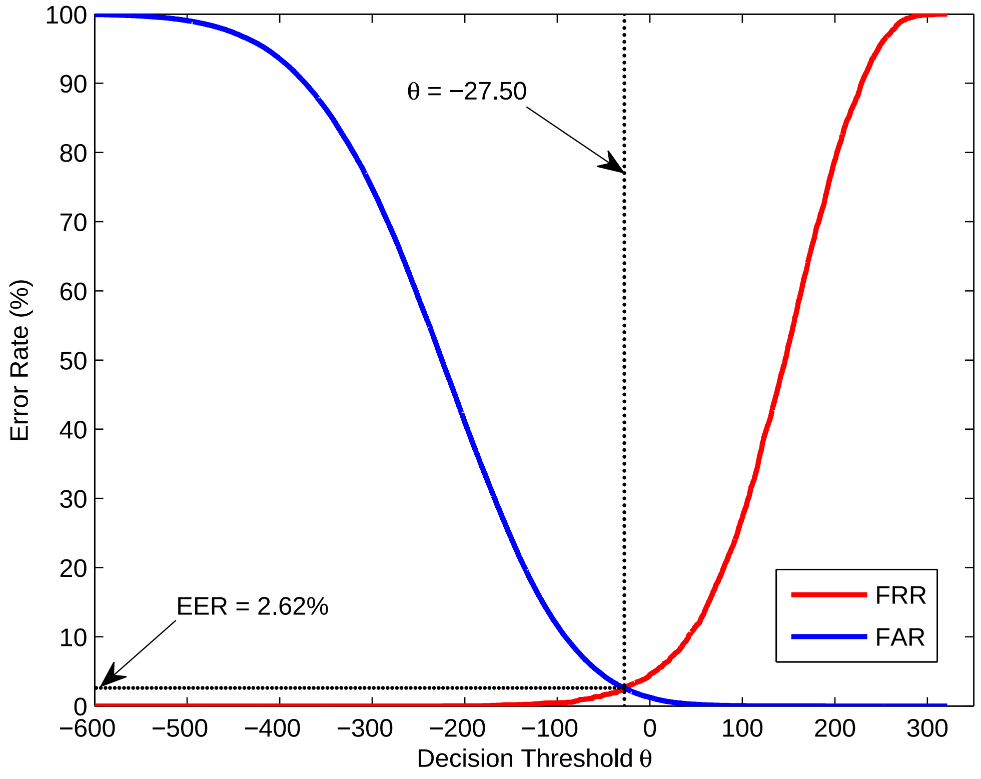


Figure 2.7: FRR and FAR with respective to the decision threshold. An EER of 2.62% is obtained when the decision threshold is equal to −27.50.

Equal error rate (EER) is used to help settling a trustworthy threshold for a speaker verification system, which is the value at which the false reject rate is equal to the false accept rate, i.e., the chance of falsely rejecting a true speaker is equal to the chance of falsely accepting an imposter. Figure 2.7 shows the relationship between FAR, FRR, EER, and the threshold at EER.

2.5.2 Detection Cost Function

Before NIST 2012 SRE, the detection cost function (DCF) is defined as:

$$\mathrm{DCF} = C_{Miss} * P_{Miss|Target} * P_{Target} + C_{FalseAlarm} * P_{FalseAlarm|Nontarget} * P_{Nontarget} \tag{2.14}$$

where

$$P_{Nontarget} = 1 - P_{Target}$$

P_{Target} and $P_{Nontarget}$ are the prior probability of target and non-target speakers, respectively, and where C_{Miss} and $C_{FalseAlarm}$ are the costs of miss and false alarm errors, respectively. For the NIST speaker recognition evaluations before 2008, $C_{Miss} = 10, C_{FalseAlarm} = 1$, and $P_{Target} = 0.01$. For evaluations after 2008, $C_{Miss} = 1, C_{FalseAlarm} = 1$, and $P_{Target} = 0.001$.[1] To improve the intuitive meaning of DCF, the normalized DCF_{norm} is proposed:

$$\mathrm{DCF}_{norm} = \frac{\mathrm{DCF}}{\mathrm{DCF}_{Default}} \tag{2.15}$$

where

$$\mathrm{DCF}_{Default} = \min\left\{C_{Miss} * P_{Target}, C_{FalseAlarm} * P_{Nontarget}\right\}$$

NIST 2012 SRE introduces a new prior probability P_{Known}, which is the prior probability that the imposter is one of the target speakers in the speaker recognition

[1]http://www.itl.nist.gov/iad/mig/tests/spk/

evaluation. Therefore, the new DCF is reformulated as:

$$\begin{aligned}\widetilde{\mathrm{DCF}} = {} & C_{Miss} * P_{Miss|Target} * P_{Target} \\ & + C_{FalseAlarm} * (P_{FalseAlarm|KnownNontarget} * P_{Known} \\ & + P_{FalseAlarm|UnknownNontarget} * (1 - P_{Known})) * P_{Nontarget}\end{aligned} \tag{2.16}$$

where

$$P_{FalseAlarm|KnownNontarget} = \frac{N_{FalseAlarm|KnownNontarget}}{N_{KnownNontarget}}$$

and

$$P_{FalseAlarm|UnknownNontarget} = \frac{N_{FalseAlarm|UnknownNontarget}}{N_{UnknownNontarget}}$$

where $N_{FalseAlarm|KnownNontarget}$ is the number of falsely accepted known non-target speakers, $N_{FalseAlarm|UnknownNontarget}$ is the number of falsely accepted unknown non-target speakers, $N_{KnownNontarget}$ is the number of known non-target speakers, and $N_{UnknownNontarget}$ is the number of unknown non-target speakers. Given $P_{Known} = 0.5$ for the core conditions, the new normalized DCF is:

$$\begin{aligned}\widetilde{\mathrm{DCF}}_{norm}(\alpha) = {} & P_{Miss|Target} \\ & + \alpha * (P_{FalseAlarm|KnownNontarget} * P_{Known} \\ & + P_{FalseAlarm|UnknownNontarget} * (1 - P_{Known}))\end{aligned} \tag{2.17}$$

where

$$\alpha = \frac{C_{FalseAlarm}}{C_{Miss}} \cdot \frac{P_{Nontarget}}{P_{Target}}. \tag{2.18}$$

Therefore, the primary DCF for NIST 2012 SRE is defined by:

$$\mathrm{DCF}_{primary} = \frac{\widetilde{\mathrm{DCF}}_{norm}(\alpha_1) + \widetilde{\mathrm{DCF}}_{norm}(\alpha_2)}{2} \tag{2.19}$$

where α_1 is obtained by substituting $C_{Miss} = 1, C_{FalseAlarm} = 1$, and $P_{Target} = 0.01$ into Eq. 2.18 and α_2 is obtained by substituting $C_{Miss} = 1, C_{FalseAlarm} = 1$, and $P_{Target} = 0.001$ into Eq. 2.18.

2.5.3 Detection Error Tradeoff Curve

Detection error tradeoff (DET) curve [43] is an effective measure for comparing the performance of different speaker verification systems. The tradeoff between FAR and FRR is illustrated by a receiver operating characteristic (ROC) curve in which the scale in the FAR and FRR axes are non-linear. Figure 2.8 shows an example of DET curves. The closer the DET curve is to the origin, the better the performance of the speaker verification system.

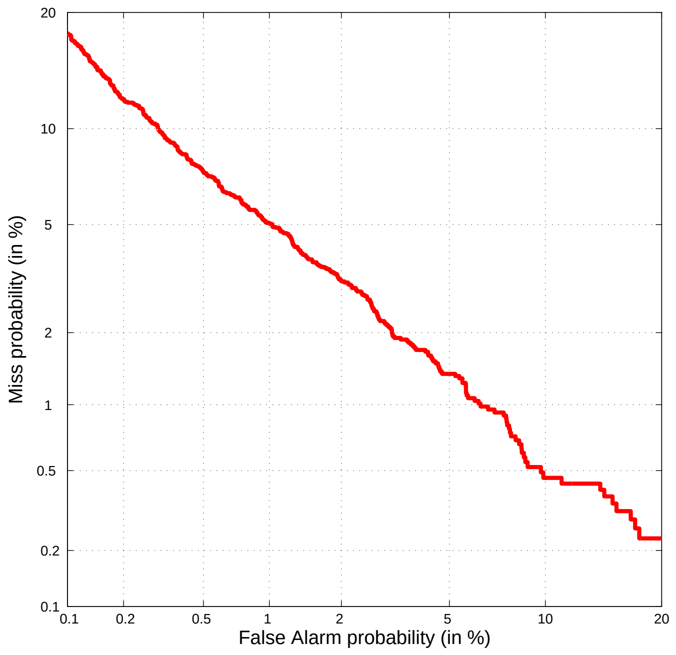


Figure 2.8: An example of DET curves.

Chapter 3

GMM-SUPERVECTOR BASED SPEAKER VERIFICATION

Speaker verification is a binary classification task. Therefore, support vector machines (SVM) are suitable for speaker verification. Recent research has demonstrated the merit of combining the GMM and SVM for text-independent speaker verification. In this chapter, we will firstly introduce GMM-supervectors. Then, we describe the concept of GMM–SVM and explain three sequence kernels that are suitable for the GMM–SVM framework. We will also explain why GMM–SVM scoring is superior to the conventional GMM-UBM likelihood ratio scoring. Finally, the issues in GMM–SVM systems will be discussed.

3.1 GMM-Supervectors

Because the feature vectors (MFCCs [31]) of the speaker class and the imposter class are highly overlap, we can not use MFCC vectors as the input to an SVM [44, 45] directly. Furthermore, the goal of speaker verification is to minimize classification error based on the whole utterance instead of minimizing the error on individual speech frames. These reasons lead to the idea of *sequence kernels* [44] – converting variable-length MFCC sequences to fixed-length vectors for classification by SVM. The fixed-length vectors are called supervectors. GMM-supervectors [10] are one type of supervectors.

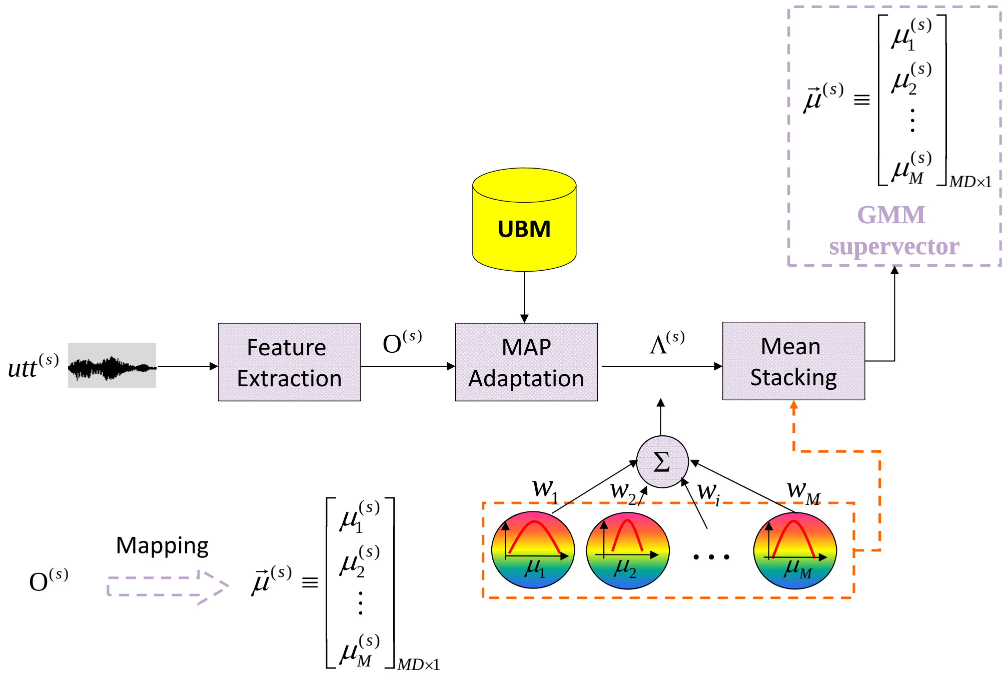

$utt^{(s)}$
Feature Extraction
$O^{(s)}$
UBM
MAP Adaptation
$\Lambda^{(s)}$
Mean Stacking
$\vec{\mu}^{(s)} \equiv \begin{bmatrix} \mu_1^{(s)} \\ \mu_2^{(s)} \\ \vdots \\ \mu_M^{(s)} \end{bmatrix}_{MD\times1}$
GMM supervector
Σ
w_1
w_2
w_i
w_M
μ_1
μ_2
μ_M
$O^{(s)}$
Mapping
$\vec{\mu}^{(s)} \equiv \begin{bmatrix} \mu_1^{(s)} \\ \mu_2^{(s)} \\ \vdots \\ \mu_M^{(s)} \end{bmatrix}_{MD\times1}$

Figure 3.1: The flow chart of GMM-supervector extraction. $\mathbf{O}^{(s)}$ is the sequence of acoustic vectors extracted from utt$^{(s)}$ of speaker s, $\mathbf{\Lambda}^{(s)}$ is the GMM model of speaker s, w_i is the i-th mixture weight of GMM model, and μ_i is the mean of i-th Gaussian component.

Figure 3.1 shows the flow chart of converting the sequence of acoustic vectors of a speaker to a GMM-supervector. Like the GMM–UBM approach, a speaker-dependent GMM is created by adapting from the UBM via MAP adaptation. However, unlike GMM–UBM, the mean vectors of the speaker-dependent GMM are stacked to form a GMM-supervector.

3.2 GMM–SVM

The idea of GMM–SVM [10] (see Figure 3.2) is to harness the discriminative information embedded in the training data by constructing an SVM that optimally separates the GMM-supervector(s) of a target speaker from the GMM-supervectors of background speakers. The target-speaker's supervector(s) together with the supervectors corresponding to individual background speakers are used to train a target-speaker SVM. Therefore, in addition to a GMM, each target speaker is also represented by an SVM that operates in a space (called GMM-supervector space) with axes corresponding to individual elements of GMM mean vectors.

In GMM–SVM, given the SVM of target speaker s, the verification score of $\text{utt}^{(c)}$ is given by

$$S_{\text{GMM–SVM}}(\text{utt}^{(c)}) = \alpha_0^{(s)} K\left(\text{utt}^{(c)}, \text{utt}^{(s)}\right) - \sum_{i \in \mathcal{S}^{(b)}} \alpha_i^{(s)} K\left(\text{utt}^{(c)}, \text{utt}^{(b_i)}\right) + d^{(s)}, \quad (3.1)$$

where $\alpha_0^{(s)}$ is the Lagrange multiplier corresponding to the target speaker,[1] $\alpha_i^{(s)}$'s are Lagrange multipliers corresponding to the background speakers, $\mathcal{S}^{(b)}$ is a set containing the indexes of the support vectors in the background-speaker set, and $\text{utt}^{(b_i)}$ is the utterance of the i-th background speaker. Note that only those background speakers with non-zero Lagrange multipliers have contribution to the score. The kernel function $K(\cdot,\cdot)$ can be of many forms. This thesis introduces three kernels. The most common kernel is the *KL divergence (KL) kernel* [1], K_{KL}:

$$K_{\text{KL}}\left(\text{utt}^{(c)}, \text{utt}^{(s)}\right) = \sum_{j=1}^{M} \left(\sqrt{w_j}\boldsymbol{\Sigma}_j^{-\frac{1}{2}}\boldsymbol{\mu}_j^{(c)}\right)^{\mathsf{T}} \left(\sqrt{w_j}\boldsymbol{\Sigma}_j^{-\frac{1}{2}}\boldsymbol{\mu}_j^{(s)}\right) \quad (3.2)$$

[1]We assume one enrollment utterance per target speaker.

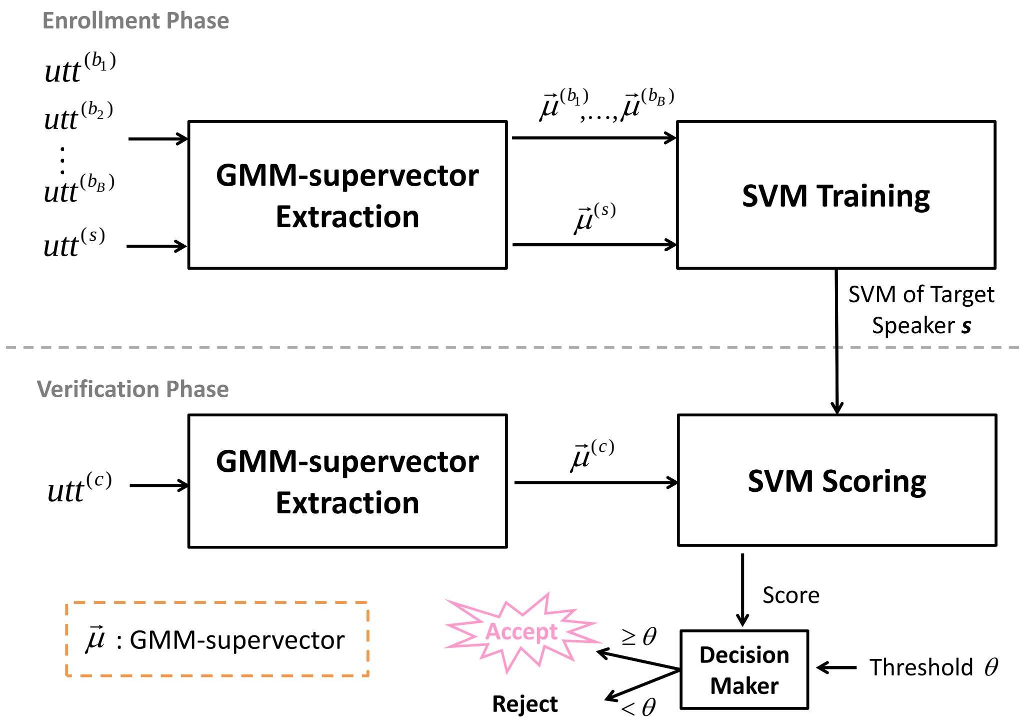


Figure 3.2: The enrollment and verification phases of GMM–SVM based speaker verification. Refer to Eq. 3.1 for the SVM scoring function.

where w_j and $\boldsymbol{\Sigma}_j$ are the mixture weight and covariance of the j-th Gaussian component of UBM, respectively, and $\boldsymbol{\mu}_j^{(s)}$ and $\boldsymbol{\mu}_j^{(c)}$ are the j-th mean vector of the GMM belonging to speaker s and claimant c, respectively. Eq. 3.2 can be written in a more compact form

$$K\left(\text{utt}^{(c)}, \text{utt}^{(s)}\right) = \left\langle \boldsymbol{\Omega}^{-\frac{1}{2}}\vec{\boldsymbol{\mu}}^{(c)}, \boldsymbol{\Omega}^{-\frac{1}{2}}\vec{\boldsymbol{\mu}}^{(s)} \right\rangle \tag{3.3}$$

where

$$\boldsymbol{\Omega} = \text{diag}\left\{w_1^{-1}\boldsymbol{\Sigma}_1, \ldots, w_M^{-1}\boldsymbol{\Sigma}_M\right\} \quad \text{and} \quad \vec{\boldsymbol{\mu}} = \left[\boldsymbol{\mu}_1^{\mathsf{T}}, \ldots, \boldsymbol{\mu}_M^{\mathsf{T}}\right]^{\mathsf{T}}. \tag{3.4}$$

M is the number of mixtures in the GMMs. In practice, $\mathbf{\Sigma}_j$'s are assumed to be diagonal.

The second kernel is the *geometric-mean-comparison (GM) kernel* [46], K_{GM}:

$$K_{\mathrm{GM}}\left(\mathrm{utt}^{(c)}, \mathrm{utt}^{(s)}\right) = \sum_{j=1}^{M} \left(\sqrt{w_j^{(c)}}\mathbf{\Sigma}_j^{-\frac{1}{2}}\boldsymbol{\mu}_j^{(c)}\right)^{\mathsf{T}} \left(\sqrt{w_j^{(s)}}\mathbf{\Sigma}_j^{-\frac{1}{2}}\boldsymbol{\mu}_j^{(s)}\right) \tag{3.5}$$

where $\mathbf{\Sigma}_j$'s are the covariance matrices of the UBM, $\boldsymbol{\mu}_j^{(s)}$ and $\boldsymbol{\mu}_j^{(c)}$ are the j-th mean vector of the GMM belonging to speaker s and claimant c, respectively, and $w_j^{(s)}$ and $w_j^{(c)}$ are the j-th mixture weight. The KL and GM kernels are different in that the mixture weights of the former are speaker-independent whereas the mixture weights of the latter are speaker-dependent.

The third kernel is the *GMM-UBM mean interval (GUMI) kernel* [47], K_{GUMI}. The GUMI kernel, which exploits the information not only from the means but also from the covariances, is derived from the Bhattacharyya distance:

$$\begin{aligned} K_{\mathrm{GUMI}}\left(\mathrm{utt}^{(c)}, \mathrm{utt}^{(s)}\right) = \sum_{j=1}^{M} &\left[\left(\frac{\mathbf{\Sigma}_j^{(c)} + \mathbf{\Sigma}_j}{2}\right)^{-\frac{1}{2}} \left(\boldsymbol{\mu}_j^{(c)} - \boldsymbol{\mu}_j\right)\right]^{\mathsf{T}} \\ &\left[\left(\frac{\mathbf{\Sigma}_j^{(s)} + \mathbf{\Sigma}_j}{2}\right)^{-\frac{1}{2}} \left(\boldsymbol{\mu}_j^{(s)} - \boldsymbol{\mu}_j\right)\right] \end{aligned} \tag{3.6}$$

where s and c represent the speaker and claimant, respectively, and $\boldsymbol{\mu}_j$ and $\mathbf{\Sigma}_j$ represent the mean and covariance of the j-th mixture of the UBM.

3.3 GMM–UBM Scoring vs. GMM–SVM Scoring

A comparison between Eq. 2.8 and Eq. 3.1 reveals that GMM–SVM scoring is superior to GMM–UBM scoring. The reason is that in GMM–UBM scoring, the information of all background speakers is represented (summarized) in a single UBM, whereas in GMM–SVM scoring, the contribution of individual background speakers and the target speaker can be optimally weighted by the Lagrange multipliers.

3.4 Nuisance Attribute Projection

Nuisance attribute projection (NAP) [1,48] is a channel compensation technique and is usually applied to SVM-based speaker verification. The idea of this technique is to remove the subspace that causes the session variability by projection.

Given a set of labeled speaker utterance $\{\text{utt}_1, ..., \text{utt}_N\}$, the kernel function with NAP can be represented by:

$$K(\text{utt}_i, \text{utt}_j) = [\mathbf{P}\phi(\text{utt}_i)] \cdot [\mathbf{P}\phi(\text{utt}_j)] \tag{3.7}$$

where

$$\mathbf{P} = \mathbf{I} - \mathbf{V}\mathbf{V}^{\mathsf{T}} \tag{3.8}$$

and $\phi(\text{utt})$ is expansion function in SVM, $\mathbf{P}$ is the projection matrix, $\mathbf{I}$ is the identity matrix, and $\mathbf{V}$ is used to define the subspace representing session variability and $\mathbf{V}^{\mathsf{T}}\mathbf{V} = \mathbf{I}$. In other words, each column of $\mathbf{V}$ is orthonormal. For obtaining the projection matrix, the following criterion function need be computed:

$$\mathbf{V}^* = \operatorname*{argmin}_{\mathbf{V}} \sum_{i,j} W_{i,j} \left\| \mathbf{P}\phi(\text{utt}_i) - \mathbf{P}\phi(\text{utt}_j) \right\|_2^2 \tag{3.9}$$

where

$$W_{i,j} = \begin{cases} 1 & i \text{ and } j \text{ belong to the same speaker} \\ 0 & \text{otherwise} \end{cases} \tag{3.10}$$

According to Eq. 3.9, we can obtain the solution of $\mathbf{V}$ by computing the eigenvalue problem:

$$\mathbf{V} = \text{eig}\left(\mathbf{\Phi}\mathbf{Z}\mathbf{\Phi}^{\mathsf{T}}\right); \tag{3.11}$$

$$\mathbf{\Phi} = [\phi(\text{utt}_1) \cdots \phi(\text{utt}_N)]; \tag{3.12}$$

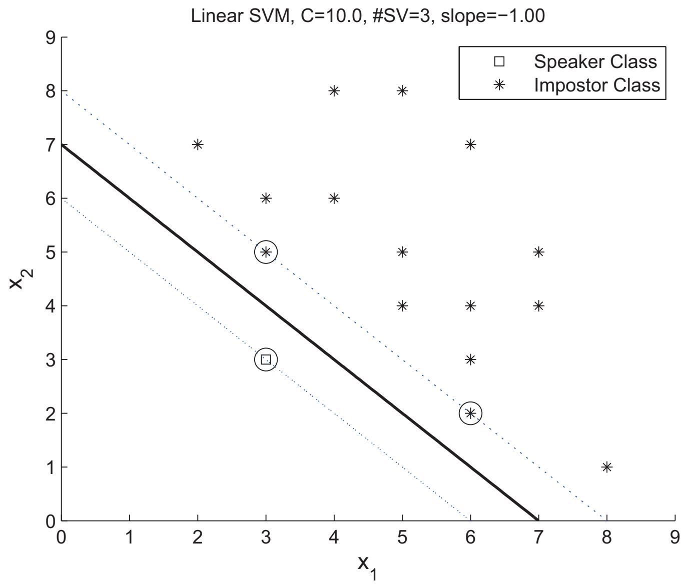
$$\mathbf{Z} = \text{diag}(\mathbf{W1}) - \mathbf{W}. \tag{3.13}$$

where $\mathbf{W}$ is a matrix containing $W_{i,j}$ and $\mathbf{1}$ is the column vector whose elements are all equal to one.

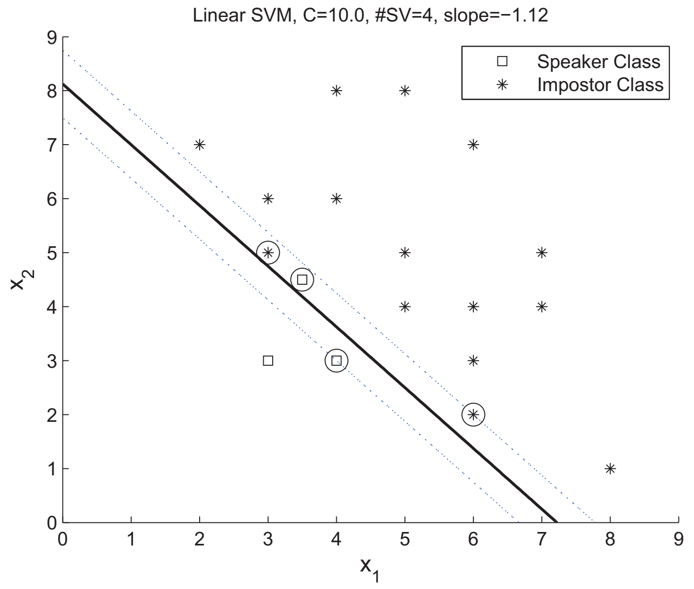
3.5 Data Imbalance Problem in GMM–SVM Systems

A major drawback of the SVM scoring approach is that the number of target speaker utterances for training the target-speaker's SVM is very limited (typically only one enrollment utterance is available). Given that the number of background speakers' utterances is typically several hundreds, the limited number of enrollment utterances leads to a severe *data imbalance problem.*

One problem of data imbalance is that the decision boundary of the resulting SVM will skew towards the minority (target speaker) class [49, 50], causing high false-rejection rate unless the decision threshold is properly adjusted to compensate for the bias. Another problem is that the orientation of the decision boundary is largely dictated by the data in the majority (background speakers) class. This situation is illustrated in Figure. 3.3(a) and Figure 3.4. Both figures show that there is a region in the feature space where the positive-class's support vector can move around without affecting the orientation of the decision boundary, but a small change in the negative-class' support vectors can tilt the decision boundary.



(a)



(b)

Figure 3.3: A two-class problem in 2-dimensional space illustrating the imbalance between the number of positive-class samples and negative-class samples. (a) The orientation (slope) of the decision boundary depends largely on the negative-class data. (b) Adding more positive-class data can enhance the influence of the positive-class data on the decision boundary (slope changes from -1.0 to -1.12). There is a region in the feature space where the positive-class's support vector (encircled $\Box$) can move around without affecting the orientation of the decision boundary, but a small change in the negative-class' support vectors (encircled $*$) can tilt the decision boundary.

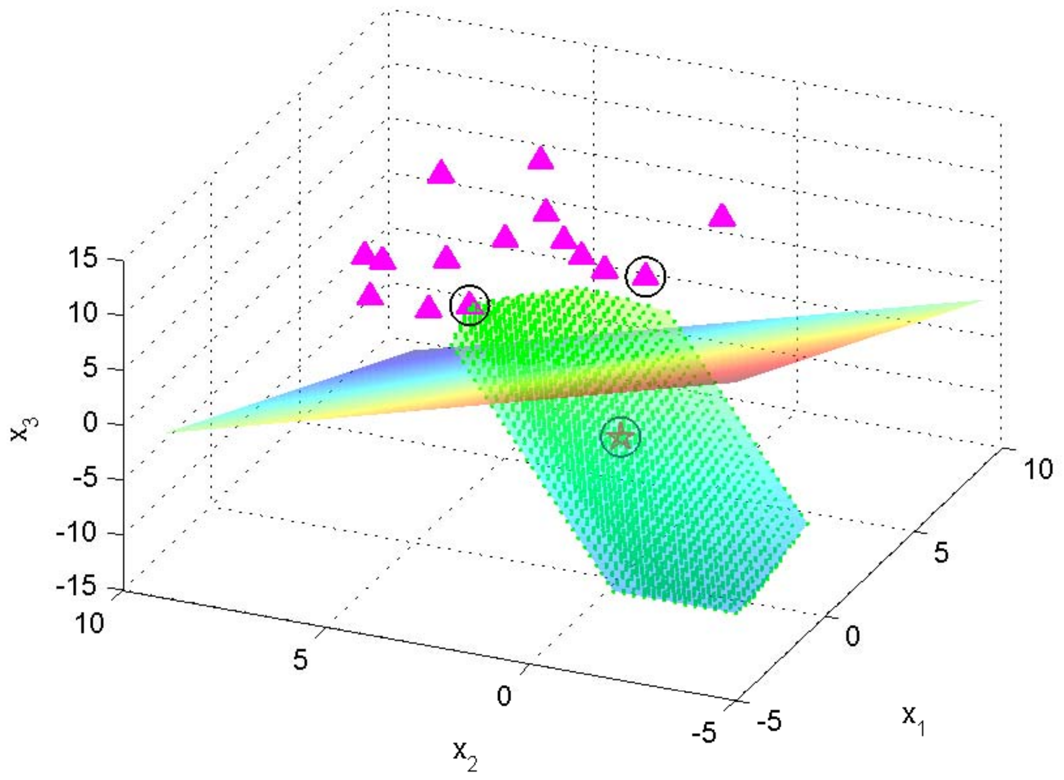


Figure 3.4: A 3-dimensional two-class problem illustrating the imbalance between the number of minority-class samples (red pentagram) and majority-class samples (pink triangles) in a linear SVM. The decision plane is defined by 3 support vectors (enclosed by black circles). The green region beneath the decision plane represents the region where the minority-class sample can be located without changing the orientation of the decision plane.

3.6 Speaker Comparison

Speaker comparison (SC) [51] computes the score of a test utterance (produced by a claimant) by evaluating the inner product between the claimant's supervector and target-speaker's supervector. The score is then compared with a threshold for making a decision:

$$\mathrm{S}_{sc}\left(\mathrm{utt}^{(c)}, \mathrm{utt}^{(s)}\right) = K\left(\mathrm{utt}^{(c)}, \mathrm{utt}^{(s)}\right) \lessgtr \quad \theta, \tag{3.14}$$

where the kernel $K(\cdot, \cdot)$ can be any valid kernel functions such as Eqs. 3.2 to 3.6. A comparison between Eq. 3.14 and Eq. 3.1 reveals that speaker comparison is a special case of GMM–SVM scoring in which no background information is used. In particular, because speaker comparison does not compute the similarity between the claimant's utterance and background utterances, score normalization (such as Z-norm) is very important for the success of this method. The experimental results in Chapter 8 also suggest that this is the case.

Unlike GMM–SVM, speaker comparison [51] does not require to train an SVM for each target speaker, thereby avoiding the data imbalance problem. An immediate question is that if speaker comparison performs better than GMM–SVM, then the proposed UP-AVR (refer to Chapter 5) becomes unnecessary. To demonstrate the value of UP-AVR, experiments have been done (see Chapter 8) to compare speaker comparison with GMM–SVM.

Chapter 4

I-VECTOR BASED SPEAKER VERIFICATION

In recent years, using i-vectors [11] as features has become one of the promising approaches to text-independent speaker verification. Unlike joint factor analysis (JFA) [52, 53] in which two distinct space (speaker space and channel space) are defined, the i-vector approach defines a single space called total variability space. The acoustic characteristics (including both speaker and channel) of an utterance are represented by a single vector called the i-vector whose elements are essentially the latent variables of a factor analyzer. Compared with the GMM-supervectors, the dimensionality of i-vectors is much lower. Therefore, statistical techniques such as linear discriminant analysis (LDA) [16], within-class covariance normalization (WCCN) [17], and probabilistic LDA (PLDA) [12] can be applied to suppress the channel- and session-variability.

Figure 4.1 shows the process of i-vector based speaker verification. The i-vector approach to speaker verification can be divided into three parts: i-vector extraction, intersession compensation and scoring. This chapter will describe the details of these three parts.

4.1 I-Vector Extraction

The i-vector approach is based on the idea of JFA [52,53]. In [11], Dehak et al. notice that the channel factors in JFA also contain speaker-dependent information. This

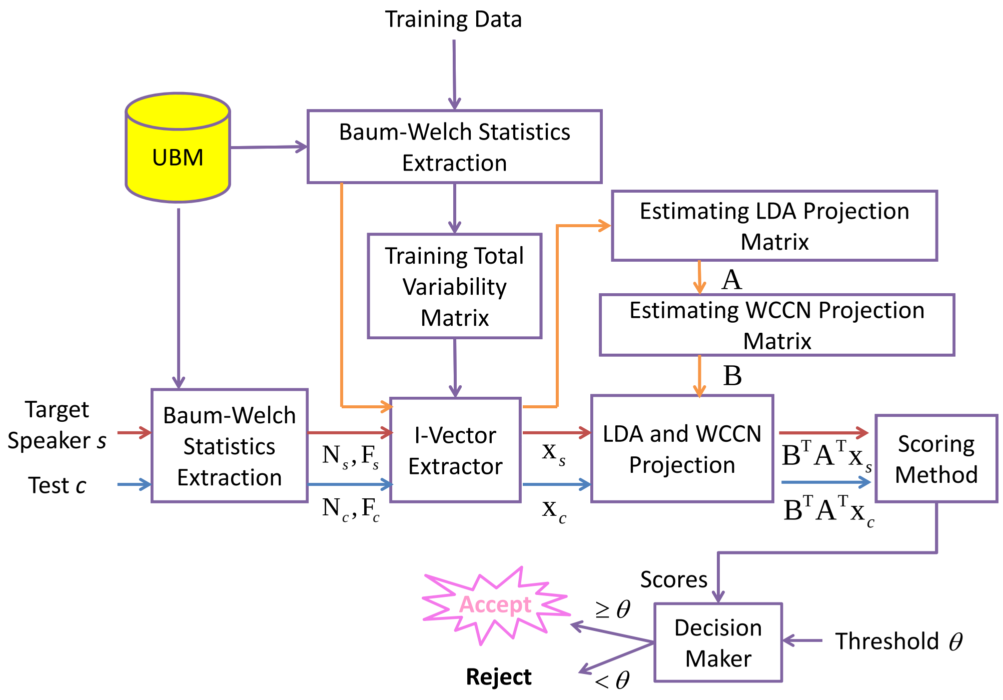


Figure 4.1: The flow chart of i-vector based speaker verification system. $\mathbf{N}_s$ and $\mathbf{N}_c$ are the zero-order Baum-Welch statistics for target speaker s and test c, respectively. $\mathbf{F}_s$ and $\mathbf{F}_c$ are the first-order Baum-Welch statistics for target speaker s and test c, respectively. $\mathbf{x}_s$ and $\mathbf{x}_c$ are i-vectors for target speaker s and test c, respectively. $\mathbf{A}$ is the LDA projection matrix and $\mathbf{B}$ is the WCCN projection matrix. $\mathbf{B}^\mathsf{T}\mathbf{A}^\mathsf{T}\mathbf{x}_s$ and $\mathbf{B}^\mathsf{T}\mathbf{A}^\mathsf{T}\mathbf{x}_c$ are the LDA and WCCN projected i-vectors for target speaker s and test c, respectively.

finding motivates them to model the total variability space (including channels and speakers) instead of modeling the channel- and speaker-spaces separately. Given an utterance of speaker s, the speaker- and channel-dependent GMM-supervector [1] $\boldsymbol{m}_s$ is written as:

$$\boldsymbol{m}_s = \boldsymbol{m} + \mathbf{T}\mathbf{x}_s \tag{4.1}$$

where $\boldsymbol{m}$ is the GMM-supervector of the universal background model (UBM) [20] which is speaker- and channel-independent, $\mathbf{T}$ is a low-rank total variability matrix, and $\mathbf{x}_s$ is a low-dimension vector called the i-vector.

Given an utterance with D-dimensional acoustic vector sequence $\mathbf{O} = \{\mathbf{o}_1, \ldots, \mathbf{o}_T\}$ belonging to speaker s and an UBM $\mathbf{\Lambda}^{(b)} = \{w_i^{(b)}, \boldsymbol{\mu}_i^{(b)}, \mathbf{\Sigma}_i^{(b)}\}_{i=1}^{M}$ with M mixture components, the zero-order and centered first-order Baum-Welch statistics are computed as follows [11, 53, 54]:

$$N_{s,i} = \sum_{t=1}^{T} \Pr(i|\mathbf{o}_t) \quad \text{and} \quad \tilde{\mathbf{F}}_{s,i} = \sum_{t=1}^{T} \Pr(i|\mathbf{o}_t)(\mathbf{o}_t - \boldsymbol{\mu}_i^{(b)}) \tag{4.2}$$

where

$$\Pr(i|\mathbf{o}_t) = \frac{w_i \mathcal{N}(\mathbf{o}_t; \boldsymbol{\mu}_i^{(b)}, \mathbf{\Sigma}_i^{(b)})}{\sum_{j=1}^{M} w_j \mathcal{N}(\mathbf{o}_t; \boldsymbol{\mu}_j^{(b)}, \mathbf{\Sigma}_j^{(b)})}, \quad i = 1, \ldots, M.$$

is the posterior probability of mixture components i given $\mathbf{o}_t$. An EM algorithm is used to estimate the total variability matrix [53]. In the E-step, the posterior covariance and posterior mean associated with an i-vector are given by:

$$\text{Cov}(\mathbf{x}_s, \mathbf{x}_s) = \mathbf{L}_s^{-1} \tag{4.3}$$

$$\mathbf{x}_s = \mathbf{L}_s^{-1} \mathbf{T}^{\mathsf{T}} {\mathbf{\Sigma}^{(b)}}^{-1} \tilde{\mathbf{F}}_s \tag{4.4}$$

where

$$\mathbf{L}_s = \mathbf{I} + \mathbf{T}^{\mathsf{T}} {\mathbf{\Sigma}^{(b)}}^{-1} \mathbf{N}_s \mathbf{T} \tag{4.5}$$

is a precision matrix and $\mathbf{I}$ is the identity matrix. $\mathbf{N}_s$ is an $MD \times MD$ diagonal matrix whose diagonal blocks are $N_{s,i}\mathbf{I}$. $\tilde{\mathbf{F}}_s$ is an $MD \times 1$ supervector formed by concatenating all centered first-order Baum-Welch statistics $\tilde{\mathbf{F}}_{s,i}$. $\mathbf{\Sigma}^{(b)}$ is a covariance matrix modeling the residual variability not captured by the $MD \times R$ total variability matrix $\mathbf{T}$. In practice, we substitute this matrix by the covariance matrices of the UBM, i.e., $\mathbf{\Sigma}^{(b)} = \mathrm{diag}\{\mathbf{\Sigma}_1^{(b)}, \dots, \mathbf{\Sigma}_M^{(b)}\}$. The posterior mean (Eq. 4.4) is the i-vector.

In the M-step, the total variability matrix $\mathbf{T}$ can be obtained by maximizing the auxiliary function [55]:

$$\begin{aligned} Q(\mathbf{T}) &= -\frac{1}{2}\sum_{s,i,t} \Pr(i|\mathbf{o}_t)\left[\log|\mathbf{L}_s| + (\mathbf{o}_t - \boldsymbol{m}_{s,i})^\mathsf{T}\,{\mathbf{\Sigma}_i^{(b)}}^{-1}(\mathbf{o}_t - \boldsymbol{m}_{s,i})\right] \\ &= -\frac{1}{2}\sum_{s,i}\left[N_{s,i}\log|\mathbf{L}_s| + N_{s,i}\mathrm{Tr}\left\{{\mathbf{\Sigma}_i^{(b)}}^{-1}\mathbf{T}_i\mathbf{x}_s{\mathbf{x}_s}^\mathsf{T}{\mathbf{T}_i}^\mathsf{T}\right\} - 2\mathrm{Tr}\left\{{\mathbf{\Sigma}_i^{(b)}}^{-1}\mathbf{T}_i\mathbf{x}_s\tilde{\mathbf{F}}_{s,i}^\mathsf{T}\right\}\right] + \mathcal{C} \end{aligned} \tag{4.6}$$

where $\boldsymbol{m}_{s,i}$ is the mean vector of i-th Gaussian component of speaker model and $\mathbf{T}_i$ is a $D \times R$ submatrix of $\mathbf{T}$. We maximize the objective function in Eq. 4.6 by taking its derivative with respect to $\mathbf{T}_i$, which results in following equations:

$$\mathbf{C}_i = \sum_s \tilde{\mathbf{F}}_{s,i}{\mathbf{x}_s}^\mathsf{T}, \tag{4.7}$$

$$\mathbf{A}_i = \sum_s N_{s,i}\left(\mathbf{L}_s^{-1} + \mathbf{x}_s{\mathbf{x}_s}^\mathsf{T}\right), \tag{4.8}$$

$$\mathbf{T}_i = \mathbf{C}_i\mathbf{A}_i^{-1} \qquad i = 1, \dots, M \tag{4.9}$$

In summary, the total variability matrix can be obtained by iteratively performing the E-step (Eq. 4.4 and Eq. 4.5) and M-step (Eq. 4.7 – Eq. 4.9).

4.2 Channel Compensation

Because i-vectors contain both speaker and channel variability in the total variability space, inter-session compensation plays an important role in the i-vector framework.

It was found in [11] that projecting the i-vectors by linear discriminant analysis followed by within class covariance normalization achieves the best performance. Another new channel compensation methods proposed in this thesis is multi-way LDA [22].

4.2.1 Linear Discriminant Analysis

Linear Discriminant Analysis (LDA) is commonly used for classification and dimensionality reduction. The idea is to find a set of orthogonal axes for minimizing the within-class variation and maximizing the between-class separation. In the i-vector framework, the i-vectors of a speaker constitute a class, leading to the following objective function for multi-class LDA [16]:

$$\mathbf{A} = \underset{\mathbf{A}}{\operatorname{argmax}} \left\{ \operatorname{tr} \left[\left(\mathbf{A}^{\mathsf{T}} \mathbf{S}_w \mathbf{A} \right)^{-1} \left(\mathbf{A}^{\mathsf{T}} \mathbf{S}_b \mathbf{A} \right) \right] \right\} \tag{4.10}$$

where $\mathbf{A}$ comprises the optimal subspace to which the i-vectors should be projected, $\mathbf{S}_w$ is the within-speaker scatter matrix, and $\mathbf{S}_b$ is the between-class scatter matrix. Eq. 4.10 leads to the projection matrix $\mathbf{A}$ that comprises the leading eigenvectors of $\mathbf{S}_w^{-1}\mathbf{S}_b$. Given a set of i-vectors $\mathcal{X} = \{\mathbf{x}_{ij}; i = 1, \ldots, N; j = 1, \ldots, H_i\}$ from N training speakers, $\mathbf{S}_w$ and $\mathbf{S}_b$ can be computed as follows:

$$\mathbf{S}_w = \sum_{i=1}^{N} \frac{1}{H_i} \sum_{j=1}^{H_i} (\mathbf{x}_{i,j} - \boldsymbol{\mu}_i)(\mathbf{x}_{i,j} - \boldsymbol{\mu}_i)^{\mathsf{T}} \tag{4.11}$$

and

$$\mathbf{S}_b = \sum_{i=1}^{N} (\boldsymbol{\mu}_i - \boldsymbol{\mu})(\boldsymbol{\mu}_i - \boldsymbol{\mu})^{\mathsf{T}} \tag{4.12}$$

where

$$\boldsymbol{\mu}_i = \frac{1}{H_i}\sum_{j=1}^{H_i}\mathbf{x}_{i,j} \tag{4.13}$$

is the mean i-vector of the i-th speaker, H_i is the number of utterances from the i-th training speaker, and $\boldsymbol{\mu}$ is the global mean of all i-vectors in the training set.

4.2.2 Within-Class Covariance Normalization

Within-class covariance normalization (WCCN) [17] was originally used for normalizing the kernels in SVMs. In the i-vector framework, WCCN is to normalize the within-speaker variation. Dehak et al. [11] found that the best approach is to project the LDA reduced i-vectors to a subspace specified by the square-root of the inverse of the following within-class covariance matrix:

$$\mathbf{W} = \sum_{i=1}^{N}\frac{1}{H_i}\sum_{j=1}^{H_i}(\mathbf{A}^\mathsf{T}\mathbf{x}_{i,j} - \widetilde{\boldsymbol{\mu}}_i)(\mathbf{A}^\mathsf{T}\mathbf{x}_{i,j} - \widetilde{\boldsymbol{\mu}}_i)^\mathsf{T} \tag{4.14}$$

where

$$\widetilde{\boldsymbol{\mu}}_i = \frac{1}{H_i}\sum_{j=1}^{H_i}\mathbf{A}^\mathsf{T}\mathbf{x}_{i,j} \tag{4.15}$$

and $\mathbf{A}$ is the LDA projection matrix found in Eq. 4.10. The WCCN projection matrix $\mathbf{B}$ can be obtained by Cholesky decomposition of $\mathbf{W}^{-1} = \mathbf{B}\mathbf{B}^\mathsf{T}$.

4.2.3 Multi-way Linear Discriminant Analysis

Conventional LDA uses the information of speaker labels and a variety of microphone recordings per speaker to obtain the within-speaker and between-speaker scatter matrices. As a result, the method performs covariance analysis on the speaker domain only, ignoring the fact that the training speakers typically use the same set

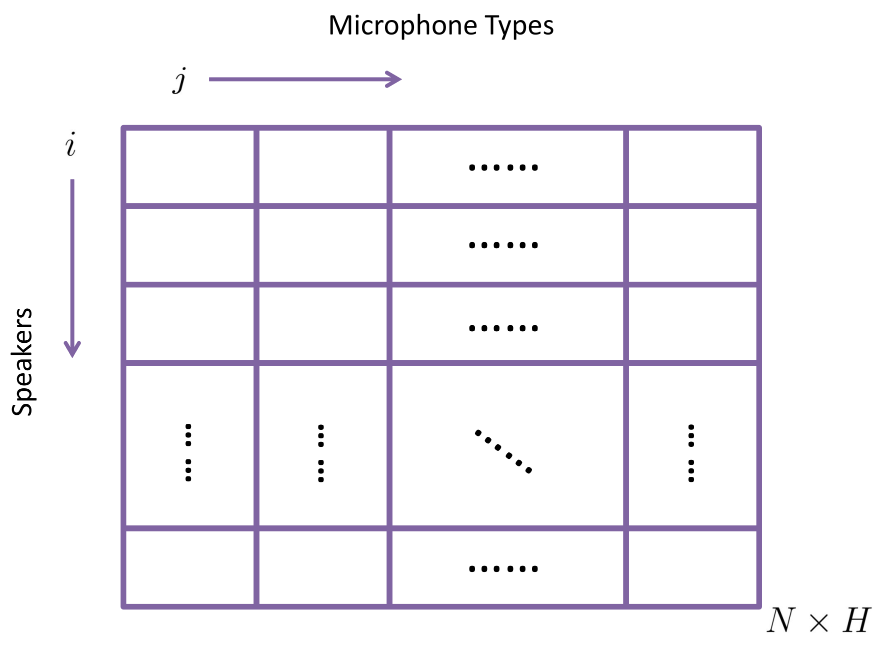


Figure 4.2: The grid for arranging the i-vectors of training speakers.

of microphones for recording. Here, we propose exploiting this extra information to strengthen the discriminative capability of LDA and refer to this method as multi-way LDA [22].

More precisely, the i-vectors of the training speakers are arranged in a grid, where the rows represent the speakers, the columns represents the microphones, and each element in the grid represents an i-vector (see Figure 4.2 for the grid). The dimension of i-vectors is firstly reduced by projecting the i-vectors to a subspace that maximizes the within-microphone variation, which represents the dispersion of i-vectors along the columns of the grid. The objective function is:

$$\mathbf{C} = \underset{\mathbf{C}:\|\mathbf{c}_i\|=1}{\operatorname{argmax}} \left[\operatorname{tr}\left(\mathbf{C}^{\mathsf{T}}\mathbf{S}_{wm}\mathbf{C}\right)\right] \qquad i = 1, \ldots, L \tag{4.16}$$

where $\mathbf{C} = [\mathbf{c}_1 \ \mathbf{c}_2 \ \cdots \ \mathbf{c}_L]$ defines the optimal discriminant subspace of dimension L on which the i-vectors should be projected and $\mathbf{S}_{wm}$ is the within-microphone scatter matrix:

$$\mathbf{S}_{wm} = \frac{1}{H} \sum_{i=1}^{N} \sum_{j=1}^{H} (\mathbf{x}_{i,j} - \boldsymbol{\mu}_j)(\mathbf{x}_{i,j} - \boldsymbol{\mu}_j)^{\mathsf{T}} \tag{4.17}$$

where $\boldsymbol{\mu}_j = \frac{1}{N} \sum_{i=1}^{N} \mathbf{x}_{i,j}$ is the mean i-vector of the j-th microphone, N is the number of training speakers, and H is the number of microphones. Maximizing Eq 4.16 leads to the projection matrix $\mathbf{C}$ that comprises the L leading eigenvectors of $\mathbf{S}_{wm}$. Then, conventional LDA can be applied to the dimension reduced i-vectors, which amounts to finding a subspace that maximize the speaker separability but minimize the within speaker variability (along the rows of the grid).

Note that unlike principal component analysis (PCA) [16, 56] where the labels of training data are ignored, Eqs. 4.16 and 4.17 make use of both speaker and microphone labels of the training data. The use of microphone labels is expected to find a more discriminative subspace than the one found by PCA. Because for each column in the grid, the i-vectors are produced by different speakers using the same microphone, more discriminative subspace can be found by maximizing the separability of different speakers using the same microphone. It is however not desirable to minimize the between-microphone variability because the rank of between-microphone scatter matrix $\mathbf{S}_{bm}$ is typically very small. For example, in our experiments, the maximum value of H is 8, meaning that the rank of $\mathbf{S}_{bm}$ is only 7.

4.3 Scoring Methods

The most common scoring methods in i-vector speaker verification are cosine distance scoring, SVM scoring and PLDA scoring.

4.3.1 Cosine Distance Scoring

Cosine distance scoring (CDS) [57] is computationally efficient. The method computes the cosine distance between the claimant's i-vector ($\mathbf{x}_t$) and target-speaker's i-vector ($\mathbf{x}_s$) in the LDA+WCCN projection space:

$$S_{\cos}\left(\mathbf{x}_t, \mathbf{x}_s\right) = \frac{\left\langle \mathbf{B}^{\mathsf{T}}\mathbf{A}^{\mathsf{T}}\mathbf{x}_t, \mathbf{B}^{\mathsf{T}}\mathbf{A}^{\mathsf{T}}\mathbf{x}_s \right\rangle}{\left\|\mathbf{B}^{\mathsf{T}}\mathbf{A}^{\mathsf{T}}\mathbf{x}_t\right\| \left\|\mathbf{B}^{\mathsf{T}}\mathbf{A}^{\mathsf{T}}\mathbf{x}_s\right\|} \tag{4.18}$$

The score is then further normalized (typically by ZT-norm) before comparing with a threshold for making a decision.

4.3.2 SVM Scoring

Another popular scoring method in i-vector system is SVM scoring [57], which has been already introduced in Section 3.2. Unlike cosine distance scoring, the advantage of SVM scoring is that the contribution of individual background speakers and the target speaker to the verification scores can be optimally weighted by the Lagrange multipliers of the target-speaker's SVM. Given the SVM of target speaker s, the verification score of test claimant t is given by

$$S_{\text{SVM}}(\mathbf{x}_t, \mathbf{x}_s) = \alpha_0^{(s)} K\left(\mathbf{x}_t, \mathbf{x}_s\right) - \sum_{i \in \mathcal{S}^{(b)}} \alpha_i^{(s)} K\left(\mathbf{x}_t, \mathbf{x}^{(b_i)}\right) + d^{(s)} \tag{4.19}$$

where $\alpha_0^{(s)}$ is the Lagrange multiplier corresponding to the target speaker,[1] $\alpha_i^{(s)}$'s are Lagrange multipliers corresponding to the background speakers, $\mathcal{S}^{(b)}$ is a set containing the indexes of the support vectors in the background-speaker set, and $\mathbf{x}^{(b_i)}$ is the i-vectors of the i-th background speaker. It was found [11] that the

[1]We assume one enrollment utterance per target speaker.

cosine kernel is the most appropriate kernel for i-vector system. Specifically,

$$K(\mathbf{x}_t, \mathbf{x}_s) = \frac{\left\langle \mathbf{B}^\mathsf{T}\mathbf{A}^\mathsf{T}\mathbf{x}_t, \mathbf{B}^\mathsf{T}\mathbf{A}^\mathsf{T}\mathbf{x}_s \right\rangle}{\left\|\mathbf{B}^\mathsf{T}\mathbf{A}^\mathsf{T}\mathbf{x}_t\right\| \left\|\mathbf{B}^\mathsf{T}\mathbf{A}^\mathsf{T}\mathbf{x}_s\right\|} \tag{4.20}$$

where we replace $\mathbf{x}_s$ by $\mathbf{x}^{(b_i)}$ for evaluating the second term of Eq. 4.19. Note that Eq. 4.18 and Eq. 4.20 are the same. However, their role in the scoring process is different. The former is directly used for calculating the score, whereas the latter is used for kernel evaluation.

Similar to GMM–SVM scoring, the SVMs in the i-vector SVM scoring also suffers from the data imbalance problem [58, 59]. The problem, however, can be alleviated by the proposed UP-AVR. See Section 5.4 for the details of UP-AVR.

4.3.3 PLDA Scoring

In i-vector/PLDA speaker verification systems [12–14], the verification decision is based on the likelihood ratio score derived from two hypotheses: (1) the test i-vector and the target-speaker i-vector are from the same speaker and (2) these two i-vectors are from two different speakers. Specifically, given a length-normalized [13] test i-vector $\mathbf{x}_t$ and a length-normalized target-speaker's i-vector $\mathbf{x}_s$, the verification score is given by:

$$\begin{aligned} S_{\mathrm{LR}}(\mathbf{x}_s, \mathbf{x}_t) &= \frac{P(\mathbf{x}_s, \mathbf{x}_t | \text{same speaker})}{P(\mathbf{x}_s, \mathbf{x}_t | \text{different speakers})} \\ &= \text{const} + \mathbf{x}_s^\mathsf{T}\mathbf{Q}\mathbf{x}_s + \mathbf{x}_t^\mathsf{T}\mathbf{Q}\mathbf{x}_t + 2\mathbf{x}_s^\mathsf{T}\mathbf{P}\mathbf{x}_t \end{aligned} \tag{4.21}$$

where

$$\begin{aligned}\mathbf{P} &= \boldsymbol{\Lambda}^{-1}\boldsymbol{\Gamma}(\boldsymbol{\Lambda} - \boldsymbol{\Gamma}\boldsymbol{\Lambda}^{-1}\boldsymbol{\Gamma})^{-1}; \quad \boldsymbol{\Lambda} = \mathbf{V}\mathbf{V}^{\mathsf{T}} + \boldsymbol{\Sigma} \\ \mathbf{Q} &= \boldsymbol{\Lambda}^{-1} - (\boldsymbol{\Lambda} - \boldsymbol{\Gamma}\boldsymbol{\Lambda}^{-1}\boldsymbol{\Gamma})^{-1}; \quad \boldsymbol{\Gamma} = \mathbf{V}\mathbf{V}^{\mathsf{T}}.\end{aligned} \tag{4.22}$$

and $\mathbf{V}$ is the factor loading matrix and $\boldsymbol{\Sigma}$ is the covariance of the PLDA model. We call this method PLDA scoring in this thesis. In Section 6.1, the theory of PLDA will be described.

4.4 *Small Sample-Size Problem in LDA and WCCN*

In i-vector based speaker verification, both LDA and WCCN involve the computation of a within-class covariance matrix that requires many speakers with multiple sessions per speaker. For best performance and numerical stability, one should opt for a large number of sessions per speaker. But in practice, it is costly and inconvenient to collect such a corpus. In a typical training dataset, the number of speakers could be fairly large, but the number of speakers who can provide many sessions is quite limited. The lack of multiple sessions per speaker could lead to singular within-speaker scatter matrix. This problematic issue is known as *small sample-size problem* [18, 19]. Chapter 5 will introduce the traditional ways to deal with this problem and explain the new method proposed by this thesis for solving this small sample-size problem.

Chapter 5

UTTERANCE PARTITIONING WITH ACOUSTIC VECTOR RESAMPLING

There are some fundamental issues in the GMM–SVM and i-vector based systems. For example, in GMM–SVM systems, there is a severe imbalance between the amount of speaker-class and impostor-class training data (cf. Section 3.5). This issue must be addressed; otherwise the decision boundaries of the SVMs will be largely dependent on the impostor-class data. To estimate the LDA projection matrix in i-vector based systems, we need to have a population of training speakers with each speaker providing a certain number of training utterances. However, in practical situations, it is difficult to obtain many utterances per speaker. The lack of training utterances per speaker will lead to the so-called small sample-size problem when estimating the LDA projection matrix (cf. Section 4.4).

Although data imbalance problem and small sample-size problem are caused by different reasons, both of them can be solved by the method described in this chapter. We refer to the method as utterance partitioning with acoustic vector resampling (UP-AVR) [15]. Results shown in Chapter 8 demonstrate that this method is effective in overcoming the data imbalance problem and small sample-size problem.

This chapter highlights some conventional techniques for solving the data imbalance problem and small sample-size problem. Particular attention will be paid on how these techniques help the SVM training algorithm to find better decision boundaries when there is a severe imbalance between the two classes. Then, the chapter explains how the proposed approach – UP-AVR – alleviates the data imbalance problem in GMM–SVM. Finally, the chapter explains how UP-AVR also helps to solve the small sample-size problem in i-vector systems.

5.1 Approaches to Alleviate Data Imbalance

Because data imbalance occurs in many problem domains, a number of strategies have been proposed to alleviate the effect of imbalanced data on SVM classifiers. These strategies can be divided into two categories: data processing approaches and algorithmic approaches.

5.1.1 Data Processing Approaches

Data processing approaches attempt to re-balance the training data without changing the SVM training algorithm. This category can be further divided into over-sampling, under-sampling, and combination of over- and under-sampling. A comparison of these methods can be found in [60].

In over-sampling, additional minority-class training examples are generated from existing data. Chawla et al. [61,62] proposed the Synthetic Minority Over-sampling Technique (SMOTE) to create minority-class training instances. This method firstly identifies k nearest neighbors for each minority-class training sample. The original minority-class sample and one of its nearest neighbors form a pair which corresponds to two points in the vector space. Then, a new minority-class sample is created by selecting any random point along the line linking these two points. SMOTE has been extended to borderline-SMOTE [63] in which SMOTE is applied to the minority-class samples that are close to the border of the minority-class decision region.

Nickerson et al. [64] proposed a cluster-based oversampling method that considers not only the between-class imbalance but also within-class imbalance. Specifically, unsupervised clustering is used to determine the within-class imbalance (relative densities of sub-components within a class). This information is then used to guide the resampling of the minority-class to alleviate the between-class imbalance. This method overcomes the data imbalance problem by identifying rare cases and resampling them individually, thus avoiding the creation of small disjuncts [65]. It was found that addressing the data-imbalance problem and the small-disjunct problem simultaneously is better than solving the data imbalance problem alone [66].

In under-sampling [67, 68], a subset of majority-class training samples are selected for training. One typical under-sampling method is random sampling (or undirected sampling) which randomly selects a subset of training samples from the majority class. Many studies have shown that random sampling reduces classification accuracy [69]. Another under-sampling technique is directed sampling. This sampling strategy selects the majority-class samples that are close to the decision hyperplane [70]. McLaren et al. [71] proposed a sample selection method for the situation where a large number of binary classifiers can share the same set of majority-class samples. In the method, a large number of SVMs are trained such that individual SVMs use their own minority-class samples but share the same pool of T majority-class samples. Then, the top N ($N < T$) majority-class samples, which are often selected by the SVM training algorithm as support vectors, are selected as the majority-class samples for training all binary classifiers. The method not only alleviates the data imbalanced problem but also reduces the number of redundant samples.

Methods that combine over- and under-sampling have also been proposed. For example, Liu et al. [72] proposed to integrate these two sampling methods and used an ensemble of SVMs to boost the performance.

While studies have shown that the data processing approaches can improve the performance of SVMs in some situations, they do have their own problems. For example, over-sampling will increase the number of support vectors, causing computational burden for large datasets [73]. Some over-sampling techniques (e.g. SMOTE [61]) assume that the samples on the line joining two neighboring minority-class samples are also of the same class. This assumption may be invalid in some situations. Although under-sampling can help move the decision boundary towards the majority class, it causes information loss if useful samples are discarded [62]. A recent study [74] also suggests that for some applications, the performance of SVMs with over- or under-sampling could be poorer than those without any sampling.

5.1.2 Algorithmic Approaches

The algorithmic approaches attempt to modify the SVM training algorithm to mitigate the effect caused by data imbalance. One earlier attempt is to assign different misclassification cost to positive and negative training samples [75, 76]. However, studies [49] have shown that this approach is not very effective, because increasing the value of the Lagrange multipliers of the minority class (due to the increase in the penalty factor) will also increase some of the Lagrange multipliers in the majority class to satisfy the constraint $\sum_i \alpha_i y_i = 0$, where α_i's are the Lagrange multipliers and y_i's are the class labels ($+1$ or -1). Another algorithmic approach is to modify the kernel according to the distribution of training data [49]. This approach, however, requires longer classification time than the standard SVM.

Recently, several new methods have been reported in the literature with good classification performance on imbalanced data. Hong et al. [77] proposed a kernel classifier construction algorithm based on orthogonal forward selection (OFS), which aims at maximizing the area under the ROCs. Fakeri-Tabrizi et al. [78] proposed a ranking SVM to optimize the area under the ROC curve. While these two methods are efficient, they are not appropriate for speaker recognition, because it is difficult to formulate the area under the ROC in speaker recognition evaluation. Hwang et al. [79] proposed to introduce weight parameters to the training algorithm of the Lagrangian SVM (LSVM) to deal with data imbalance; however, this method requires longer training time than the standard SVM.

5.2 Approaches to Solve Small Sample-Size Problem

There are many applications in which the dimensionality of data is larger than the number of training samples. For examples, in microarray data analysis [80], the number of genes tends to be much larger than the number of samples. In face recognition [81], the feature dimension is usually very high because it is proportional to the number of pixels. In machine learning literature, this is known as the *small sample-size problem*.

To apply LDA for classification or dimension reduction, the small sample-size problem will become an issue when the number of training samples is small but the feature dimension is high. This is because the LDA solution requires the computation of the inverse of the within-class scatter matrix, which may become singular when the number of training samples is small. Over the years, a number of methods have been proposed to address this problem.

A simple approach is to use PCA to reduce the dimension of the original vector before applying LDA to find the optimal discriminant subspace [82,83]. The method is known as PCA+LDA in the literature. The dimension of the PCA projection space is selected such that the within-class scatter matrix becomes nonsingular so that LDA can be applied without numerical difficulty. The singularity problem has also been overcome by replacing the matrix inverse by pseudo-inverse [84, 85] or by adding a constant to the diagonal elements of the scatter matrix [86]. The former is called pseudo-inverse LDA and the latter is known as regularized LDA. The advantage of pseudo-inverse LDA is that components with eigenvalues smaller than a threshold are automatically discarded, thus avoiding the singularity problem. While it can be shown that the regularized scatter matrix is always nonsingular, the regularized LDA is harder to use because the amount of diagonal offset needs to be determined by cross validation. A more general form of regularized LDA is the penalized LDA [87]. Instead of adding a positive diagonal matrix to the scatter matrix, penalized LDA adds a symmetric and positive semi-definite matrix to the scatter matrix in order to produce spatially smooth LDA coefficients.

More recently, null-space LDA [19, 88] and orthogonal LDA [18, 89] have been proposed to address the small sample-size problem. In null-space LDA, the between-class distance is maximized in the null space of the within-class scatter matrix. The singularity problem is implicitly avoided because no matrix inverse is needed. The method reduces to the conventional LDA when the within-class scatter matrix has full rank. In orthogonal LDA, the discriminant vectors are orthogonal to each other, and the optimal transformation matrix is obtained by simultaneous diagonalization of the between-class, within-class, and total scatter matrices. Again, the singularity problem has been avoided because matrix inverse is only applied to the diagonal matrix containing non-zero singular values [18]. It has been shown that when the rank of the total scatter matrix is equal to the sum of the rank of the between-class and within-class scatter matrices, orthogonal LDA is equivalent to null-space LDA [89].

5.3 UP-AVR for GMM–SVM Speaker Verification

5.3.1 Motivation of UP-AVR

In GMM–SVM speaker verification, the number of minority-class samples (enrollment utterances) is extremely small. In fact, it is not uncommon to have only one enrollment utterance per client speaker. This extreme data imbalance excludes the use of over-sampling methods such as SMOTE where minority-class samples are generated based on the existence of some (but not one) minority-class samples. Under-sampling is also not an option, because of the information loss that arises from discarding important background speakers. The problem of this extremely data imbalance is that the SVM's decision boundary is largely governed by the impostor-class supervectors (support vectors).

To increase the influence of speaker-class data on the decision boundary, one may use more enrollment utterances, which means more supervectors from the speaker class. However, as mentioned earlier, it is not practical to request users to provide multiple enrollment utterances. To solve this problem without introducing extra burden on users, this thesis proposes partitioning an enrollment utterance into a number of sub-utterances. The method is referred to as *utterance partitioning* (UP).

Given an enrollment utterance, a large number of partitions will produce many sub-utterances, but their length may be too short to represent the speaker. On the other hand, if the number of partitions is too small, the benefit of utterance partitioning diminishes. Obviously, there is a trade-off between the length of the sub-utterances and the representation capability of the resulting GMM supervectors.

5.3.2 Procedure of UP-AVR

One possible way to generate more sub-utterances with reasonable length is to use the notion of random resampling in bootstrapping [90]. The idea is based on the fact that the MAP adaptation algorithm [20] uses the statistics of the whole utterance to update the GMM parameters (see Eq. 2.7). In other words, changing the order of acoustic vectors will not affect the resulting MAP-adapted model. Therefore, we may randomly rearrange the acoustic vectors in an utterance and then partition the utterance into N sub-utterances and repeat the process as many times as appropriate. More precisely, if this process is repeated R times, we obtain RN sub-utterances from a single enrollment utterance. We refer to this approach as utterance partitioning with acoustic vector resampling (UP-AVR). Its procedure is as follows:

Step 1: For each utterance from the background speakers, compute its acoustic vectors (MFCCs and their derivatives). Then, remove the acoustic vectors corresponding to the silence frames and divide the resulting vectors into N partitions (sub-utterances).

Step 2: For each background speaker, use his/her N sub-utterances and full-length utterance to create $N{+}1$ background GMM-supervectors. For B background speakers, this procedure results in $B(N+1)$ background supervectors.

Step 3: Given an enrollment utterance of a target speaker, compute its acoustic vectors and remove the vectors corresponding to the silence frames. Then, randomize the indexes of the resulting vectors in the sequence. Divide the randomized sequence of acoustic vectors into N partitions (sub-sequences).

Use the N sub-sequences to create N GMM-supervectors by adapting the UBM.

Step 4: Repeat Step 3 R times to obtain RN target speaker's supervectors; together with the full-length utterance, form $RN + 1$ speaker's supervectors.

Step 5: Use the $RN + 1$ supervectors created in Steps 3 and 4 as positive-class data and the $B(N + 1)$ background supervectors created in Step 2 as negative-class data to train a linear SVM for the corresponding target speaker.

Figure 5.1 illustrates the procedure of UP-AVR. The same partitioning strategy is applied to both target-speaker utterances and background utterances so that the length of target-speaker's sub-utterances matches that of the background speakers' sub-utterances. Matching the duration of target-speaker utterances with that of background utterances has been found useful in previous studies [91].

5.3.3 Advantages of UP-AVR

The advantages of the utterance partitioning approach are two-fold. First, it can increase the influence of positive-class data on the decision boundary. Second, when the original enrollment utterances are significantly longer than the verification utterances, utterance partitioning can create sub-utterances with length that matches the verification utterances. This can reduce the mismatches between the test supervectors and the enrollment supervectors, because the amount of MAP adaptation depends on the length of the adaptation utterances.

Figure 5.2 further demonstrates the benefit of increasing the number of speaker-class supervectors. In the figure, the larger the difference between the speaker-class scores and impostor-class scores (Eq. 3.1), the larger the discriminative power of

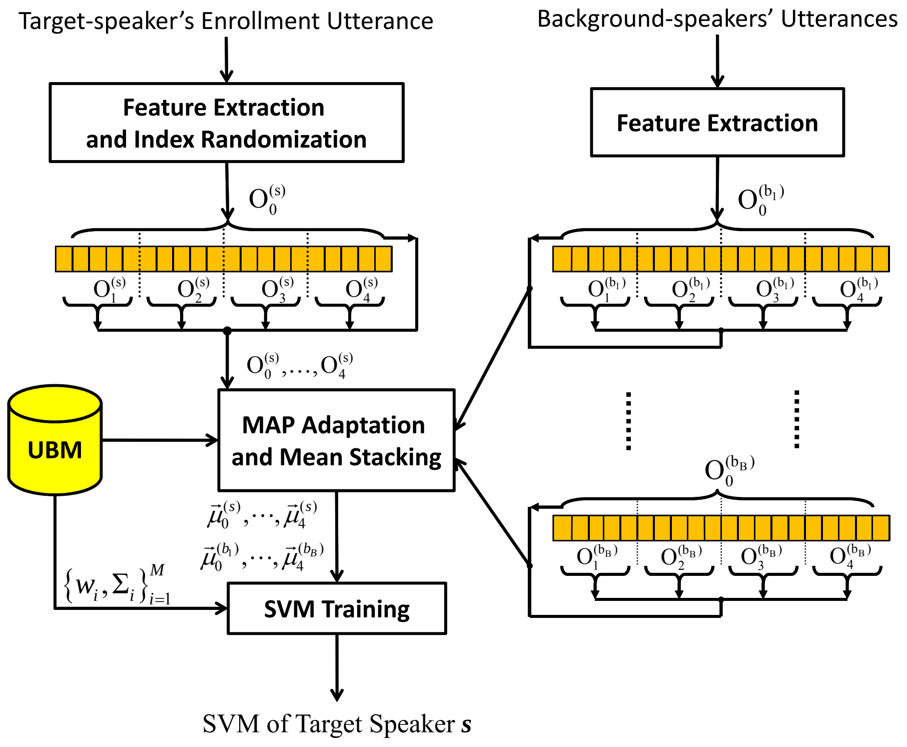

Target-speaker's Enrollment Utterance
Background-speakers' Utterances
Feature Extraction and Index Randomization
Feature Extraction
$O_0^{(s)}$
$O_0^{(b_1)}$
$O_1^{(s)}$ $O_2^{(s)}$ $O_3^{(s)}$ $O_4^{(s)}$
$O_1^{(b_1)}$ $O_2^{(b_1)}$ $O_3^{(b_1)}$ $O_4^{(b_1)}$
$O_0^{(s)},\dots,O_4^{(s)}$
UBM
MAP Adaptation and Mean Stacking
$O_0^{(b_B)}$
$O_1^{(b_B)}$ $O_2^{(b_B)}$ $O_3^{(b_B)}$ $O_4^{(b_B)}$
$\vec{\mu}_0^{(s)},\cdots,\vec{\mu}_4^{(s)}$
$\vec{\mu}_0^{(b_1)},\cdots,\vec{\mu}_4^{(b_B)}$
$\{w_i,\Sigma_i\}_{i=1}^{M}$
SVM Training
SVM of Target Speaker *s*

Figure 5.1: The procedure of utterance partitioning with acoustic vector resampling (UP-AVR). Note that randomization and creation of target-speaker's supervectors can be repeated several times to obtain a sufficient number of target-speaker's supervectors.

the SVM. As expected, the SVM trained with 5 target-speaker utterances [(A), green] exhibits the greatest discriminative power and the largest score difference (1.37). However, this strategy uses 5 times as much speech materials as using one target-speaker utterance [(B), blue-dashed] for training. Evidently, the discriminative power of the SVM trained with the speaker-class supervectors generated by UP-AVR [(D)–(F)] is greater than that trained with only one speaker-class supervector

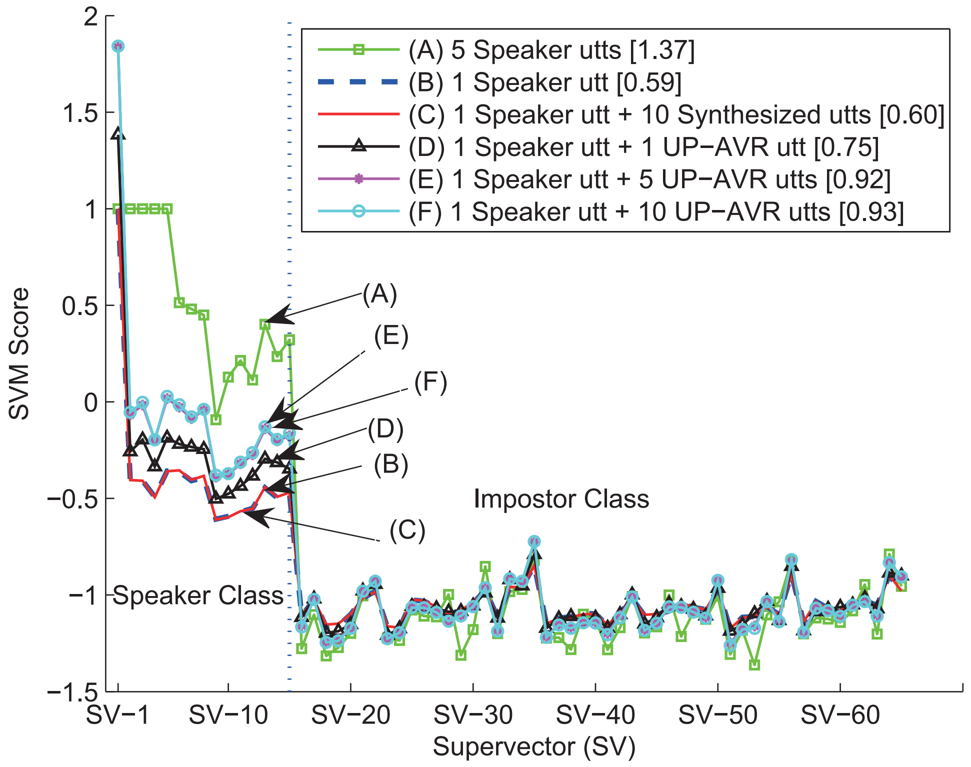


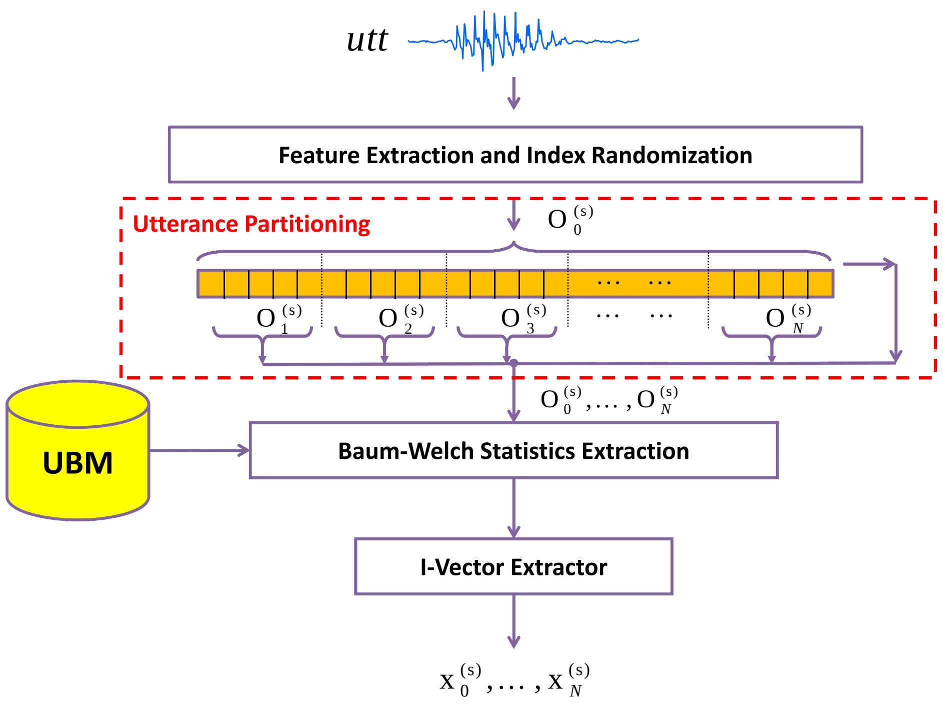
Figure 5.2: Scores produced by SVMs that use one or more speaker-class supervectors (SVs) and 250 background SVs for training. The horizontal axis represents the training/testing SVs. Specifically, SV-1 to SV-15 were obtained from 15 utterances of the same target-speaker, and SV-16 to SV-65 were obtained from the utterances of 50 impostors. For (B) to (F) in the legend, SV-1 was used for training and SV-2 to SV-15 were used as speaker-class test vectors, i.e., the score of SV-1 is the training-speaker score, whereas the scores of SV-2 to SV-15 are test speaker scores. For (A), SV-1 to SV-5 were used for training and SV-6 to SV-15 were used for testing. In (C), 10 speaker-class SVs were generated by a Gaussian generator using SV-1 as the mean vector and component-wise intra-speaker variances as the diagonal covariance matrix; whereas in (D)–(F), 1, 5, and 10 speaker-class UP-AVR SVs were obtained from the utterance corresponding to SV-1 by UP-AVR. $N = 5, R = 1$ is set for generating 5 speaker-class UP-AVR SVs and $N = 5, R = 2$ is set for generating 10 speaker-class UP-AVR SVs. Values inside the squared brackets are the mean difference between speaker scores and impostor scores. NAP [1] had been applied to the SVs.

(B). The overlapping between (B) and (C) suggests that synthesizing speaker-class supervectors in the neighborhood of the speaker-class supervector has little effect on the decision boundary, resulting in almost no change in the SVM scores. This agrees with the hypothetical situation shown in Figure 3.3 and Figure 3.4, where changing the speaker-class supervectors within a certain region will not change the decision plane. The overlapping between (E) and (F) suggests that just 5 speaker-class supervectors is sufficient, which agrees with the results in Table 8.8, Table 8.9, and Figure 8.3.

5.4 *UP-AVR for I-Vector Speaker Verification*

UP-AVR [15] was originally introduced to alleviate the the data imbalance problem in GMM–SVM [1] (Section 5.3). In the current work, we found that UP-AVR is also applicable to the i-vector framework. There are two reasons: (1) changing the order of acoustic vectors will not affect the resulting i-vector and (2) the amount of speaker-dependent information that an i-vector can capture will become saturated when the utterance length exceeds a certain threshold (cf. Section 5.4.1).

Figure 5.3 illustrates the procedure of UP-AVR for i-vector extraction. It is similar as the procedure of UP-AVR for GMM-supervector system. To increase the number of i-vectors for each utterance but maintaining their representation power, the order of the acoustic vectors in the sequence is randomly rearranged, then the sequence is partitioned into N equal-length segments, and an i-vector is estimated from each segment. If this partitioning-randomization process is repeated R times, $RN + 1$ i-vectors can be obtained from a single conversation, where the additional one is obtained from the entire acoustic sequence. In theory, we can obtain many i-vectors when R is large. However, when R increases, the segments will contain



utt
Feature Extraction and Index Randomization
Utterance Partitioning
$O_0^{(s)}$
$O_1^{(s)}$
$O_2^{(s)}$
$O_3^{(s)}$
$O_N^{(s)}$
$O_0^{(s)}, \ldots, O_N^{(s)}$
UBM
Baum-Welch Statistics Extraction
I-Vector Extractor
$x_0^{(s)}, \ldots, x_N^{(s)}$

Figure 5.3: The procedure of utterance partitioning with acoustic vector resampling (UP-AVR) for i-vector system. Note that index randomization, utterance partitioning, and i-vector extraction can be repeated several times to obtain a sufficient number of i-vectors.

many acoustic vectors that are identical to each other, resulting in many similar i-vectors. Similar situation occurs if two or more segments are used for estimating an i-vector. To avoid this situation, R should be small. In this work, R was limited to 4. Chapter 8 analyzes the effect of varying R and N on the discriminative power of i-vectors.

There are two advantages of UP-AVR in i-vector based speaker verification. First, it can improve the effectiveness of LDA and WCCN under limited speech resources.

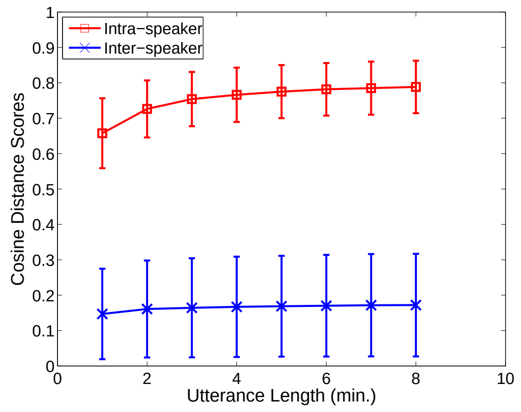
Second, it can be applied to alleviate the data imbalance problem in SVM scoring.

5.4.1 Effect of Utterance Length on I-Vectors

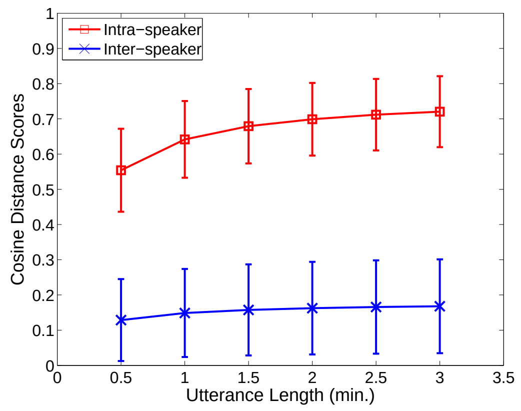
The major advantage of the i-vector framework is that a variable-length utterance can now be represented by a low-dimensional i-vector. This low-dimensional space facilitates the application of LDA and WCCN, which require low-dimensionality to ensure numerical stability (unless abundant training data are available). As the i-vectors are very compact, it is interesting to investigate if short utterances are still able to maintain the discriminative power of i-vectors. To this end, we computed the intra- and inter-speaker cosine-distance scores of 272 speakers extracted from the interview_mic and phonecall_tel sessions of NIST 2010 SRE.

Each conversation of these speakers was divided into a number of equal-length segments. Then, sub-utterances of variable length were obtained by concatenating variable numbers of equal-length segments. A voice activity detector (VAD) [92] was applied to extract the acoustic vectors corresponding to the speech regions in the sub-utterances. The acoustic vectors of each sub-utterance were then used for estimating an i-vector, followed by LDA and WCCN projections to 150-dim i-vectors, which were used for computing the cosine distance scores.

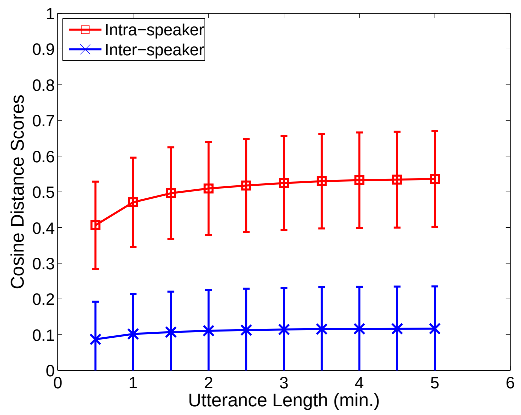
Figure 5.4 shows the mean intra- and inter-speaker scores (with error bars indicating two standard deviations) of the three types of speech. For "8-min interview_mic", the scores were obtained from the 8-min interview sessions of 29 male speakers in NIST 2010 SRE, each providing 4 interview conversations. This amounts to 174 intra-speaker scores and 6,496 inter-speaker scores for each utterance length. For "3-min interview_mic", the scores were obtained from the 3-min interview sessions of 196 male speakers, each providing 4 interview conversations. This amounts to 1,176



(a) 8-min interview_mic



(b) 3-min interview_mic



(c) 5-min phonecall_tel

Figure 5.4: Intra-speaker and inter-speaker cosine-distance scores versus utterance length for (a) 8-min interview conversation, (b) 3-min interview conversation, and (c) 5-min telephone conversation.

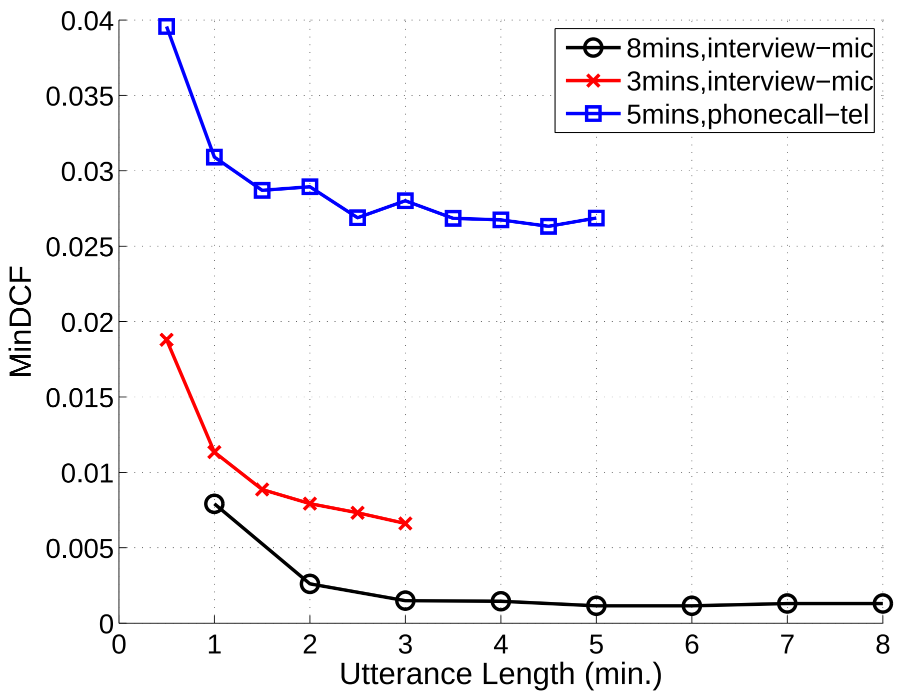


Figure 5.5: Minimum decision costs (old MinDCF) achieved by i-vectors derived from utterances of various lengths. The costs were based on the intra- and inter-speaker cosine-distances shown in Figure. 5.4. For computation efficiency, no score normalization was applied.

intra-speaker scores and 305,760 inter-speaker scores for each utterance length. For "5-min phonecall_tel", the scores were obtained from the 5-min phonecall conversations of 47 male speakers, each providing 4 conversations. This amounts to 282 intra-speaker scores and 17,296 inter-speaker scores for each utterance length. Evidently, both types of scores flatten out after the segment length used for estimating the i-vectors exceeds a certain threshold.

To further analyze the discriminative power of i-vectors with respect to the utterance length, we plot in Figure 5.5 the minimum decision cost (MinDCF) versus the utterance length for estimating the i-vectors using the intra- and inter-speaker scores shown in Figure 5.4. The lower the cost, the higher the discriminative power of the i-vectors. The result clearly suggests that the discriminative power becomes saturated for utterance length exceeding 2 minutes. This finding suggests that it is not necessary to record very long utterances for the i-vectors to achieve good performance. From another perspective, if long conversations are already available, it may be beneficial to divide the long conversations into a number of sub-utterances to produce more i-vectors per conversation.

5.4.2 Properties of I-Vectors Produced by UP-AVR

It is of interest to investigate the statistical properties of the generated i-vectors when the values of N and R vary. To this end, we selected one hundred 3-min interview utterances from NIST 2010 SREs and estimate the i-vectors produced by UP-AVR for different N and R, followed by LDA and WCCN projection to 150-dimensional vectors. Then, for each full-length utterance, we computed the cosine distance scores between the i-vector derived from the full-length utterance and the RN i-vectors generated by UP-AVR. We also computed the cosine distance scores among these generated i-vectors. The scores across all of the 100 full-length utterances are then averages, which results in two sets of average scores: one representing the similarity between full-length utterances and sub-utterances and another representing the similarity among sub-utterances.

Figure 5.6 shows how these scores vary with respect to the number of partitions (N) per conversation while keeping R fixed. The figure clearly suggests that when $N = 2$ (i.e., sub-utterances are long), the i-vectors of sub-utterances are similar to that of the full-length utterances. They are also similar among themselves. The similarity decreases but the score variances increase when the number of partitions

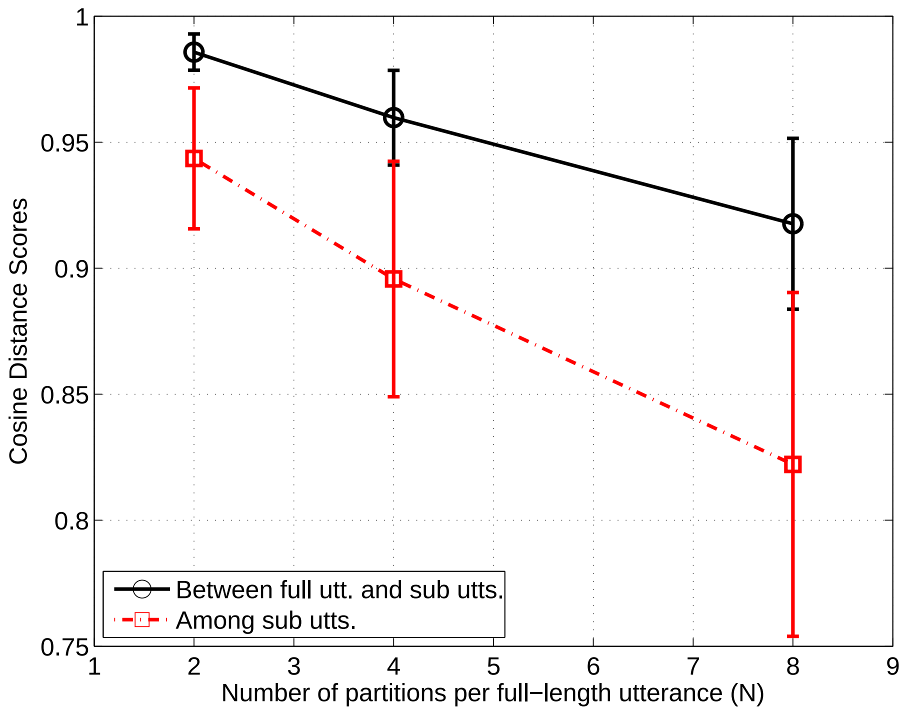


Figure 5.6: Average cosine distance scores with error bars versus the number of partitions per full-length utterance. In all cases, the number of re-sampling in UP-AVR was set to 1, i.e. $R = 1$.

increases, i.e., sub-utterances become shorter. This is reasonable because when N is small, the acoustic vectors of a sub-utterance are identical to a large portion of the acoustic vectors in the full-length utterance. When N increases, the sub-utterances become shorter, resulting in lower similarity but higher variability with respect to each others.

Figure 5.7 shows the effect of varying the number of resampling R on these two sets of scores when N is fixed. As opposed to Figure 5.6, when R increases, the similarity among the sub-utterances of a full-length utterance increases but their score variances decrease. The reason is that after several cycles of resampling, the sub-utterances in the current resampling cycle will contain some acoustic vectors

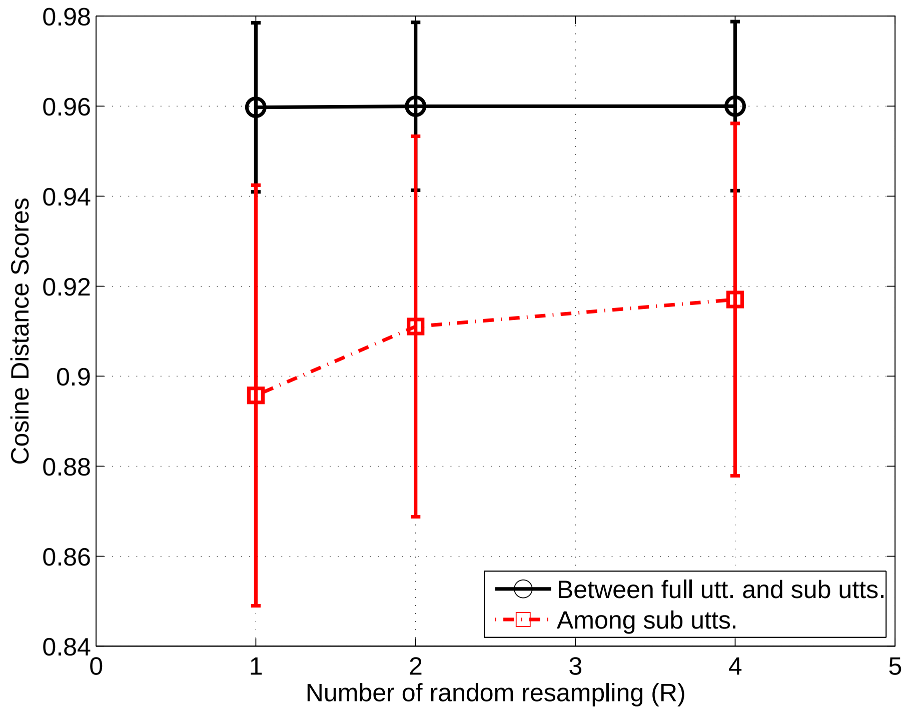


Figure 5.7: Average cosine distance scores with error bars versus the number of random resampling R. In all cases, the number of partitions in UP-AVR was set to 4, i.e. $N = 4$.

that have already appeared in the sub-utterances of the previous resampling cycles, causing higher similarity among the i-vectors of the sub-utterances. However, these sub-utterances are still not identical and they play important role in the training of target-speaker SVMs, which will be further elaborated in Chapter 8.

5.4.3 Alleviating Small Sample-Size Problem

The aim of LDA is to find a subspace in which the intra-speaker variation is minimal and the inter-speaker variation is maximal. It requires a sufficient number of recording sessions per training speaker for estimating the inter- and intra-speaker covariance matrices. However, collecting such recordings is costly and inconvenient.

As demonstrated in Section 5.4.1, when the utterance length for i-vector extraction is sufficiently long, further increasing the length will not increase the i-vectors' discriminative power significantly. Therefore, given a long utterance, some intrinsic speaker information will be wasted if the whole utterance is used for estimating the i-vector. To make a better use of the long utterance, we can apply UP-AVR to partition the utterance so that more i-vectors can be produced for estimating the LDA and WCCN projection matrix. It helps the LDA to find a subspace with less intra-speaker variation by alleviating the numerical problem.

5.4.4 Alleviating Data Imbalance Problem

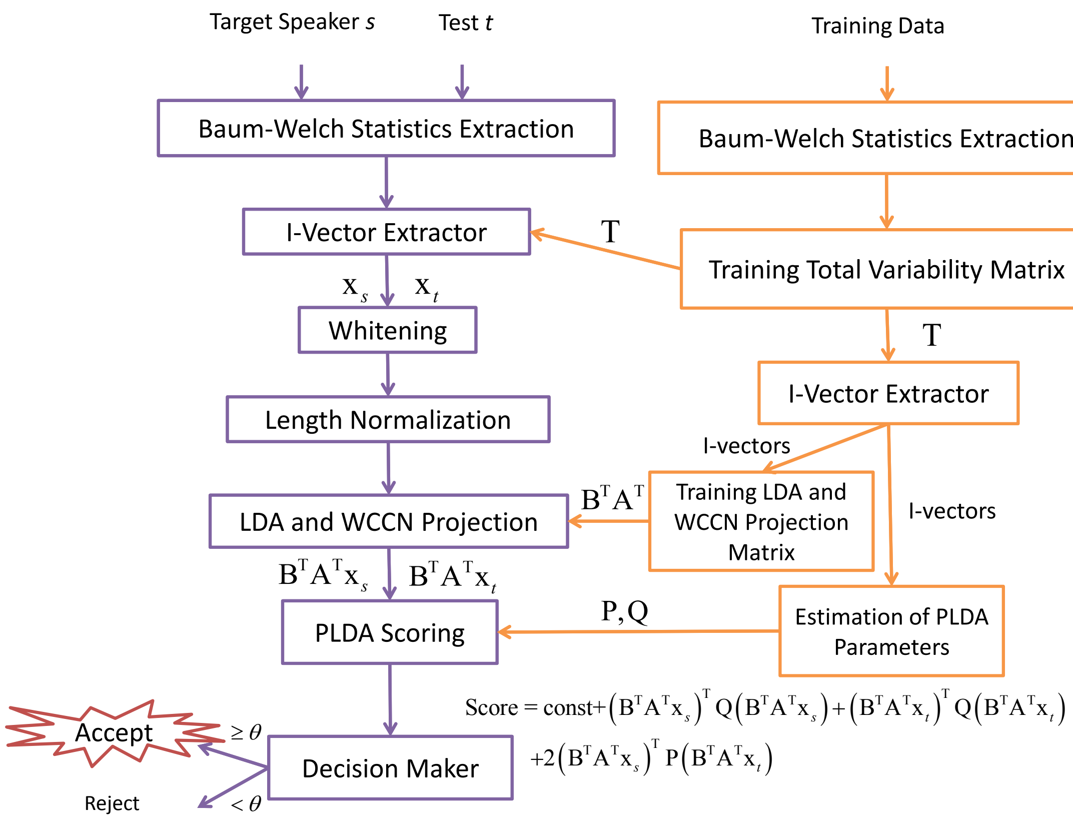
A simple strategy for solving the data imbalance problem in SVM scoring is to increase the number of minority-class samples for training the SVMs [59]. One may use more enrollment utterances, which means more i-vectors from the speaker class. However, this strategy shifts the burden to the client speakers by requesting them to provide multiple enrollment utterances, which may not be practical. With UP-AVR, a number of i-vectors can be produced for training the target-speaker dependent SVM even if the target-speaker provides only one enrollment utterance, which can enhance the influence of the target-speaker data on the SVM's decision boundary.

Chapter 6

GAUSSIAN-PLDA WITH SPARSE KERNEL MACHINES

In Chapter 4, we introduce two popular scoring approaches in i-vector based speaker verification systems: PLDA scoring (Section 4.3.3) and cosine distance scoring (Section 4.3.1). Given the i-vector of a test utterance and the i-vectors of the claimed target-speaker, these scoring methods compute the scores without referring to any other speakers. Taking PLDA scoring as an example, the verification decision is based on the likelihood ratio (LR) score derived from two hypotheses: (1) the test i-vector and the target-speaker i-vector are from the same speaker and (2) these two i-vectors are from two different speakers. Because the computation of the likelihood ratio does not involve other i-vectors, this scoring method *implicitly* uses background information through the UBM [20], total variability matrix [11], PLDA's factor loading matrix. This LR scoring method is computationally efficient, however, the implicit use of background information is a drawback of the method.

To address the limitation of these scoring methods, this thesis proposes a method called PLDA-SVM scoring that uses empirical kernel maps to create a PLDA score space for each target speaker and train an SVM that operates in the score space to produce verification scores [23–25]. This method captures the discrimination between a target-speaker and non-target-speakers in the SVM weights as well as in the score vectors that live in an empirical score space. With the new protocol brought by NIST SRE, this thesis shows that PLDA-SVM scoring not only performs



Target Speaker s
Test t
Training Data
Baum-Welch Statistics Extraction
Baum-Welch Statistics Extraction
I-Vector Extractor
T
Training Total Variability Matrix
x_s x_t
Whitening
T
I-Vector Extractor
Length Normalization
I-vectors
$B^T A^T$
Training LDA and WCCN Projection Matrix
I-vectors
LDA and WCCN Projection
$B^T A^T x_s$ $B^T A^T x_t$
P,Q
PLDA Scoring
Estimation of PLDA Parameters
$\text{Score} = \text{const} + (B^T A^T x_s)^T Q (B^T A^T x_s) + (B^T A^T x_t)^T Q (B^T A^T x_t) + 2 (B^T A^T x_s)^T P (B^T A^T x_t)$
Accept
$\geq \theta$
Decision Maker
Reject
$< \theta$

Figure 6.1: The flow chart of i-vector/PLDA speaker verification system. **T** is the total variability matrix. Refer to the caption in Figure 4.1 for the definitions of $\mathbf{x}_s$, $\mathbf{x}_t$, **A**, and **B** and Eq. 6.13 for **P** and **Q**.

significantly better than the conventional PLDA scoring and utilizes the multiple enrollment utterances of target speakers effectively, but also opens up opportunity for adopting sparse kernel machines for PLDA-based speaker verification systems.

Figure 6.1 illustrates the structure of i-vector/PLDA speaker verification system. The i-vector extraction and the training of LDA and WCCN projection matrices have already been introduced in Chapter 4. This chapter focuses on the theory of Gaussian-PLDA. It also introduces the idea of empirical kernel maps and applies the idea to sparse kernel machines – SVM and relevance vector machine (RVM) [93]. The main difference between SVM and RVM lies in the learning methods. The former is based on structural risk minimization, whereas the latter is based on a fully probabilistic framework. RVMs do not suffer from the limitations of SVM [93], but can obtain a comparable performance as SVM.

6.1 Gaussian–PLDA

The aim of LDA [16] is to find an orthogonal subspace for minimizing the within-class variation and maximizing the between-class separation. In 2007, Prince and Elder [14] proposed a probabilistic approach to the same problem and named the method probabilistic LDA. Kenny [12] applied a similar spirit but replaced the Gaussian distribution of the i-vectors with Student's t-distribution and used a fully Bayesian approach to estimating the model parameters. The resulting model is commonly referred to as heavy-tailed PLDA in the literature. Recently, Garcia-Romero and Espy-Wilson [13] showed that transforming the heavy-tailed distributed i-vectors by whitening and length normalization enables the use of Gaussian assumptions for the PLDA model. It was found that the resulting model, namely Gaussian PLDA, can achieve performance equivalent to that of more complicated systems based on the heavy-tailed assumption.

The idea of Gaussian PLDA is based on factor analysis [16, 56]. The relationship between LDA and PLDA is similar to the relationship between PCA [16, 56] and probabilistic PCA [16, 56]. The former requires speaker labels, whereas the latter dose not. In this thesis, we focus on Gaussian–PLDA.

6.1.1 Effect of Length Normalization on I-vectors

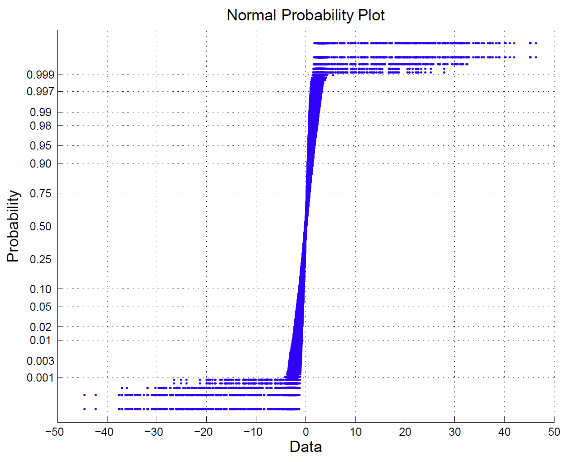
Because i-vectors have non-Gaussian behavior, the assumption of Gaussian–PLDA is not appropriate for i-vectors. However, after performing length normalization on i-vectors, Gaussian–PLDA can be applied and achieve similar performance as heavy-tailed PLDA [13]. Therefore, whitening and length normalization play an important role as a preprocessing step in Gaussian–PLDA.

Given an i-vector $\mathbf{x}$, the length-normalized i-vector is represented as:

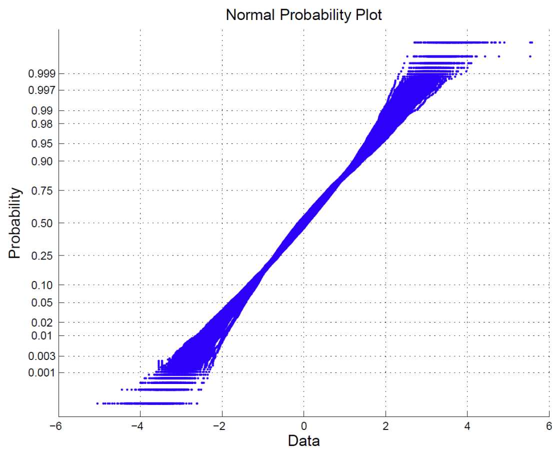
$$\mathbf{x}_{LN} = \frac{\mathbf{x}}{\|\mathbf{x}\|} \tag{6.1}$$

It is interesting to investigate the effect of length normalization on i-vectors. To this end, 5,251 microphone enrollment i-vectors from 5,251 male speakers in NIST 2010 SRE were used for the investigation. Each utterance was used to produce one i-vector, therefore 5,251 i-vectors were utilized for plotting the i-vector distributions. To investigate the distribution of i-vectors, we modified the "normplot" function in MATLAB by commenting out the reference line and changing the symbol of sample data from "+" to dots. The distributions of individual dimensions in the i-vectors are overlaid on the same normplot for ease of comparison. The benefit of this tool is that it helps researchers to judge whether the data follow a Gaussian distribution through inspecting the plot. If the data are Gaussian, the plot will be linear.

Figure 6.2(a) shows that there are a lot of outliers in the distribution of i-vectors, which indicates the non-Gaussian behavior of i-vectors. After length normalization, the norm plots of i-vectors in Figure 6.2(b) become more linear. In other words, the distribution of i-vectors after length normalization is close to a Gaussian distribution, which explains the excellent performance of length normalization in Gaussian-PLDA



(a) w/o length normalization



(b) w/ length normalization

Figure 6.2: The effect of length normalization on the distribution of i-vectors. (a) The distribution of i-vectors without length normalization. (b)The distribution of i-vectors with length normalization. The normal probability plots help to check whether the data follow a Gaussian distribution. If the data are Gaussian, the plots will be linear.

based speaker verification.

6.1.2 Estimation of PLDA Parameters

Given a set of D-dimensional length-normalized [13] i-vectors $\mathcal{X} = \{\mathbf{x}_{ij}; i = 1, \ldots, N; j = 1, \ldots, H_i\}$ obtained from N training speakers, each having H_i i-vectors, we aim to estimate the latent variables $\mathcal{Z} = \{\mathbf{z}_i;\ i = 1, \ldots, N\}$ and parameters $\boldsymbol{\theta} = \{\mathbf{m}, \mathbf{V}, \mathbf{\Sigma}\}$ of a factor analyzer [14]:

$$\begin{aligned} &\mathbf{x}_{ij} = \mathbf{m} + \mathbf{V}\mathbf{z}_i + \boldsymbol{\epsilon}_{ij}, \\ &\mathbf{x}_{ij}, \mathbf{m} \in \Re^D,\ \ \mathbf{V} \in \Re^{D\times M},\ \ \mathbf{z}_i \in \Re^M,\ \ \boldsymbol{\epsilon}_{ij} \in \Re^D \end{aligned} \tag{6.2}$$

where $\mathbf{V}$ is a $D \times M$ factor loading matrix ($M < D$), $\mathbf{m}$ is the global mean of $\mathcal{X}$, $\mathbf{z}_i$'s are the speaker factors, and $\boldsymbol{\epsilon}_{ij}$'s are residual noise assumed to follow a Gaussian distribution with zero mean and covariance $\mathbf{\Sigma}$. Because the i-vector dimension D is sufficiently small, it is possible to estimate the full covariance matrix [12, 13]. We used full covariance for $\mathbf{\Sigma}$ for all systems in this thesis.

Note that Eq. 6.2 assumes that the prior of the latent factors and the condition distribution of i-vectors follow a Gaussian distribution:

$$p(\mathbf{z}) = \mathcal{N}(\mathbf{z}|\mathbf{0}, \mathbf{I}) \ \text{ and } \ p(\mathbf{x}|\mathbf{z}) = \mathcal{N}(\mathbf{x}|\mathbf{m} + \mathbf{V}\mathbf{z}, \mathbf{\Sigma}). \tag{6.3}$$

According to these assumptions and the definition of marginal distributions, the generative model of PLDA (Eq. 6.2) obeys the following distribution:

$$\begin{aligned} p(\mathbf{x}) &= \int p(\mathbf{x}|\mathbf{z})p(\mathbf{z})d\mathbf{z} = \int \mathcal{N}(\mathbf{x}|\mathbf{m} + \mathbf{V}\mathbf{z}, \mathbf{\Sigma})\mathcal{N}(\mathbf{z}|\mathbf{0}, \mathbf{I})d\mathbf{z} \\ &= \mathcal{N}(\mathbf{x}|\mathbf{m}, \mathbf{V}\mathbf{V}^{\mathsf{T}} + \mathbf{\Sigma}) \end{aligned} \tag{6.4}$$

where the covariance matrix of the marginal distribution $p(\mathbf{x})$ comprises two terms:

$\mathbf{V}\mathbf{V}^{\mathsf{T}}$ representing the between-speaker variation and $\boldsymbol{\Sigma}$ describing the channel variation and other variability that cannot be represented by $\mathbf{V}\mathbf{V}^{\mathsf{T}}$.

Denote the parameters of the PLDA model as $\boldsymbol{\theta} = \{\mathbf{m}, \mathbf{V}, \boldsymbol{\Sigma}\}$, an EM algorithm is used to estimate the parameters in $\boldsymbol{\theta}$ [14, 56]. In the E-step, given the observed data $\mathcal{X} = \{\mathbf{x}_{ij}; i = 1, \ldots, N; j = 1, \ldots, H_i\}$ and denote $\mathcal{X}_i = \{\mathbf{x}_{i1}, \ldots, \mathbf{x}_{iH_i}\}$ as the set of i-vectors from speaker i, the posterior probability distribution $p(\mathbf{z}_i|\mathcal{X}_i)$ over each hidden variable $\mathbf{z}_i$ is computed by Bayes' rule:

$$p(\mathbf{z}_i|\mathcal{X}_i, \boldsymbol{\theta}) \propto \prod_{j=1}^{H_i} p(\mathbf{x}_{ij}|\mathbf{z}_i, \boldsymbol{\theta})p(\mathbf{z}_i), \qquad i = 1, \ldots, N. \tag{6.5}$$

Because both terms on the right-hand-side of Eq. 6.5 are Gaussian, the term on the left will also be a Gaussian. Therefore, the first and second moments of $p(\mathbf{z}_i|\mathcal{X}_i, \boldsymbol{\theta})$ are given by:

$$\begin{aligned} \mathbb{E}\left(\mathbf{z}_i|\mathcal{X}_i\right) &= \mathbf{L}_i^{-1}\mathbf{V}^{\mathsf{T}}\boldsymbol{\Sigma}^{-1}\sum_{j=1}^{H_i}(\mathbf{x}_{ij} - \mathbf{m}) \\ \mathbb{E}\left(\mathbf{z}_i\mathbf{z}_i^{\mathsf{T}}|\mathcal{X}_i\right) &= \mathbf{L}_i^{-1} + \mathbb{E}\left(\mathbf{z}_i|\mathcal{X}_i\right)\mathbb{E}\left(\mathbf{z}_i|\mathcal{X}_i\right)^{\mathsf{T}} \end{aligned} \tag{6.6}$$

where

$$\mathbf{L}_i = \mathbf{I} + H_i\mathbf{V}^{\mathsf{T}}\boldsymbol{\Sigma}^{-1}\mathbf{V}. \tag{6.7}$$

In the M-step, the new parameter $\boldsymbol{\theta}' = \{\mathbf{m}', \mathbf{V}', \boldsymbol{\Sigma}'\}$ can be obtained by maximizing the auxiliary function with respect to $\boldsymbol{\theta}'$.

$$\mathbf{Q}(\boldsymbol{\theta}'|\boldsymbol{\theta}) = \mathbb{E}\left\{\sum_{ij}\log\left[p(\mathbf{x}_{ij}|\mathbf{z}_i, \boldsymbol{\theta}')p(\mathbf{z}_i)\right]\middle|\mathcal{X}, \boldsymbol{\theta}\right\} \tag{6.8}$$

$$= \mathbb{E}\left\{ \sum_{ij} \log\left[\mathcal{N}(\mathbf{x}_{ij}|\mathbf{m}' + \mathbf{V}'\mathbf{z}, \boldsymbol{\Sigma}')\mathcal{N}(\mathbf{z}_i|\mathbf{0}, \mathbf{I})\right] \middle| \mathcal{X}, \boldsymbol{\theta} \right\}.$$

Because the term $p(\mathbf{z}_i)$ is independent of the parameter $\boldsymbol{\theta}$, it can be ignored in Eq. 6.8. Therefore, the auxiliary function can be simplified as:

$$\begin{aligned} \mathbf{Q}(\boldsymbol{\theta}'|\boldsymbol{\theta}) &= \mathbb{E}\left\{ \sum_{ij} \left[-\frac{1}{2} D \log(2\pi) - \frac{1}{2}\log|\boldsymbol{\Sigma}'| \right] \right\} \\ &- \mathbb{E}\left\{ \sum_{ij} \left[\frac{1}{2} (\mathbf{x}_{ij} - \mathbf{m}' - \mathbf{V}'\mathbf{z}_i)^{\mathsf{T}} \{\boldsymbol{\Sigma}'\}^{-1} (\mathbf{x}_{ij} - \mathbf{m}' - \mathbf{V}'\mathbf{z}_i) \right] \right\} \end{aligned} \tag{6.9}$$

We maximize the auxiliary function in Eq. 6.9 by taking its derivatives with respect to $\boldsymbol{\theta}' = \{\mathbf{m}', \mathbf{V}', \boldsymbol{\Sigma}'\}$, which result in following equations:

$$\begin{aligned} \mathbf{m}' &= \frac{\sum_{ij} \mathbf{x}_{ij}}{\sum_i H_i}; \\ \mathbf{V}' &= \left[\sum_{ij} (\mathbf{x}_{ij} - \mathbf{m}') \mathbb{E}\left(\mathbf{z}_i|\mathcal{X}_i\right)^{\mathsf{T}} \right] \left[\sum_{ij} \mathbb{E}\left(\mathbf{z}_i\mathbf{z}_i^{\mathsf{T}}|\mathcal{X}_i\right) \right]^{-1}; \\ \boldsymbol{\Sigma}' &= \frac{1}{\sum_{i=1}^{N} H_i} \left\{ \sum_{i=1}^{N} \sum_{j=1}^{H_i} \left[(\mathbf{x}_{ij} - \mathbf{m}')(\mathbf{x}_{ij} - \mathbf{m}')^{\mathsf{T}} - \mathbf{V}'\mathbb{E}\left(\mathbf{z}_i|\mathcal{X}_i\right)(\mathbf{x}_{ij} - \mathbf{m}')^{\mathsf{T}} \right] \right\}. \end{aligned} \tag{6.10}$$

6.1.3 PLDA Scoring

Given a length-normalized [13] test i-vector $\mathbf{x}_t$ and target-speaker's i-vector $\mathbf{x}_s$, the log-likelihood ratio score can be computed as follows [13]:

$$S_{\text{LR}}(\mathbf{x}_s, \mathbf{x}_t) = \log\left[\frac{P(\mathbf{x}_s, \mathbf{x}_t|\text{same speaker})}{P(\mathbf{x}_s, \mathbf{x}_t|\text{different speakers})} \right] \tag{6.11}$$

$$
\begin{aligned}
&= \log\left[\frac{\int p(\mathbf{x}_s,\mathbf{x}_t,\mathbf{z}|\boldsymbol{\theta})\mathrm{d}\mathbf{z}}{\int p(\mathbf{x}_s,\mathbf{z}_s|\boldsymbol{\theta})\mathrm{d}\mathbf{z}_s\int p(\mathbf{x}_t,\mathbf{z}_t|\boldsymbol{\theta})\mathrm{d}\mathbf{z}_t}\right]\\
&= \log\left[\frac{\int p(\mathbf{x}_s,\mathbf{x}_t|\mathbf{z},\boldsymbol{\theta})p(\mathbf{z})\mathrm{d}\mathbf{z}}{\int p(\mathbf{x}_s|\mathbf{z}_s,\boldsymbol{\theta})p(\mathbf{z}_s)\mathrm{d}\mathbf{z}_s\int p(\mathbf{x}_t|\mathbf{z}_t,\boldsymbol{\theta})p(\mathbf{z}_t)\mathrm{d}\mathbf{z}_t}\right]\\
&= \log\mathcal{N}\left(\begin{bmatrix}\mathbf{x}_s\\ \mathbf{x}_t\end{bmatrix};\begin{bmatrix}\mathbf{m}\\ \mathbf{m}\end{bmatrix},\begin{bmatrix}\boldsymbol{\Sigma}_{tot} & \boldsymbol{\Sigma}_{ac}\\ \boldsymbol{\Sigma}_{ac} & \boldsymbol{\Sigma}_{tot}\end{bmatrix}\right)\\
&- \log\mathcal{N}\left(\begin{bmatrix}\mathbf{x}_s\\ \mathbf{x}_t\end{bmatrix};\begin{bmatrix}\mathbf{m}\\ \mathbf{m}\end{bmatrix},\begin{bmatrix}\boldsymbol{\Sigma}_{tot} & 0\\ 0 & \boldsymbol{\Sigma}_{tot}\end{bmatrix}\right)
\end{aligned}
$$

where $\boldsymbol{\Sigma}_{tot} = \mathbf{V}\mathbf{V}^{\mathsf{T}} + \boldsymbol{\Sigma}$ and $\boldsymbol{\Sigma}_{ac} = \mathbf{V}\mathbf{V}^{\mathsf{T}}$. Details of PLDA scoring can be found in [12, 13, 23]. Since $\mathbf{m}$ is a global offset that can be computed firstly and removed from all i-vectors, we set it to zero and expand Eq. 6.11 to obtain:

$$
\begin{aligned}
\text{score} &= -\frac{1}{2}\begin{bmatrix}\mathbf{x}_s\\ \mathbf{x}_t\end{bmatrix}^{\mathrm{T}}\begin{bmatrix}\boldsymbol{\Sigma}_{tot} & \boldsymbol{\Sigma}_{ac}\\ \boldsymbol{\Sigma}_{ac} & \boldsymbol{\Sigma}_{tot}\end{bmatrix}^{-1}\begin{bmatrix}\mathbf{x}_s\\ \mathbf{x}_t\end{bmatrix} + \frac{1}{2}\begin{bmatrix}\mathbf{x}_s\\ \mathbf{x}_t\end{bmatrix}^{\mathrm{T}}\begin{bmatrix}\boldsymbol{\Sigma}_{tot} & 0\\ 0 & \boldsymbol{\Sigma}_{tot}\end{bmatrix}^{-1}\begin{bmatrix}\mathbf{x}_s\\ \mathbf{x}_t\end{bmatrix} + \text{const}\\
&= -\frac{1}{2}\begin{bmatrix}\mathbf{x}_s\\ \mathbf{x}_t\end{bmatrix}^{\mathrm{T}}\begin{bmatrix}(\boldsymbol{\Sigma}_{tot}-\boldsymbol{\Sigma}_{ac}\boldsymbol{\Sigma}_{tot}^{-1}\boldsymbol{\Sigma}_{ac})^{-1} & -\boldsymbol{\Sigma}_{tot}^{-1}\boldsymbol{\Sigma}_{ac}(\boldsymbol{\Sigma}_{tot}-\boldsymbol{\Sigma}_{ac}\boldsymbol{\Sigma}_{tot}^{-1}\boldsymbol{\Sigma}_{ac})^{-1}\\ -(\boldsymbol{\Sigma}_{tot}-\boldsymbol{\Sigma}_{ac}\boldsymbol{\Sigma}_{tot}^{-1}\boldsymbol{\Sigma}_{ac})^{-1}\boldsymbol{\Sigma}_{ac}\boldsymbol{\Sigma}_{tot}^{-1} & (\boldsymbol{\Sigma}_{tot}-\boldsymbol{\Sigma}_{ac}\boldsymbol{\Sigma}_{tot}^{-1}\boldsymbol{\Sigma}_{ac})^{-1}\end{bmatrix}\begin{bmatrix}\mathbf{x}_s\\ \mathbf{x}_t\end{bmatrix}\\
&+ \frac{1}{2}\begin{bmatrix}\mathbf{x}_s\\ \mathbf{x}_t\end{bmatrix}^{\mathrm{T}}\begin{bmatrix}\boldsymbol{\Sigma}_{tot}^{-1} & 0\\ 0 & \boldsymbol{\Sigma}_{tot}^{-1}\end{bmatrix}\begin{bmatrix}\mathbf{x}_s\\ \mathbf{x}_t\end{bmatrix} + \text{const}\\
&= \frac{1}{2}\begin{bmatrix}\mathbf{x}_s\\ \mathbf{x}_t\end{bmatrix}^{\mathrm{T}}\begin{bmatrix}\boldsymbol{\Sigma}_{tot}^{-1}-(\boldsymbol{\Sigma}_{tot}-\boldsymbol{\Sigma}_{ac}\boldsymbol{\Sigma}_{tot}^{-1}\boldsymbol{\Sigma}_{ac})^{-1} & \boldsymbol{\Sigma}_{tot}^{-1}\boldsymbol{\Sigma}_{ac}(\boldsymbol{\Sigma}_{tot}-\boldsymbol{\Sigma}_{ac}\boldsymbol{\Sigma}_{tot}^{-1}\boldsymbol{\Sigma}_{ac})^{-1}\\ \boldsymbol{\Sigma}_{tot}^{-1}\boldsymbol{\Sigma}_{ac}(\boldsymbol{\Sigma}_{tot}-\boldsymbol{\Sigma}_{ac}\boldsymbol{\Sigma}_{tot}^{-1}\boldsymbol{\Sigma}_{ac})^{-1} & \boldsymbol{\Sigma}_{tot}^{-1}-(\boldsymbol{\Sigma}_{tot}-\boldsymbol{\Sigma}_{ac}\boldsymbol{\Sigma}_{tot}^{-1}\boldsymbol{\Sigma}_{ac})^{-1}\end{bmatrix}\begin{bmatrix}\mathbf{x}_s\\ \mathbf{x}_t\end{bmatrix} + \text{const}\\
&= \frac{1}{2}\begin{bmatrix}\mathbf{x}_s^{\mathrm{T}} & \mathbf{x}_t^{\mathrm{T}}\end{bmatrix}\begin{bmatrix}\mathbf{Q} & \mathbf{P}\\ \mathbf{P} & \mathbf{Q}\end{bmatrix}\begin{bmatrix}\mathbf{x}_s\\ \mathbf{x}_t\end{bmatrix} + \text{const}
\end{aligned}
\tag{6.12}
$$

$$= \frac{1}{2}[\mathbf{x}_s^{\mathrm{T}}\mathbf{Q}\mathbf{x}_s + \mathbf{x}_s^{\mathrm{T}}\mathbf{P}\mathbf{x}_t + \mathbf{x}_t^{\mathrm{T}}\mathbf{P}\mathbf{x}_s + \mathbf{x}_t^{\mathrm{T}}\mathbf{Q}\mathbf{x}_t] + \text{const}$$

$$= \frac{1}{2}[\mathbf{x}_s^{\mathrm{T}}\mathbf{Q}\mathbf{x}_s + 2\mathbf{x}_s^{\mathrm{T}}\mathbf{P}\mathbf{x}_t + \mathbf{x}_t^{\mathrm{T}}\mathbf{Q}\mathbf{x}_t] + \text{const}$$

where

$$\begin{aligned}\mathbf{Q} &= \mathbf{\Sigma}_{tot}^{-1} - (\mathbf{\Sigma}_{tot} - \mathbf{\Sigma}_{ac}\mathbf{\Sigma}_{tot}^{-1}\mathbf{\Sigma}_{ac})^{-1}, \\ \mathbf{P} &= \mathbf{\Sigma}_{tot}^{-1}\mathbf{\Sigma}_{ac}(\mathbf{\Sigma}_{tot} - \mathbf{\Sigma}_{ac}\mathbf{\Sigma}_{tot}^{-1}\mathbf{\Sigma}_{ac})^{-1}.\end{aligned} \tag{6.13}$$

Eq. 6.11 suggests that the verification process focuses on the likelihood that the two i-vectors share the same latent variable rather than the actual value of the latent variable [14]. Therefore, the verification decision is based on the likelihood ratio (LR) score derived from two hypotheses: (1) the test i-vector and the target-speaker i-vector are from the same speaker and (2) these two i-vectors are from two different speakers. Hence, a high verification score means that the observed i-vectors are more likely to be generated by the same speaker (i.e., a single latent variable) than from two different speakers (i.e., two distinct latent variables).

6.2 *Multi-Utterance Scoring for PLDA Speaker Verification*

Compared with previous SREs, NIST 2012 SRE [7] introduces some new protocols that help researchers to enhance the performance of speaker verification systems. One of the new protocols is that some target speakers have multiple enrollment utterances. In i-vector/PLDA framework, common approaches to dealing with multiple enrollment utterances include averaging the i-vectors of the enrollment utterances, averaging the PLDA scores and the so-called *by-the-book* multi-session PLDA scoring [94–96]. The first two methods achieve similar performance [97]. The performance of *by-the-book* multi-session PLDA scoring is poorer than that of averaging i-vectors [95]. The poor performance is caused by the variable numbers of enrollment utterances and the improper assumption that the i-vectors generated from multiple enrollment utterances are conditionally independent [95]. In [98], the authors proposed computing the weighted average of the enrollment i-vectors. However, the performance of this weighted averaging approach is even poorer than that of simply averaging the i-vectors.

In this thesis, three common methods used for multi-utterance scoring in i-vector/PLDA systems are introduced. They are i-vectors averaging, PLDA score averaging and *by-the-book* multi-session PLDA scoring:

- **I-vector Averaging (I-vector Avg.)**: Given H_s multi-enrollment utterances of a target speaker, this method computes the average of all of the target-dependent i-vectors and scores the averaged i-vectors against the i-vector of the test utterance. Specifically, the PLDA LR score of multi-enrollment sessions is defined as:

  $$S_{\text{LR}}(\mathcal{X}_s, \mathbf{x}_t) \equiv S_{\text{LR}}(\mathbf{x}_{\text{avg}}, \mathbf{x}_t) \tag{6.14}$$

  where

  $$\mathbf{x}_{\text{avg}} = \frac{1}{H_s} \sum_{\mathbf{x}_s \in \mathcal{X}_s} \mathbf{x}_s, \tag{6.15}$$

  $\mathbf{x}_t$ is the test i-vectors, and $\mathcal{X}_s$ comprises a set of enrollment i-vectors for speaker s.

- **PLDA Score averaging (Score Avg.)**: The idea is to score a test i-vector against each of the enrollment i-vectors of a target speaker and then average the scores. Specifically, the averaged PLDA score is given by:

$$S_{\mathrm{LR}}(\mathcal{X}_s, \mathbf{x}_t) \equiv \frac{1}{H_s} \sum_{\mathbf{x}_s \in \mathcal{X}_s} S_{\mathrm{LR}}(\mathbf{x}_s, \mathbf{x}_t) \tag{6.16}$$

- ***By-the-book*** **Multi-session PLDA Scoring**: The idea is to expand the traditional PLDA scoring (one against one scoring) to N against one (*by-the-book*) scoring [94–96]. The equation of *by-the-book* scoring [96] is:

$$\begin{aligned} S_{\mathrm{LR}}(\mathcal{X}_s, \mathbf{x}_t) &\equiv S_{\mathrm{LR}}(\mathbf{x}_{s,1}, \ldots, \mathbf{x}_{s,H_s}, \mathbf{x}_t) \\ &= \log \mathcal{N}\left(\begin{bmatrix} \mathbf{x}_{s,1} \\ \vdots \\ \mathbf{x}_{s,H_s} \\ \mathbf{x}_t \end{bmatrix} ; \begin{bmatrix} 0 \\ \vdots \\ 0 \end{bmatrix} , \begin{bmatrix} \mathbf{A}_{11} & \mathbf{A}_{12} \\ \mathbf{A}_{21} & \mathbf{A}_{22} \end{bmatrix} \right) \\ &- \log \mathcal{N}\left(\begin{bmatrix} \mathbf{x}_{s,1} \\ \vdots \\ \mathbf{x}_{s,H_s} \\ \mathbf{x}_t \end{bmatrix} ; \begin{bmatrix} 0 \\ \vdots \\ 0 \end{bmatrix} , \begin{bmatrix} \mathbf{A}_{11} & 0 \\ 0 & \mathbf{A}_{22} \end{bmatrix} \right) \end{aligned} \tag{6.17}$$

where $\mathbf{A}_{11}$ is an $H_s \times H_s$ block matrix and $\mathbf{A}_{12}$ is a $H_s \times 1$ matrix.

$$\begin{aligned} &\mathbf{A}_{11} \equiv \begin{bmatrix} \boldsymbol{\Sigma}_{tot} & \boldsymbol{\Sigma}_{ac} & \cdots & \boldsymbol{\Sigma}_{ac} \\ \boldsymbol{\Sigma}_{ac} & \boldsymbol{\Sigma}_{tot} & \ddots & \vdots \\ \vdots & \ddots & \ddots & \boldsymbol{\Sigma}_{ac} \\ \boldsymbol{\Sigma}_{ac} & \cdots & \boldsymbol{\Sigma}_{ac} & \boldsymbol{\Sigma}_{tot} \end{bmatrix}; \quad \mathbf{A}_{12} \equiv [\boldsymbol{\Sigma}_{ac} \cdots \boldsymbol{\Sigma}_{ac}]^{\mathsf{T}} \\ &\mathbf{A}_{21} = \mathbf{A}_{12}^{\mathsf{T}}; \quad \mathbf{A}_{22} \equiv \boldsymbol{\Sigma}_{tot} \end{aligned} \tag{6.18}$$

where $\boldsymbol{\Sigma}_{tot}$ and $\boldsymbol{\Sigma}_{ac}$ are introduced in Section 6.1.3. [96] shows that Eq. 6.17 can be further simplified as:

$$S_{\text{LR}}(\mathcal{X}_s, \mathbf{x}_t) = \hat{\mathbf{x}}_s^{\mathsf{T}}\mathbf{Q}_1\hat{\mathbf{x}}_s + \mathbf{x}_t^{\mathsf{T}}\mathbf{Q}_2\mathbf{x}_t + \hat{\mathbf{x}}_s^{\mathsf{T}}\mathbf{P}_1\mathbf{x}_t + \mathbf{x}_t^{\mathsf{T}}\mathbf{P}_2\hat{\mathbf{x}}_s \tag{6.19}$$

where

$$\begin{aligned}
\hat{\mathbf{x}}_s &= [\mathbf{x}_{s,1}^{\mathsf{T}} \cdots \mathbf{x}_{s,H_s}^{\mathsf{T}}]^{\mathsf{T}} \\
\mathbf{Q}_1 &= \mathbf{A}_{11}^{-1} - \left(\mathbf{A}_{11} - \mathbf{A}_{12}\mathbf{A}_{22}^{-1}\mathbf{A}_{12}^{\mathsf{T}}\right)^{-1} \\
\mathbf{Q}_2 &= \mathbf{A}_{22}^{-1} - \left(\mathbf{A}_{22} - \mathbf{A}_{12}^{\mathsf{T}}\mathbf{A}_{11}^{-1}\mathbf{A}_{12}\right)^{-1} \\
\mathbf{P}_1 &= \mathbf{A}_{11}^{-1}\mathbf{A}_{12} - \left(\mathbf{A}_{22} - \mathbf{A}_{12}^{\mathsf{T}}\mathbf{A}_{11}^{-1}\mathbf{A}_{12}\right)^{-1} \\
\mathbf{P}_2 &= \mathbf{A}_{22}^{-1}\mathbf{A}_{12}^{\mathsf{T}} - \left(\mathbf{A}_{11} - \mathbf{A}_{12}\mathbf{A}_{22}^{-1}\mathbf{A}_{12}^{\mathsf{T}}\right)^{-1}
\end{aligned} \tag{6.20}$$

Although the theory of *by-the-book* multi-session PLDA scoring seems much more reasonable, the performance of this method is typically poorer than that of i-vector averaging [95]. Therefore, only the performance of i-vector averaging and PLDA score averaging will be reported in this thesis.

6.3 Empirical Kernel Maps for PLDA-SVM Scoring

Because the computation of the likelihood ratio does not involve other i-vectors (Section 6.1.3), PLDA scoring method (Eq. 6.11) *implicitly* uses background information through the UBM [20], total variability matrix [11], PLDA's factor loading matrix. This LR scoring method is computationally efficient, However, the implicit use of background information is a drawback of this method.

To address the limitation of these scoring methods, we have recently proposed to apply SVM scoring with empirical kernel maps for taking the background speaker information *explicitly* during the scoring process [23–25]. This method captures the discrimination between a target-speaker and non-target-speakers in the SVM weights as well as in the score vectors that live in an empirical score space. Specifically, for each target speaker, an empirical score space with dimension equal to the number of training i-vectors for this target speaker is defined by using the idea of empirical kernel maps [26–28]. Given an i-vector, a score vector living in this space is formed by computing the LR scores of this i-vector with respect to each of the training i-vectors. A speaker-dependent SVM – referred to as PLDA-SVM – can then be trained using the training score vectors. During verification, given a test i-vector and the target-speaker under test, the LR scores are mapped to a score vector, which is then fed to the target-speaker's SVM to obtain the final test score.

The empirical kernel map proposed in this thesis is also a novel way of incorporating multiple enrollment i-vectors in the scoring process. In fact, the conventional score averaging is a special case of the proposed method. Specifically, when the SVM kernel ($\mathbb{K}$ in Eq. 6.26) is linear and all weights are equal, the proposed method reduces to score averaging.

SVM is one of the back-end classifiers adopted in the original i-vector approach [11]. However, SVM scoring is not very common in other i-vector systems, primary because of its inferior performance when compared with cosine distance scoring [11] and PLDA scoring [12]. The poorer performance of SVM scoring, however, is mainly due to the severe imbalance between the number of target-speaker i-vectors and the number of background speaker i-vectors. Before NIST 2012 SRE, there is only one i-vector per target speaker, because there is only one enrollment session per target speaker. Although NIST 2012 SRE provides multiple enrollment utterances for some target speakers, there are also target speakers who have one or a few enrollment sessions only. This difficulty, however, can be overcome by a technique called UP-AVR [15, 21] as mentioned in Chapter 5. This technique has successfully boosted the performance of GMM-SVM [1, 10, 99] and i-vector based systems [5].

There has been previous work that uses discriminative models for PLDA scoring. For example, in [100–103], for each verification trial, the LR score of a test i-vector and a target-speaker i-vector is expressed as a dot product between a speaker-independent weight vector and a vector whose elements are derived from these two i-vectors in the trial. The weight vector is discriminatively trained by a logistic regression or SVM training algorithm using all of the available i-vector pairs (same-speaker pairs and different-speaker pairs) in the development set. Essentially, this method trains a binary classify that takes a pair of i-vectors as input and produces a score that better reflects the similarity/difference of the pair. This idea has been extended to gender-independent PLDA scoring in [104]. Another related work [105], which is based on [100], unifies PLDA and second degree polynomial kernels for SVM scoring.

The SVM scoring method proposed in this thesis is different from these previous studies in three aspects. First, all of these studies use a large number of same-speaker and different-speaker i-vector pairs to train a speaker-independent SVM for scoring. As a result, the discrimination between the same-speaker and different-speaker pairs are encoded in the SVM weights. On the other hand, the proposed method captures the discrimination between the target-speaker and impostors in the SVM weights as well as in the score vectors that live in the empirical feature space. Second, in the proposed method, the SVMs can be optimized for individual target-speakers, whereas the speaker-independent SVM in [100,101,104,105] is optimized for all target speakers. Third, because the dimension of the empirical feature space depends on the number of non-target-speaker i-vectors, it is possible to limit the dimension so that more flexible non-linear SVMs can be applied to the score vectors.

The empirical kernel map in this thesis is related to the anchor models [106–108]. However, in the anchor model, a test utterance is projected into a space represented by a set of reference speakers *unrelated* to the target-speakers, whereas the empirical feature space is represented by the target speaker and a set of non-target speakers.

6.3.1 Empirical Kernel Maps

To make better use of multiple enrollment utterances of target speakers and to explicitly use the information of background speakers, we have recently proposed a speaker-dependent discriminative model that incorporates the empirical kernel maps for scoring [23–25]. We refer to the mapping from i-vectors to PLDA score vectors as empirical kernel maps.

Assume that target-speaker s has H_s enrollment utterances and that each enrollment utterance leads to one i-vector. Then, H_s i-vectors will be obtained. In case the speaker provides one or a very small number of enrollment utterances only, we can apply an utterance partitioning technique [5] to produce multiple i-vectors from his/her enrollment utterance. Denote these i-vectors as:

$$\mathcal{X}_s = \{\mathbf{x}_{s,1}, \ldots, \mathbf{x}_{s,j}, \ldots, \mathbf{x}_{s,H_s}\}. \tag{6.21}$$

Denote the set of non-target-speaker i-vectors as:[1]

$$\mathcal{X}_b = \{\mathbf{x}_{b,1}, \ldots, \mathbf{x}_{b,i}, \ldots, \mathbf{x}_{b,B}\}\,. \tag{6.22}$$

Therefore, $\mathcal{X} = \{\mathcal{X}_s, \mathcal{X}_b\}$ is the training set for target-speaker s. Figure 6.3 illustrates the process of constructing the empirical kernel maps in i-vector/PLDA speaker verification. Because target speakers have different numbers of enrollment utterances, the dimension of the resulting PLDA score vectors are different for different speakers.

There are several possibilities for the empirical kernel map. We focus on the following two cases.

1. ***Empirical Kernel MAP I***: The definition is written as:

$$\vec{S}_{\text{LR}}(\mathbf{x}, \mathcal{X}_s) = \begin{bmatrix} S_{\text{LR}}(\mathbf{x}, \mathbf{x}_{s,1}) \\ S_{\text{LR}}(\mathbf{x}, \mathbf{x}_{s,2}) \\ \vdots \\ S_{\text{LR}}(\mathbf{x}, \mathbf{x}_{s,H_s}) \end{bmatrix} \tag{6.23}$$

   where $S_{\text{LR}}(\mathbf{x}, \mathbf{x}_{s,j})$ is defined in Eq. 6.12. Therefore, the PLDA score space is defined by target-speaker's i-vectors through the PLDA model. Because H_s is typically small, the dimension of $\vec{S}_{\text{LR}}(\mathbf{x}, \mathcal{X}_s)$ is low.

2. ***Empirical Kernel MAP II***: Let's denote $\mathcal{X} = \{\mathcal{X}_s, \mathcal{X}_b\}$ as the training set

[1]It is not necessary to apply UP-AVR to non-target speakers because non-target i-vectors are abundant.

for target-speaker s. Then,

$$\overrightarrow{S}_{\mathrm{LR}}(\mathbf{x}, \mathcal{X}) = \begin{bmatrix} S_{\mathrm{LR}}(\mathbf{x}, \mathbf{x}_{s,1}) \\ \vdots \\ S_{\mathrm{LR}}(\mathbf{x}, \mathbf{x}_{s,H_s}) \\ S_{\mathrm{LR}}(\mathbf{x}, \mathbf{x}_{b,1}) \\ \vdots \\ S_{\mathrm{LR}}(\mathbf{x}, \mathbf{x}_{b,B'}) \end{bmatrix} \quad (6.24)$$

where the B' non-target i-vectors are selected from the non-target speaker set $\mathcal{X}_b$. To keep the dimension of the score vector low, we recommend $B' < B$. Unlike Empirical Kernel Map I, the score vector in Eq. 6.24 also contains the LR scores of $\mathbf{x}_t$ with respect to the non-target i-vectors. As a result, discriminative information between same-speaker pairs $\{\mathbf{x}_t, \mathbf{x}_{s,j}\}_{j=1}^{H_s}$ and different-speaker pairs $\{\mathbf{x}_t, \mathbf{x}_{b,i}\}_{i=1}^{B'}$ is embedded in the score vector. Note that the vector size in Eq. 6.24 is independent of the number of target-speakers. Therefore, the method is scalable to large systems with thousands of speakers.

6.3.2 PLDA-SVM

This thesis uses PLDA score vectors (via the empirical kernel maps) as the input to the SVMs and applies the speaker-dependent SVMs for i-vector/PLDA speaker verification. Figure 6.4 shows the procedures of PLDA–SVM training. Specifically, for target speaker s, after obtaining the PLDA score vectors of target speaker-class and non-target speaker-class with labels $\{+1, -1\}$, these score vectors are used for training the SVM model for target speaker s.

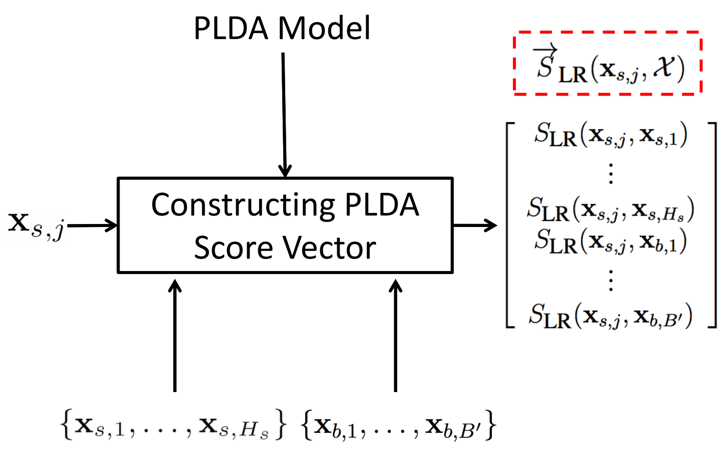

PLDA Model
$\vec{S}_{\text{LR}}(\mathbf{x}_{s,j}, \mathcal{X})$
$\mathbf{x}_{s,j}$
Constructing PLDA Score Vector
$\begin{bmatrix} S_{\text{LR}}(\mathbf{x}_{s,j}, \mathbf{x}_{s,1}) \\ \vdots \\ S_{\text{LR}}(\mathbf{x}_{s,j}, \mathbf{x}_{s,H_s}) \\ S_{\text{LR}}(\mathbf{x}_{s,j}, \mathbf{x}_{b,1}) \\ \vdots \\ S_{\text{LR}}(\mathbf{x}_{s,j}, \mathbf{x}_{b,B'}) \end{bmatrix}$
$\{\mathbf{x}_{s,1}, \ldots, \mathbf{x}_{s,H_s}\}$ $\{\mathbf{x}_{b,1}, \ldots, \mathbf{x}_{b,B'}\}$

Figure 6.3: Construction of empirical kernel maps for i-vector/PLDA speaker verification.

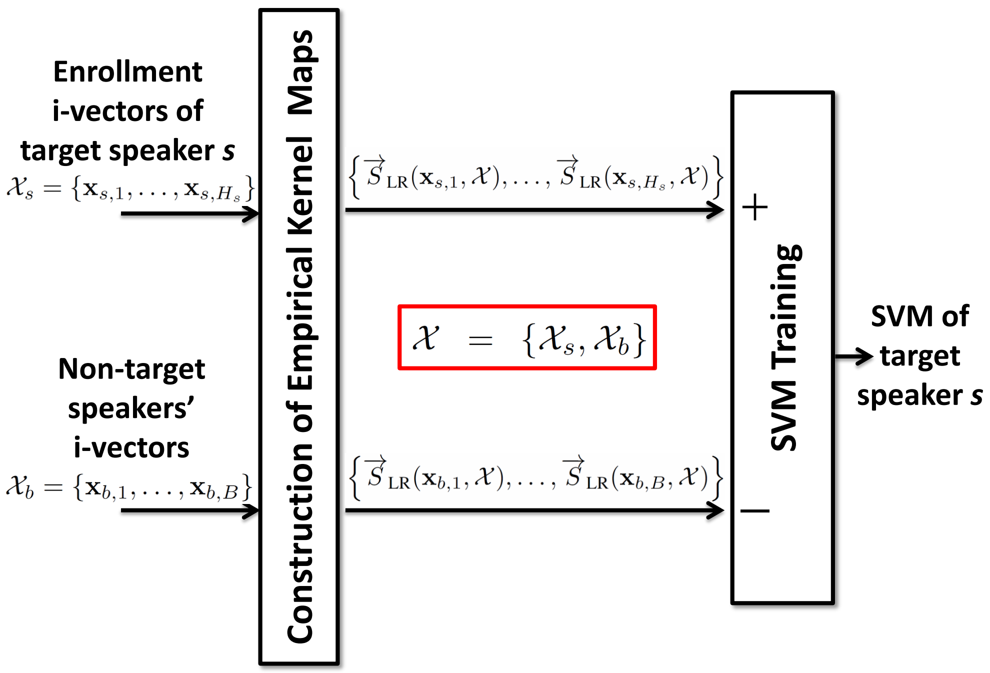

Enrollment i-vectors of target speaker *s*
$\mathcal{X}_s = \{\mathbf{x}_{s,1}, \ldots, \mathbf{x}_{s,H_s}\}$
Non-target speakers' i-vectors
$\mathcal{X}_b = \{\mathbf{x}_{b,1}, \ldots, \mathbf{x}_{b,B}\}$
Construction of Empirical Kernel Maps
$\{\vec{S}_{\text{LR}}(\mathbf{x}_{s,1}, \mathcal{X}), \ldots, \vec{S}_{\text{LR}}(\mathbf{x}_{s,H_s}, \mathcal{X})\}$
$\mathcal{X} = \{\mathcal{X}_s, \mathcal{X}_b\}$
$\{\vec{S}_{\text{LR}}(\mathbf{x}_{b,1}, \mathcal{X}), \ldots, \vec{S}_{\text{LR}}(\mathbf{x}_{b,B}, \mathcal{X})\}$
+
−
SVM Training
SVM of target speaker *s*

Figure 6.4: The procedure of PLDA–SVM training with empirical kernel maps.

Given a test vector $\mathbf{x}_t$, the PLDA-SVM's output is written as

$$S_{\text{SVM}}(\mathbf{x}_t, \mathcal{X}_s, \mathcal{X}_b) = \sum_{j \in \text{SV}_s} \alpha_{s,j} K(\mathbf{x}_t, \mathbf{x}_{s,j}) - \sum_{i \in \text{SV}_b} \alpha_{b,i} K(\mathbf{x}_t, \mathbf{x}_{b,i}) + w_0 \tag{6.25}$$

where SV_s and SV_b contain the indexes of the support vectors corresponding to the speaker class and impostor class, respectively. $\alpha_{s,j}$ and $\alpha_{b,i}$ are the Lagrange multipliers of the SVM. $K(\mathbf{x}_t, \mathbf{x}_{s,j})$ is a kernel function and has the following two forms according to the types of empirical kernel maps in Eq. 6.23 and Eq. 6.24:

$$K(\mathbf{x}_t, \mathbf{x}_{s,j}) = \mathbb{K}\left(\overrightarrow{S}_{\text{LR}}(\mathbf{x}_t, \mathcal{X}_s), \overrightarrow{S}_{\text{LR}}(\mathbf{x}_{s,j}, \mathcal{X}_s)\right) \tag{6.26}$$

and

$$K(\mathbf{x}_t, \mathbf{x}_{s,j}) = \mathbb{K}\left(\overrightarrow{S}_{\text{LR}}(\mathbf{x}_t, \mathcal{X}), \overrightarrow{S}_{\text{LR}}(\mathbf{x}_{s,j}, \mathcal{X})\right) \tag{6.27}$$

where $\mathbb{K}(\cdot, \cdot)$ is a standard SVM kernel, e.g., linear or RBF. Only RBF kernel $\mathbb{K}(\mathbf{x}, \mathbf{y}) = \exp(-\frac{\|\mathbf{x}-\mathbf{y}\|^2}{2\gamma^2})$ was adopted in this thesis. $K(\mathbf{x}_t, \mathbf{x}_{b,i})$ can be obtained by replacing $\mathbf{x}_{s,j}$ in Eq. 6.26 and Eq. 6.27 with $\mathbf{x}_{b,i}$.

While our earlier studies [23, 24] have demonstrated that PLDA-SVM scoring (Eq. 6.25) performs better than simple PLDA-LR scoring (Eq. 6.12), the SVMs in Eq. 6.25 still has some limitations [93]. First, although SVM is a sparse model, the number of support vectors increases linearly with the size of the training set. In our case, this property limits the value of B (Eq. 6.22) for training the SVMs. Second, the SVM scores in Eq. 6.25 are not probabilistic, meaning that score normalization may be needed to adjust the score range of individual SVMs. Third, to achieve the best performance, it is necessary to tradeoff the training error and the margin of separation through adjusting the penalty factor for each target speaker during SVM training. Given the limited number of enrollment utterances for some speakers, this is not easy to achieve. As a result, [23,24] used the same penalty factor for all target speakers. In our experiments, the penalty factor C was set to 1 for all SVMs.

6.4 Empirical Kernel Maps for PLDA-RVM Scoring

Adopting PLDA-SVM scoring with empirical kernel maps not only utilizes the multiple enrollment utterances of target speakers, but also opens up opportunity for adopting sparse kernel machines in PLDA-based speaker verification systems. Accordingly, this thesis proposes incorporating the empirical kernel maps into a sparse kernel machine known as the relevance vector machine [93]. The main difference between SVM and RVM lies in the learning methods. The former is based on structural risk minimization, whereas the latter is based on a fully probabilistic framework. RVMs do not suffer from the limitations of SVMs (cf. Section 6.3.2), but their performance is comparable to that of the SVMs.

6.4.1 Relevance Vector Machine

Assume that we are given N training vectors $\{\mathbf{x}_1, ..., \mathbf{x}_N\}$ with labels $y_n \in \{+1, -1\}, n = 1, ..., N$ and a test vector $\mathbf{x}_t$, relevance vector machine (RVM) is a Bayesian treatment of Eq. 6.28 below:

$$f(\mathbf{x}_t; \mathbf{w}) = \sum_{i=1}^{N} w_i K(\mathbf{x}_t, \mathbf{x}_i) + w_0 \tag{6.28}$$

where $\mathbf{w} = [w_0, ..., w_N]$ are the weights determined by minimizing the error on the training set while maximizing the margin between the two classes, w_0 is a bias term, and $K(\mathbf{x}_t, \mathbf{x}_i)$ is a kernel function.

When an RVM is applied to regression, the target y is assumed to follow a Gaussian distribution with mean $f(\mathbf{x};\mathbf{w})$ and variance σ^2; when it is applied to classification, the target conditional distribution $p(y|\mathbf{x})$ is assumed to follow a Bernoulli distribution. This thesis focuses on the regression mode of RVMs because it performs significantly better than the classification mode in NIST SRE (see Chapter 8).

Assume that for target speaker s, we have a set of training i-vectors $\mathcal{X} = \{\mathcal{X}_s, \mathcal{X}_b\}$ as in Eq. 6.21 and Eq. 6.22 and that $y_i = 1$ when $\mathbf{x}_i \in \mathcal{X}_s$ and $y_i = -1$ when $\mathbf{x}_i \in \mathcal{X}_b$. Assume also that y_i's ($i = 1, ..., N$) are independent, the likelihood of the training data set can be written as [93]:

$$p(\mathbf{y}|\mathbf{w},\sigma^2) = (2\pi\sigma^2)^{-\frac{N}{2}} \exp\left\{-\frac{1}{2\sigma^2}\|\mathbf{y}-\mathbf{\Phi}\mathbf{w}\|^2\right\} \tag{6.29}$$

where

$$\begin{aligned} &N = |\mathcal{X}_s| + |\mathcal{X}_b|;\ \mathbf{y} = [y_1, ..., y_N]^{\mathsf{T}} \\ &\mathbf{w} = [w_0, ..., w_N]^{\mathsf{T}};\ \mathbf{\Phi} = [\boldsymbol{\phi}(\mathbf{x}_1), \boldsymbol{\phi}(\mathbf{x}_2), ..., \boldsymbol{\phi}(\mathbf{x}_N)]^{\mathsf{T}} \\ &\boldsymbol{\phi}(\mathbf{x}_i) = [1, K(\mathbf{x}_i,\mathbf{x}_1), K(\mathbf{x}_i,\mathbf{x}_2), ..., K(\mathbf{x}_i,\mathbf{x}_N)]^{\mathsf{T}} \end{aligned} \tag{6.30}$$

and σ^2 is the variance of the additive Gaussian noise ϵ in the model $y = f(\mathbf{x};\mathbf{w}) + \epsilon$. To avoid over-fitting, RVM defines a zero-mean Gaussian prior distribution over $\mathbf{w}$:

$$p(\mathbf{w}|\boldsymbol{\alpha}) = \prod_{i=0}^{N} \mathcal{N}(w_i|0, \alpha_i^{-1}) \tag{6.31}$$

where $\boldsymbol{\alpha} = [\alpha_0, \alpha_1, ..., \alpha_N]^{\mathsf{T}}$ and α_i is the hyperparameter associated with weight w_i. By considering $\mathbf{w}$ probabilistic and using the notion of conditional independence [16], the predictive distribution of y_t given a test vector $\mathbf{x}_t$ is

$$p(y_t|\mathbf{y}) = \int p(y_t|\mathbf{w}, \boldsymbol{\alpha}, \sigma^2) p(\mathbf{w}, \boldsymbol{\alpha}, \sigma^2|\mathbf{y}) d\mathbf{w} d\boldsymbol{\alpha} d\sigma^2 \tag{6.32}$$

where

$$p(y_t|\mathbf{w}, \boldsymbol{\alpha}, \sigma^2) = p(y_t|\mathbf{w}, \sigma^2) \tag{6.33}$$

$$p(\mathbf{w}, \boldsymbol{\alpha}, \sigma^2|\mathbf{y}) = p(\mathbf{w}|\mathbf{y}, \boldsymbol{\alpha}, \sigma^2) p(\boldsymbol{\alpha}, \sigma^2|\mathbf{y}). \tag{6.34}$$

After some derivations [16, 93], we can obtain the posterior distribution over the weights as follows:

$$p(\mathbf{w}|\mathbf{y}, \boldsymbol{\alpha}, \sigma^2) = \mathcal{N}(\mathbf{w}|\boldsymbol{\mu}, \boldsymbol{\Sigma}) \tag{6.35}$$

where

$$\begin{gathered} \boldsymbol{\mu} = \sigma^{-2} \boldsymbol{\Sigma} \boldsymbol{\Phi}^\top \mathbf{y} \\ \boldsymbol{\Sigma} = (\sigma^{-2} \boldsymbol{\Phi}^\top \boldsymbol{\Phi} + \mathbf{A})^{-1};\ \mathbf{A} = \mathrm{diag}(\alpha_0, \alpha_1, ..., \alpha_N). \end{gathered} \tag{6.36}$$

Instead of computing the posterior $p(\boldsymbol{\alpha}, \sigma^2|\mathbf{y})$ in Eq. 6.33, [93] uses a delta function at the most probable values of $\boldsymbol{\alpha}$ and σ^2 as an approximation. Therefore, using Eq. 6.34 and assuming uniform priors for $\boldsymbol{\alpha}$ and σ^2, Eq. 6.32 reduces to

$$p(y_t|\mathbf{y}) = \int p(y_t|\mathbf{w}, \boldsymbol{\alpha}_{\mathrm{MP}}, \sigma^2_{\mathrm{MP}}) p(\mathbf{w}|\mathbf{y}, \boldsymbol{\alpha}_{\mathrm{MP}}, \sigma^2_{\mathrm{MP}}) d\mathbf{w} \tag{6.37}$$

where

$$\begin{aligned} (\boldsymbol{\alpha}_{\mathrm{MP}}, \sigma^2_{\mathrm{MP}}) &= \arg\max_{\boldsymbol{\alpha}, \sigma^2} p(\boldsymbol{\alpha}, \sigma^2|\mathbf{y}) \\ &= \arg\max_{\boldsymbol{\alpha}, \sigma^2} \int p(\mathbf{y}|\mathbf{w}, \sigma^2) p(\mathbf{w}|\boldsymbol{\alpha}) d\mathbf{w}. \end{aligned} \tag{6.38}$$

Because both terms in the integrand of Eq. 6.37 are Gaussians, the predictive distribution is also a Gaussian:

$$p(y_t|\mathbf{y}, \boldsymbol{\alpha}_{\text{MP}}, \sigma^2_{\text{MP}}) = \mathcal{N}(y_t|g(\mathbf{x}_t), \sigma^2_t) \tag{6.39}$$

with

$$g(\mathbf{x}_t) = \boldsymbol{\mu}^{\mathsf{T}}\boldsymbol{\phi}(\mathbf{x}_t) \tag{6.40}$$

$$\sigma^2_t = \sigma^2_{\text{MP}} + \boldsymbol{\phi}(\mathbf{x}_t)^{\mathsf{T}}\boldsymbol{\Sigma}\boldsymbol{\phi}(\mathbf{x}_t). \tag{6.41}$$

The two optimized hyperparameters $\boldsymbol{\alpha}_{\text{MP}}$ and σ^2_{MP} can be obtained by maximum likelihood. Readers may refer to Section 2.3 in [93] for the details of the optimization. During the optimization, many of the hyperparameters α_i tend to infinity and the corresponding weights w_i become zero; the vectors $\mathbf{x}_i$ corresponding to the non-zero weights are considered as **relevance vectors**.

6.4.2 PLDA-RVM

In our case, we used $g(\mathbf{x}_t)$ in Eq. 6.40 as the output of RVM regression. After incorporating the empirical kernel maps (Eq. 6.26 and Eq. 6.27), the score of RVM regression can be written as

$$S_{\text{RVM}}(\mathbf{x}_t, \mathcal{X}_s, \mathcal{X}_b) = g(\mathbf{x}_t) = \boldsymbol{\mu}^T\boldsymbol{\phi}(\mathbf{x}_t, \mathcal{X}_s, \mathcal{X}_b) \tag{6.42}$$

where

$$\boldsymbol{\phi}(\mathbf{x}_t, \mathcal{X}_s, \mathcal{X}_b) = [1, K(\mathbf{x}_t, \mathbf{x}_{s,1}), ..., K(\mathbf{x}_t, \mathbf{x}_{s,H_s}), K(\mathbf{x}_t, \mathbf{x}_{b,1}), ..., K(\mathbf{x}_t, \mathbf{x}_{b,B})]^{\mathsf{T}} \tag{6.43}$$

6.5 *Known Non-targets for PLDA-SVM/PLDA-RVM Scoring*

In previous SREs, for each verification trial, only the knowledge of the target under test can be used for computing the score. This restriction, however, has been removed in NIST 2012 SRE. The permission to use other target speakers (considered as known non-target speakers) leads to the compound likelihood ratio [109–111] and anti-models [112], which improves verification performance substantially. Unlike the compound likelihood ratio, this thesis exploits the information of the known non-targets to improve the discrimination power of PLDA-SVMs/PLDA-RVMs.

Specifically, instead of injecting the likelihood-ratio scores of known non-targets into the posterior probability computation as in [109,111], this thesis uses the PLDA scores arising from any i-vectors with respect to the target-speaker's i-vectors and a group of background speakers' i-vectors to define a speaker-dependent empirical score space. Then, for each target speaker, an SVM/RVM is trained by pooling the score vectors produced by all of the known and unknown non-targets.

Assume that M target speakers have been enrolled in a system. When training the SVM of a target speaker, the remaining $(M-1)$ competing known non-target speakers from the target-speaker set are considered as the new background training set. Specifically, the speaker-class and impostor-class i-vectors for training the SVM of target speaker s are

$$\mathcal{X}_s = \{\mathbf{x}_{s,1}, \ldots, \mathbf{x}_{s,H_s}\} \text{ and } \mathcal{X}_{s,a} = \{\mathbf{x}_{a,1}, \ldots, \mathbf{x}_{a,M-1}\}, \tag{6.44}$$

respectively, where $\mathcal{X}_{s,a}$ contains the i-vectors of the competing known non-target speakers with respect to s. As a result, the SVM and RVM regression score of a test i-vector $\mathbf{x}_t$ is

$$S'_{\mathrm{SVM}}(\mathbf{x}_t, \mathcal{X}_s, \mathcal{X}_{s,a}) = \sum_{j \in \mathrm{SV}_s} \alpha_{s,j} K(\mathbf{x}_t, \mathbf{x}_{s,j}) \; - \sum_{i \in \mathrm{SV}_a} \alpha_{a,i} K(\mathbf{x}_t, \mathbf{x}_{a,i}) + d'_s \tag{6.45}$$

where $K(\cdot, \cdot)$ is a kernel function (Eq. 6.26 or Eq. 6.27), SV_s and SV_a contain the indexes of the support vectors corresponding to the speaker class and impostor class, respectively, $\alpha_{s,j}$ and $\alpha_{a,i}$ are the Lagrange multipliers of the SVM, and d'_s is a speaker-dependent bias.

$$S'_{\mathrm{RVM}}(\mathbf{x}_t, \mathcal{X}_s, \mathcal{X}_{s,a}) = g(\mathbf{x}_t) = \boldsymbol{\mu}^T \boldsymbol{\phi}(\mathbf{x}_t, \mathcal{X}_s, \mathcal{X}_{s,a}) \tag{6.46}$$

where

$$\boldsymbol{\phi}(\mathbf{x}_t, \mathcal{X}_s, \mathcal{X}_{s,a}) = [1, K(\mathbf{x}_t, \mathbf{x}_{s,1}), ..., K(\mathbf{x}_t, \mathbf{x}_{s,H_s}), K(\mathbf{x}_t, \mathbf{x}_{a,1}), ..., K(\mathbf{x}_t, \mathbf{x}_{a,M-1})]^{\mathsf{T}} \tag{6.47}$$

Chapter 7

EXPERIMENTAL SETUP

This chapter describes the experimental setups. They include the selections of evaluation data and development data for the experiments, the details of acoustic front-end processing, speaker modeling, session and channel variability modeling and scoring methods. The experiments involve three systems: (1) GMM-supervector based system (introduced in Chapter 3), (2) i-vector based system (introduced in Chapter 4), and (3) Gaussian-PLDA with sparse kernel machines (introduced in Chapter 6).

7.1 Speech Data

In the last two decades, a series of text-independent speaker recognition evaluations (SRE) that supply speech data and define the evaluation rules have been organized by US National Institute of Standards and Technology (NIST). For each SRE, there are some new challenges for researchers on the basis of the results and expectations from the previous SREs [113]. The general configuration of NIST SRE is to use the speech data of previous NIST SREs or those unrelated to the evaluation as development data and compute the scores according to the verification trials using the given enrollment (training) and test segments. The benefits of NIST SREs are simple to use, provision of standards for researchers to compare their methods, and relevance to practical applications of speaker recognition technologies.

The experiments in this thesis use four NIST SREs[1] for performance evaluation. They are NIST'02 [114], NIST'04 [115], NIST'10 [116] and NIST'12 [7]:[2]

- **NIST'02** contains cellular phone conversations of 139 male and 191 female target speakers taken from Switchboard Cellular Part 2. Each target speaker provides 2 minutes of speech for training. There are 2,983 true-speaker trials and 36,287 impostor attempts.

- **NIST'04** contains 28 evaluation conditions. This thesis focuses on the 1side-1side condition, i.e., each of the training and test segments contains a whole conversation side. This condition contains 246 male and 376 female target speakers, each providing 5 minutes of speech (including silence) for training. The evaluation condition also contains 2,386 true-speaker trials and 23,838 impostor attempts, with each test segment containing 5 minutes of speech (including silence).

- **NIST'10** contains 9 common evaluation conditions. The major differences between NIST'10 and NIST'02 and NIST'04 include the following: (1)interview and microphone speeches are included in NIST'10, (2) some vocal effort conversational telephone speeches are used as test segment, (3) the duration of interview speeches in the common evaluation is varied from 3 to 15 minutes, (4) two common evaluation conditions are cross-channel conditions, and (5) some conversational telephone speeches are recorded through the microphone, which are used as enrollment speeches.

[1]http://www.itl.nist.gov/iad/mig/tests/sre

[2]Hereafter, all NIST SREs are abbreviated as NIST'XX, where XX stands for the year of evaluation.

Common Condition		CC1	CC2	CC4	CC7	CC9	Mic
Speech Type	Train	int-mic	int-mic	int-mic	phn-mic	phn-mic	int-mic & phn-mic
	Test	int-mic	int-mic	phn-mic	phn-mic	phn-mic	int-mic & phn-mic
No. of target trials		989	3,463	1,225	179	117	5,973
No. of impostor trials		28,114	98,282	39,166	12,786	10,697	189,045

Table 7.1: The number of male target-speaker trials and impostor trials and the speech types for training and testing under the common conditions that involve microphone recordings in the core set of NIST 2010 SRE. *Mic*: Combining the trials of all common conditions that involve microphone recordings. *phn-mic*: Telephone conversation recorded by microphones. *int-mic*: Interview sessions recorded by microphones.

Common Condition		CC1	CC2	CC4	CC7	CC9	Mic
Speech Type	Train	int-mic	int-mic	int-mic	phn-mic	phn-mic	int-mic & phn-mic
	Test	int-mic	int-mic	phn-mic	phn-mic	phn-mic	int-mic & phn-mic
No. of target trials		1,978	6,932	1,886	179	117	11,092
No. of impostor trials		346,857	1,215,586	364,308	39,898	29,667	1,996,316

Table 7.2: The number of male target-speaker trials and impostor trials and the speech types for training and testing under the common conditions that involve microphone recordings in the extended core set of NIST 2010 SRE. *Mic*: Combining the trials of all common conditions that involve microphone recordings. *phn-mic*: Telephone conversation recorded by microphones. *int-mic*: Interview sessions recorded by microphones.

This thesis mainly focuses on male interview and microphone speech of the *core set* and *extended core set*, i.e., Common Conditions 1, 2, 4, 7 and 9. In the sequel, we use "CC" to denote common evaluation conditions. Table 7.1 and Table 7.2 show the number of trials and speech types for training and testing under these common conditions. In CC1, the same microphone was used for recording both training and test segments, whereas in CC2 different

Common Condition	CC2	CC4	CC5
No. of target trials	2,830	2,775	1,534
No. of impostor trials	161,719	122,625	61,311

Table 7.3: The number of male target-speaker trials and impostor trials under the common conditions that involve telephone recordings in the core set of NIST 2012 SRE.

microphones were used. In CC7, high vocal-effort speech segments were used for testing, whereas in CC9 low vocal-effort segments were used.

- **NIST'12** contains 5 common evaluation conditions. Comparing with the previous NIST SREs, NIST'12 introduces some new challenges and some new protocols [97, 113]: (1) additive and environmental noises are included in the test segments of several common conditions, (2) the duration of test segments is varied from 30 seconds to 300 seconds, (3) some target speakers in NIST'12 have multiple enrollment utterances, whereas the target speakers in previous NIST SREs only contain one enrollment utterance, and (4) the restriction that only the knowledge of the target speaker under test can be used for computing the score for each verification trial in previous NIST SREs is removed.

  The *core set* of NIST'12 was used for performance evaluation. This thesis focuses on the male telephone speech of the core task, i.e., Common Evaluation Conditions 2, 4, and 5. Table 7.3 shows the number of trials under these common conditions. In the evaluation dataset, no noise was added to the test segments of CC2, whereas noise was added to the test segments of CC4 and test segments in CC5 were collected in a noisy environment. All of these conditions contain training segments with variable length and variable numbers

	NIST'02 Eval	NIST'04 Eval	NIST'10 Eval	NIST'12 Eval
UBMs	1,006 male and 1,344 female utterances from NIST'01	1,100 male and 1,640 female utterances from Fisher	4,072 male microphone utterances from NIST'05–NIST'08	3,500 male telephone utterances and 3,501 male microphone utterances from NIST'05–NIST'08
Total Variability Matrix	——	——	9,511 microphone utterances from 191 male speakers in NIST'05–NIST'08	6,380 male telephone utterances and 8,495 male microphone utterances from NIST'06–NIST'10
Background Speakers	112 male and 122 female speakers from training sessions of NIST'01	300 male and 300 female speakers from Fisher	300 male microphone utterances from NIST'05–NIST'06 for System 1; 633 male microphone utterances from NIST'05–NIST'08 for System 2	704 male microphone utterances from NIST'05–NIST'08
NAP	74 male and 100 female speakers from test sessions# of NIST'01	236 male and 266 female speakers from test sessions# of NIST'99 and NIST'00	143 male speakers from NIST'05–NIST'08#	——
LDA and WCCN	——	——	9,511 microphone utterances from 191 speakers in NIST'05–NIST'08	15,662 original utterances from 673 male speakers from NIST'06–NIST'10
PLDA	——	——	9,511 microphone utterances from 191 speakers in NIST'05–NIST'08	15,662 original utterances from 673 male speakers from NIST'06–NIST'10
T-norm Models	127 male and 145 female speakers from test sessions of NIST'01	200 male and 200 female speakers from Fisher	300 male microphone utterances from NIST'05 for System 1; 288 male microphone utterances from NIST'05–NIST'06 for System 2	——
Z-norm Models	——	——	300 male microphone utterances from NIST'05–NIST'06 for System 1; 288 male microphone utterances from NIST'05–NIST'06 for System 2	180 male telephone utterances from NIST'08

Table 7.4: The roles played by different corpora in the performance evaluations. #Only speakers with 8 or more utterances were used for estimating the NAP matrices. "System 1" represents GMM-supervector based speaker verification system (Chapter 3). "System 2" represents i-vector based speaker verification system (Chapter 4).

of training segments per target speaker. We removed the 10-second utterances and the summed-channel utterances from the training segments of NIST'12 but ensured that all target speakers have at least one utterance for training.

Table 7.4 summarizes the roles played by these corpora in the evaluations. NIST'01 contains 2,350 cellular phone conversations extracted from the Switchboard-II Phase IV Corpus. All of these utterances were used for training the gender-dependent background models in NIST'02 evaluation. All of the utterances from the training sessions in the corpus were used as gender-dependent impostor data for training the target-speaker SVMs. Test utterances with length (after silence removal) longer than 25 seconds were used for creating the T-norm [30] speaker models, which amount to 127 male and 145 female speaker models for T-norm. The corpus was also used for computing the projection matrices in Nuisance Attribute Projection (NAP) [1]. Specifically, speakers with multiple conversations were identified and the conversations of these speakers are assumed to be extracted from different sessions. This amounts to 74 male speakers and 100 female speakers, each providing 12 conversations on average. The number of nuisance dimensions (corank in [48]) to be projected out is eight for male and one for female. These numbers were found empirically to produce the best performance on a baseline system (see Section 5.2 in [15]).

For the NIST'04 evaluation, the Fisher corpus was used for training the gender-dependent UBMs. A subset of speakers was used for training the gender-dependent T-norm models, and another subset was used as impostor-class data for training the target-speaker SVMs and T-norm SVMs. Finally, 236 male and 266 female speakers from NIST'99 and NIST00 were used for estimating the gender-dependent NAP matrices. Each of these speakers has at least 8 utterances. The NAP corank was set to 64 for both genders.

NIST'10 evaluation has been used for three systems. Therefore, the selections of development data are different for different systems. For GMM-supervector based speaker verification systems, we used 300 male microphone utterances from NIST'05–NIST'06 whereas 633 male microphone utterances from NIST'05–NIST'08 were selected for i-vector based speaker verification system. Moreover, 300 T-norm models were created from the microphone utterances in NIST'05 and 300 Z-norm utterances were selected from NIST'05–NIST'06 in GMM-supervector based systems. For i-vector based systems, 288 T-norm utterances and 288 Z-norm utterances were selected from the microphone speech in NIST'05–NIST'08.

In NIST'12 evaluation, noise was added in the test segments. To improve noise robustness, we followed the suggestions in [110] to add noise to the training files. To this end, we constructed a noise dataset comprising 13 real crowd noise files and 17 heating, ventilation, and air conditioning (HVAC) noise files from [117] and 10 artificial crowd noise files generated by summing 441 utterances from male and female speakers in pre-2012 NIST SRE. For each training file with SNR above 15dB, we generated two noisy speech files at an SNR of 6dB and 15dB by randomly selecting two noise files from the noise dataset. For each training file with SNR between 6dB and 15dB, we produced a noisy speech file at 6dB.

7.2 Acoustic Front-End Processing

An in-house VAD [92, 118] was applied to detect the speech regions of each utterance. Briefly, for each conversation side, the VAD uses spectral subtraction with a large over-subtraction factor to remove the background noise. The low energy and high energy regions of the noise-removed speech were used for estimating a decision threshold. This energy-based threshold was then applied to the whole utterance to detect the speech regions.

For GMM-supervector based speaker verification system, 12 MFCCs plus their first derivative were extracted from the speech regions of the utterance, leading to 24-dim acoustic vectors. For i-vector/PLDA system, 19 MFCCs together with energy plus their 1st- and 2nd-derivatives were extracted from the speech regions of each utterance, leading to 60-dim acoustic vectors. Cepstral mean normalization [119] was applied to the MFCCs, followed by feature warping [120].

7.3 GMM-supervector Based Speaker Verification

The experiments of GMM-supervector based speaker verification were performed in NIST'02, NIST'04, and NIST'10 evaluation datasets. The common evaluation conditions of these SREs used in the experiments are introduced in Section 7.1. Table 7.4 shows the details of development data (NAP, UBM, background speakers for SVM, Z-norm, and T-norm) for the experiments.

7.3.1 Speaker Models

In the NIST'02 and NIST'04 evaluation, for the GMM–UBM systems, the number of mixtures for gender-dependent UBMs is 1,024. The GMMs of target speakers were adapted from the UBMs using MAP adaptation [20] with relevance factor r in Eq. 2.7 set to 16. For the GMM–SVM systems, the number of mixtures was set to 256. Each supervector in the GMM–SVM comprises the means of a MAP-adapted GMM.

In NIST'10 evaluation, the GMM-SVM and speaker comparison systems use three different kernels, including the KL divergence kernel K_{KL}, GMM-UBM mean interval kernel K_{GUMI}, and the geometric-mean-comparison kernel K_{GM}. The GMM-supervectors (speaker models) for these kernels were adapted from a 512-Gaussian UBM created from a subset (totally 5,077 utterances) of microphone speech in

NIST'05 and NIST'06. Different MAP adaptation parameters were used to create the supervectors for different kernels. Specifically, for the KL kernel, only the means were adapted, using a relevance factor of 16. For the GUMI kernel, a relevance factor of 16 was used to adapt the means and variances. For the geometric-mean-comparison kernel, the relevance factors for the means and mixture weights were set to 0.01 and zero, respectively, meaning that the mixture weights were estimated using maximum-likelihood principle with mixture indexes aligned with those of the UBM. We followed the setting of the relevance factors as in [46]. The reason of using such a small relevance factors will be explained in Chapter 8.

For each target-speaker SVM, positive (target-speaker) class supervectors were obtained by stacking the means of the MAP-adapted GMMs created from the utterance of the corresponding speaker, whereas the negative (impostor) class supervectors were obtained from the NIST'01, Fisher, or NIST'05–06 (see Table 7.4).

7.3.2 Session and Channel Variability Modelling

For the GMM-SVM and speaker comparison systems, we applied NAP [1] to all GMM-supervectors for channel compensation. Table 7.4 list the databases used for training the NAP projection matrices.

For the JFA system, due to insufficient microphone data, we used both telephone and microphone utterances to estimate two channel-dependent eigenvoice matrices and two eigenchannel matrices. Then, these matrices were concatenated to produce the eigenvoice and eigenchannel matrices for session and channel variability modelling. Here, the term "channel" refers to either the telephone channel (speech recorded by telephone handsets) or microphone channel (speech recorded by microphones).

Specifically, for the eigenvoice matrices, we used 5,616 telephone utterances from NIST'04–06 to train a 1024-Gaussian telephone-channel UBM. Then, we used 1,194 speakers from NIST'04–08, Switchboard II Phase 2 and Phase 3, and Switchboard Cellular Parts 2 to estimate a telephone eigenvoice matrix with a speaker factor of 230. Each of these speakers has at least 8 utterances, amounting to a total of 14,562 telephone utterances. This procedure produces a 61400×230 eigenvoice matrix. For the microphone channel, we selected 4,072 utterances from NIST'05–08 to train a 1024-Gaussian microphone-channel UBM, using the telephone-channel UBM as the seed for the EM algorithm. This ensures a meaningful correspondence between the indexes of telephone-channel and microphone-channel GMM-supervectors. Then, we selected 4,072 utterances from 144 speakers (each with at least 5 utterances) in NIST'05–08 to estimate a 120-factor eigenvoice matrix. Then, the telephone and microphone eigenvoice matrices were concatenated to produce a 61400×350 eigenvoice matrix.

To estimate an eigenchannel matrix with 100 channel factors for the telephone channel, we selected 8,795 telephone utterances from 806 speakers in NIST04–08 and Switchboard Cellular Parts 2. We used the same set of data that used for estimating the microphone eigenvoice matrix to estimate a 50-factor microphone eigenchannel matrix. The two channel-dependent eigenchannel matrices were then concatenated to form a 61400×150 eigenchannel matrix. The concatenated eigenvoice and eigenchannel matrices were then used for estimating the speaker residual matrix. A modified version of the BUT JFA Matlab Demo[3] was used for JFA training and scoring.

[3]http://speech.fit.vutbr.cz/en/software/joint-factor-analysis-matlab-demo

7.3.3 Score Normalization

For GMM–UBM, GMM–SVM and speaker comparison, T-norm or ZT-norm (Z-norm followed by T-norm) [30] was applied during the scoring stage. For JFA, ZT-norm was applied to the likelihood ratio score.

7.4 I-Vector Based Speaker Verification

7.4.1 Total Variability Modeling and Channel Compensation

The i-vector systems are based on a gender-dependent UBM with 1024 mixtures. 9,511 utterances from NIST'05–NIST'08 were selected for estimating a total variability matrix with 400 total factors. JFA Matlab code from Brno University of Technology (BUT) [121] was modified for i-vector training and scoring. Before calculating the verification scores, LDA and WCCN projections were performed for channel compensation. We selected 6,102 utterances from 191 speakers in NIST'05–NIST'08 to estimate the LDA and WCCN matrices. After LDA and WCCN projections, the dimension of i-vectors was reduced to 150.

7.4.2 Scoring Method and Score Normalization

In GMM-supervector based systems, we adopted two scoring method: SVM scoring and cosine distance scoring. For building the SVM classifiers, we selected 633 impostors from NIST'05–08 SREs. ZT-norm [30] was used for score normalization. 288 T-norm utterances and 288 Z-norm utterances (each from a different set of speakers) were selected from the interview and microphone speech in NIST'05–08 SREs.

7.4.3 Utterance Length after Utterance partitioning

Figure 5.4 and Figure 5.5 demonstrate that the discriminative power of i-vector increases most rapidly from 0.5 to 1.0 minute and becomes saturated after 2 minutes. This information gives us some guidelines on how to partition an utterance. Therefore, except for the experiments investigating the effect of the number of partitions, we partitioned all utterances into four segments. This amounts to sub-utterances length of 0.75, 1.25, and 2 minutes for the 3-min, 5-min, and 8-min utterances. While the sub-utterance length for 3-min utterances is relatively short, partitioning the 3-min utterances into 4 segments gives us more i-vectors for training the LDA+WCCN matrices. The results suggest that this is a reasonable compromise between the number of i-vectors and their discriminative power.

7.5 Gaussian-PLDA with Sparse Kernel Machines

7.5.1 Enrollment Utterances and Non-Target Speaker Utterances

Because the test conditions involve phonecall speech only, only telephone utterances were selected as enrollment utterances for matching the channel between enrollment and test sessions. Although many target speakers in NIST'12 have multiple training segments, some of them have a few training segments only. More precisely, after removing the 10-second segments and summed-channel segments, 50 out of 723 target speakers have one long training segment only. Therefore, UP-AVR ($N = 4$ and $R = 4$) was applied to construct empirical kernel maps and to alleviate the effect of data imbalance in training the SVMs.

Reference [95] suggests that it is more preferable to use the same number of less dependent enrollment utterances per target speaker. We adopted this suggestion and configured UP-AVR to produce H_s i-vectors for speaker s using the following rule:

$$H_s = \begin{cases} 17, & |\mathcal{U}_s| < 17 \\ |\mathcal{U}_s|, & |\mathcal{U}_s| \geq 17 \end{cases} \tag{7.1}$$

where $\mathcal{U}_s$ is the set of full-length enrollment utterances and $|\mathcal{U}_s|$ represents its cardinality.

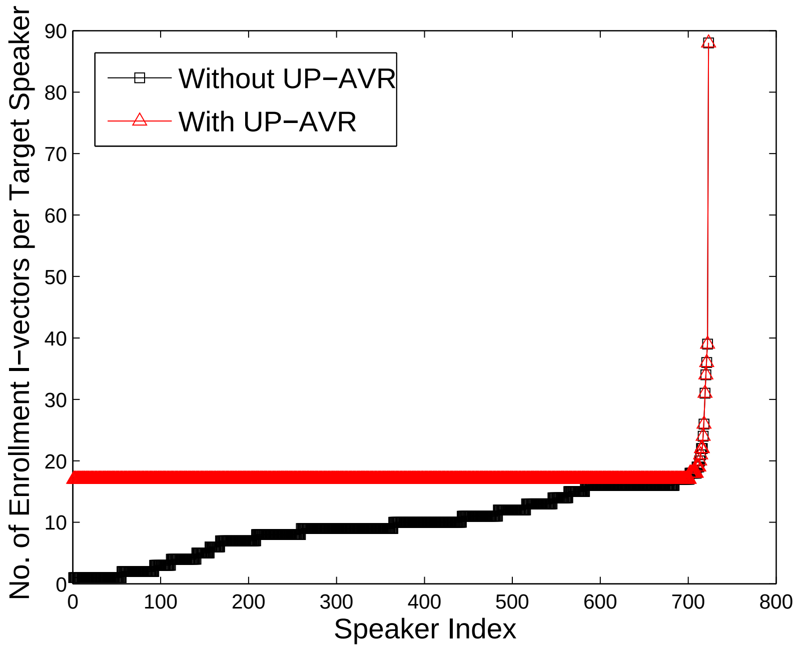


Figure 7.1: Numbers of enrollment i-vectors per target speaker with and without performing UP-AVR. Each speaker index represents one target speaker.

Evidently, the black curve in Figure 7.1 suggests that without UP-AVR, many target speakers only have a few enrollment i-vectors, because of the small number of enrollment utterances. However, the UP-AVR can produce a fixed number of enrollment i-vectors (the red curve) for these speakers.

Non-target speakers can be divided into unknown non-targets and known non-targets. 704 unknown non-target speakers were selected for the experiments. Each non-target speaker provides one utterance. Therefore, $B = 704$ utterances were used to train the speaker-dependent SVMs for all common conditions. Because all the common conditions investigated in this study involve 723 target speakers, each target speaker has 722 competing known non-targets to train his SVM.

7.5.2 Total Variability Modeling and PLDA

The i-vector systems are based on a gender-dependent UBM with 1024 mixtures. 3,500 microphone utterances and 3,501 telephone utterances from NIST'05–NIST'08 were used for training the UBM. We selected 14,875 telephone and interview conversations from 575 speakers in NIST'06–NIST'10 to estimate a total variability matrix with 400 total factors. We applied whitening [17] and i-vector length normalization [13] to the 400-dimensional i-vectors. Then, we performed LDA [16] and WCCN [17] on the resulting vectors to reduce the dimension to 200 before training the PLDA models with 150 latent variables.

According to [122], adding noise to the training files of UBM and total variability modeling receives insignificant performance gain. Hence, we followed the steps in [122] and only added noise to the training files of LDA and PLDA models.

Chapter 8

RESULTS AND DISCUSSION

This chapter elaborates the experimental results and related analyses of the proposed methods in three different speaker verification systems introduced in previous chapters, including GMM-supervector, i-vector, and PLDA based systems.

8.1 GMM-supervector Based Speaker Verification

8.1.1 Statistical Properties of GMM-Supervectors

The feature dimension of GMM-supervectors is MD, where M is the number of mixtures in a GMM and D is the dimension of acoustic vectors. For systems that use a large number of mixtures (e.g., $M \geq 1024$), the dimension of the GMM-supervectors will become very large, which introduces excessive computational burden on the training of speaker-dependent SVMs. If M is too small, the resulting supervectors may not be able to represent the characteristics of the target speakers. Nevertheless, one may ask: Among all the features in the supervectors, how many of them are relevant or useful for recognizing speakers?

To answer this question, we computed the variances of MD-dimensional features from 50 normalized GMM-supervectors $\boldsymbol{\Omega}^{-\frac{1}{2}}\vec{\boldsymbol{\mu}}^{(b_k)}$, where M ranges from 64 to 1,024, $D = 24$, $k = 1, \ldots, 50$, and $\boldsymbol{\Omega}$ and $\vec{\boldsymbol{\mu}}$ are defined in Eq. 3.4. Features were then sorted in descending order of variances. Figure 8.1 shows the variances of features against the feature indexes. The horizontal line is a threshold (0.05) below which

Performance	No. of Centers in GSV				
	64	128	256	512	1024
EER (%)	10.85	9.84	**9.05**	9.98	11.78
Minimum DCF	0.0414	0.0372	**0.0362**	0.0367	0.0428

Table 8.1: The effect of varying the number of Gaussian components in GMM-supervectors on the verification performance of the GMM–SVM baseline system (with NAP and T-norm) in NIST'02.

the features are considered to have no significant contribution to the classification task. Evidently, when $M \geq 512$, a large percentage of features have variances below the threshold, which means that the resulting SVMs have input dimension larger than necessary. A model size of 256 mixtures seems to be a good compromise. This observation also agrees with the verification performance shown in Table 8.1, where the best speaker verification performance (in terms of EER and Minimum DCF) is obtained when $M = 256$. Based on this finding, we set $M = 256$ for the rest of the experiments under the NIST'02 and NIST'04 evaluations.

8.1.2 GMM–UBM vs. GMM–SVM

Table 8.2 and Table 8.3 compares the performance GMM–UBM and GMM–SVM. The results clearly demonstrate the merit of GMM–SVM and confirm the claims in Chapter 3.3.

8.1.3 Comparing Kernels in GMM–SVM

Table 8.4 shows the performance of different GMM–SVM systems under different common conditions in NIST'10. Systems A to C allow us to compare the performance of the KL kernel, GUMI kernel and K_{GM} kernel in GMM-SVM. The re-

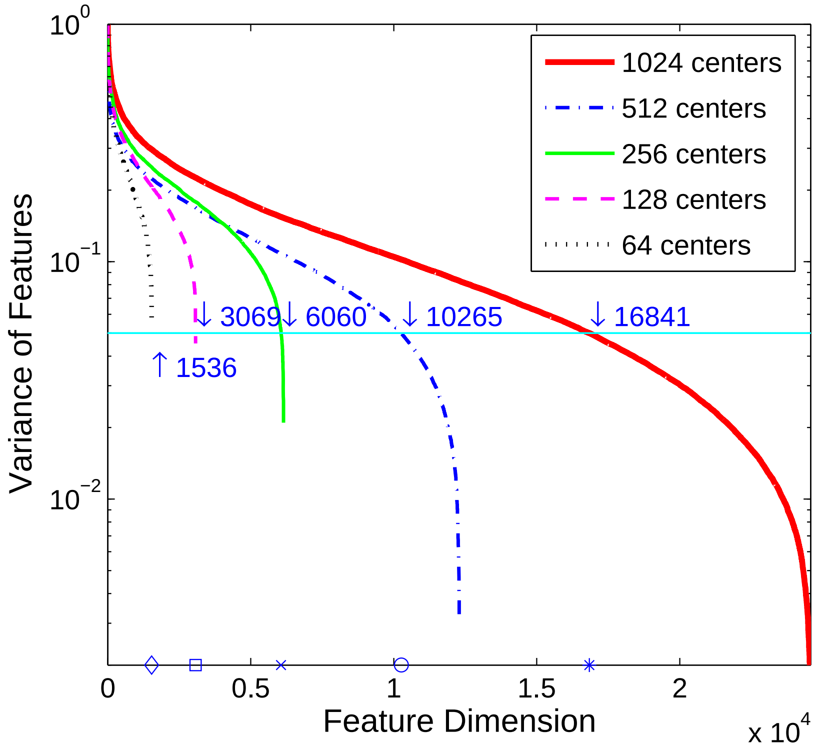


Figure 8.1: The variance of features in the GMM-supervectors with different numbers of mixture components. To facilitate comparison, the maximum variance has been normalized to 1.0. The horizontal line is the variance threshold (= 0.05) above which the features are deemed relevant. The numbers along the horizontal line are the number of relevant features in the supervectors. All results are based on NIST'02.

Methods	EER (%)	Minimum DCF
(A) GMM–UBM	11.19	0.0546
(B) GMM–UBM+TNorm	10.29	0.0428
(C) GMM–UBM+ZTNorm	9.39	0.0393
(D) GMM–SVM+NAP+TNorm	**9.05**	**0.0362**

Table 8.2: Performance of GMM–UBM and GMM–SVM in NIST'02.

Methods	EER (%)	Minimum DCF
(A) GMM–UBM	17.05	0.0615
(B) GMM–UBM+TNorm	16.05	0.0601
(C) GMM–UBM+ZTNorm	15.95	0.0638
(D) GMM–SVM+TNorm	13.40	0.0516
(E) GMM–SVM+NAP+TNorm	**10.42**	**0.0458**

Table 8.3: Performance of GMM–UBM and GMM–SVM in NIST'04 (core test, all trials).

sults show that the KL kernel is slightly better than the GUMI kernel in most of the Common Conditions, and that the K_{GM} kernel performs poorly. Apparently, the K_{GM} kernel is not appropriate for GMM-SVM. One possible reason for the poor performance of the K_{GM} kernel in GMM-SVM is the small relevance factor ($r = 0.01$) in the MAP adaptation. A small relevance factor means that the test- and target-speaker supervectors depend almost exclusively on the test and enrollment utterances, respectively. In other words, the resulting supervectors lose all of the background information pertained in the UBM. As GMM-SVM harnesses the background information via the background speakers' supervectors, a small relevance factor means that these supervectors only represent some speakers different from the target speaker. As a result, a small number of background speakers – which in our case is only 300 (refer to Table 7.4) – may not be able to represent a general population. The results in Table 8.5 agree with this argument; in particular, the performance in terms of EER of the K_{GM} kernel has improved after increasing the relevance factors for both means and variances. While the MinNDCF increases slightly when the relevance factor increases, the increase in MinNDCF is significantly smaller than the reduction in EER.

System	EER (%)					MinNDCF				
	CC1	CC2	CC4	CC7	CC9	CC1	CC2	CC4	CC7	CC9
(A) SVM K_{KL}	**2.82**	**5.21**	**3.81**	**6.70**	**4.27**	0.61	0.62	0.63	0.63	**0.19**
(B) SVM K_{GUMI}	2.83	5.39	4.06	6.99	4.27	**0.57**	**0.61**	**0.61**	0.67	0.24
(C) SVM K_{GM}	4.85	7.31	5.71	7.82	5.13	0.77	0.83	0.83	**0.58**	0.26

Table 8.4: The performance of GMM–SVM (SVM) using three different kernels under different common conditions (CC) in NIST'10. ZT-norm was applied in all cases. "MinNDCF" represents the minimum norm DCF.

Relevance factor	EER (%)			MinNDCF		
	CC4	CC7	CC9	CC4	CC7	CC9
0.01	5.71	7.82	5.13	**0.83**	**0.58**	0.26
8	**4.94**	**7.17**	**4.27**	0.85	0.60	**0.21**
16	5.06	7.26	4.27	0.84	0.60	0.30

Table 8.5: The performance of the K_{GM} kernel in GMM-SVM systems under CC4, CC7, and CC9 of NIST'10 with increasing value of relevance factors for both means and variances.

System	EER (%)					MinNDCF				
	CC1	CC2	CC4	CC7	CC9	CC1	CC2	CC4	CC7	CC9
(A) SC K_{KL}	3.94	6.81	**3.91**	7.26	5.00	0.57	0.69	**0.52**	0.69	0.36
(B) SC K_{GUMI}	**2.93**	**5.25**	4.07	7.04	4.27	0.65	0.65	0.68	**0.64**	**0.26**
(C) SC K_{GM}	3.44	5.66	4.37	**6.70**	**4.24**	**0.54**	**0.65**	0.75	0.77	0.45
(D) SC K_{GM} (T-norm Only)	4.65	9.56	6.53	11.13	7.69	0.50	0.71	0.66	0.80	0.68

Table 8.6: The performance of speaker comparison (SC) using three different kernels under different common conditions (CC) in NIST 2010 SRE. Except for System D, ZT-norm was applied in all cases.

8.1.4 Comparing Kernels in Speaker Comparison

Table 8.6 shows that under the speaker comparison framework, the GUMI kernel outperforms the KL kernel and the K_{GM} kernel in terms of EER. As the GUMI kernel is derived from the Bhattacharyya distance, which is a better similarity measure between two probability distributions than the simplified KL divergence, its performance is better.

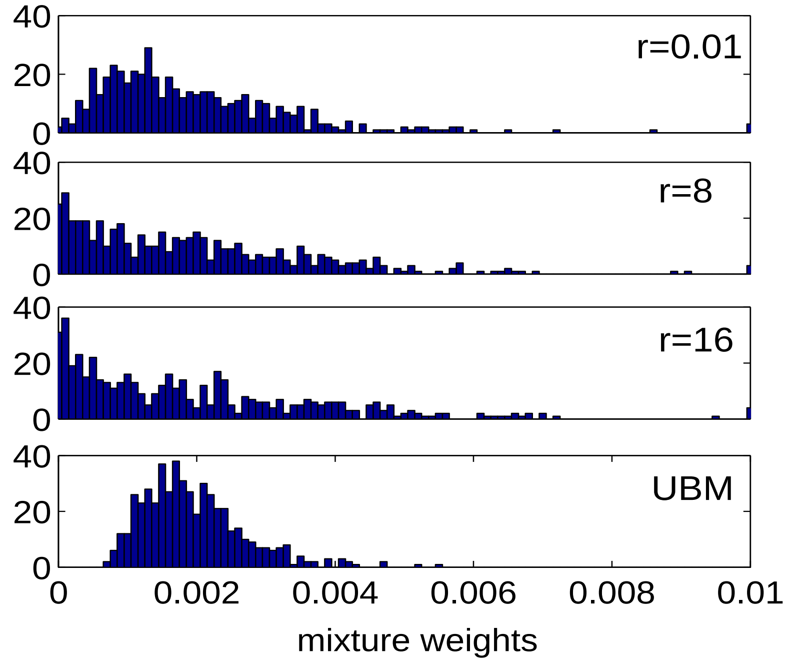


Figure 8.2: Histograms of mixture weights with increasing relevance factor r for the means in the K_{GM} kernel. The bottom panel shows the histogram of UBM's mixture weights.

Unlike the GMM-SVM, we observed that under the speaker comparison framework, the K_{GM} kernel requires a small relevance factor ($r = 0.01$). It is of interest to understand why the same kernel has different requirements under different scoring framework. Consider the case where $r = 16$. With such a large relevance factor,

System	EER (%)					MinNDCF				
	CC1	CC2	CC4	CC7	CC9	CC1	CC2	CC4	CC7	CC9
(A) SVM K_{KL}	2.82	5.21	3.81	6.70	4.27	0.61	0.62	0.63	0.63	0.19
(B) SVM K_{GUMI}	2.83	5.39	4.06	6.99	4.27	0.57	0.61	0.61	0.67	0.24
(C) SVM K_{GM}	4.85	7.31	5.71	7.82	5.13	0.77	0.83	0.83	0.58	0.26
(D) SVM K_{KL} + UP-AVR	2.22	5.37	3.43	5.59	**3.42**	0.53	0.75	0.61	0.64	0.21
(E) SVM K_{KL} + UP-AVR (T-norm Only)	**2.02**	4.91	**3.35**	5.59	4.27	0.44	0.70	0.58	**0.58**	**0.13**
(F) SVM K_{GUMI} + UP-AVR	2.63	5.48	3.51	**5.52**	4.27	0.52	0.74	0.58	0.69	0.18
(G) SVM K_{GUMI} + UP-AVR (T-norm Only)	2.53	5.22	3.43	5.57	4.27	**0.39**	0.67	0.54	0.59	0.15
(H) SC K_{KL}	3.94	6.81	3.91	7.26	5.00	0.57	0.69	**0.52**	0.69	0.36
(I) SC K_{GUMI}	2.93	5.25	4.07	7.04	4.27	0.65	0.65	0.68	0.64	0.26
(J) SC K_{GM}	3.44	5.66	4.37	6.70	4.24	0.54	0.65	0.75	0.77	0.45
(K) SC K_{GM} (T-norm Only)	4.65	9.56	6.53	11.13	7.69	0.50	0.71	0.66	0.80	0.68
(L) JFA	2.72	**4.75**	3.90	6.14	4.27	0.46	**0.53**	0.67	0.68	0.22
(E)+(L)	1.62	3.49	2.69	5.52	4.25	0.31	0.44	0.48	0.59	0.19

Table 8.7: The performance of different systems on NIST'10 under different common conditions (CC). Results were divided into three groups: (1) SVM – GMM-SVM with different kernels, (2) SC – speaker comparison using different inner-product discriminant kernels, and (3) JFA – joint factor analysis models. The lowest EER and MinNDCF across all three groups were displayed in bold. The kernels being compared include the KL-divergence kernel K_{KL}, GMM-UBM mean interval kernel K_{GUMI}, and geometric-mean comparison kernel K_{GM}. Except for Systems E, G and K, ZT-norm was applied in all cases. UP-AVR, the method proposed in this thesis, was applied to two of the GMM-SVM systems for comparison. (E)+(L) denotes the linear score fusion of the best GMM-SVM system and the JFA system.

only the Gaussians that are close to the adaptation data will shift towards the adaptation data, the rest will remain unadapted or only slightly shift in position. This means that these unadapted Gaussians will only occupy a small amount or even no adaptation data. However, in K_{GM}, the mixture weights are estimated using the maximum-likelihood principle, meaning that the mixture weights of the unadapted Gaussians will be very small or even zero. This is undesirable because the extremely small mixture weights severely suppress the contribution of the corresponding components in the inner product (cf. Eq. 3.5), causing loss of speaker information. On the other hand, if the relevance factor is small, say $r = 0.01$, almost all Gaussians will shift towards the adaptation data, resulting in a more even sharing of adaptation data among the Gaussians. This is desirable because almost all mixture components will have contribution to the scoring function. Evidences supporting this argument can be found in Figure 8.2. It shows that when the relevance factor for the means become large, the number of small mixture weights (many of them are zero) increases.

Because no adaptation is applied to the mixture weights in the KL and GUMI kernels, the over-suppression phenomenon will not occur. So, a larger relevance factor for these two kernels can be used.

8.1.5 GMM–SVM vs. Speaker Comparison

Results in Table 8.7 suggest that the performance of GMM-SVM with a KL kernel is better than that of the speaker comparison with a GUMI kernel and a K_{GM} kernel. Comparing Systems J and K in Table 8.7 shows that Z-norm plays an important role in speaker comparison. This is because the scoring function in speaker comparison does not consider the background speakers. Therefore, it is important to harness the impostor information through score normalization methods such as Z-norm. In fact, Z-norm can be considered as a special case of SVM scoring in Eq. 3.1 where the weights corresponding to all of the background speakers are equal.

8.1.6 Effectiveness of UP–AVR

Table 8.8 shows the effect of varying the number of target-speaker's GMM-supervectors (GSVs) on the verification performance in NIST'02. The GMM-supervectors (GSVs) were obtained by either UP or UP-AVR. In all cases, the same partitioning strategy was applied to both target-speaker utterances and background-speaker utterances so that the length of target-speaker sub-utterances matches that of the background-speaker sub-utterances (see Figure 5.3). Because UP-AVR randomizes the feature

No. of Target-Speaker's GSVs	GSV Generation Method	EER (%)	Minimum DCF × 100
1	None ($N = 1$)	9.05 [8.74,9.27]	3.61 [3.45,3.77]
3	UP ($N = 2$)	8.66 [8.36,8.83]	3.45 [3.30,3.59]
	UP-AVR ($N = 4$, $R = 1$)	8.43 [8.09,8.55]	3.65 [3.49,3.81]
5	UP ($N = 4$)	8.46 [8.18,8.64]	3.42 [3.27,3.58]
	UP-AVR ($N = 4$, $R = 1$)	8.21 [7.98,8.46]	3.43 [3.27,3.59]
9	UP ($N = 8$)	8.30 [8.04,8.53]	3.36 [3.21,3.52]
	UP-AVR ($N = 4$, $R = 2$)	8.18 [7.80,8.25]	3.38 [3.22,3.52]
33	UP ($N = 32$)	15.06 [14.63,15.23]	5.14 [4.96,5.29]
	UP-AVR ($N = 4$, $R = 8$)	8.16 [7.93,8.41]	3.38 [3.23,3.53]

Table 8.8: Effect of varying the number of speaker-class supervectors on speaker verification performance in NIST'02. The speaker-class supervectors were generated by utterance partitioning (UP) and utterance partitioning with acoustic vector resampling (UP-AVR). The first column shows the number of speaker-class GSVs, which include the GSVs created by UP or UP-AVR and the GSV produced by the full-length utterance. "None" in the 2nd column means a full-length utterance was used to produce a single supervector for training a speaker-dependent SVM. N and R are the number of partitions per full-length utterance and the number of times resampling were performed to obtain the speaker-class GSVs. When the number of GSVs generated by UP-AVR is larger than the number of required speaker's GSV (e.g., 2nd row with UP-AVR, $N = 4$ and $R = 1$), the speaker's GSVs were randomly selected from the pool. The same utterance partitioning procedure was also applied to the background speakers' utterances so that the length of partitioned background utterances matches with that of the speaker's partitioned utterances. The number of background GSVs is $B(N + 1)$, where B is the number of background speakers (112 for male and 122 for female). The numbers inside the square brackets are the 90% confidence intervals of FAR (at equal error threshold) and minimum DCF found by bootstrapping techniques [3, 4].

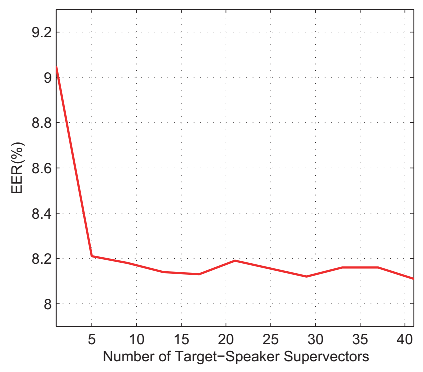
indexes before partitioning the utterances, the supervectors created will be different from simulation run to simulation run. To investigate the reliability of the estimated EER and minimum DCF, 17 independent simulation runs were performed. The mean and standard deviation of 17 EERs and minimum DCFs are shown in the last row of Table 8.8.

Table 8.8 shows that for the same number of target-speaker GSVs, UP-AVR achieves a lower EER than that of UP. Although there is a slight increase in minimum DCF, except for the 2nd row, the increase in minimum DCF is not as significant as the decrease in EERs. Table 8.8 also shows that setting $N = 32$ for UP leads to very poor performance. The reason is that excessive partitioning will produce very short sub-utterances, making the resulting speaker-class GSVs almost identical to the GSV of the UBM after MAP adaptation.

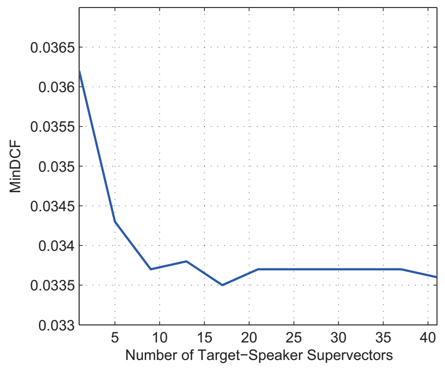
Figures 8.3(a) and (b) show the trend of EER and minimum DCF when the number of speaker-class supervector increases. The figures demonstrate that utterance partitioning can reduce EER and minimum DCF. More importantly, the most significant performance gain is obtained when the number of speaker-class supervectors increases from 1 to 5, and the performance levels off when more supervectors are added. This is reasonable because a large number of positive supervectors will only result in a large number of zero Lagrange multipliers for the speaker class and increase the correlation among the synthesized supervectors.[1] Figure 8.3(c) shows the p-values of McNemar's tests [2] on the pairwise differences between the EERs under different numbers of speaker-class supervectors. The first row suggests that increasing the number of speaker-class supervectors from 1 to 5 and beyond by means of UP-AVR can bring significant reduction in EER. On the other hand, five speaker-class supervectors may already be sufficient because further increase in this number does not bring significant performance gain, as evident by the high p-values in the entries other than the first row.

Figure 8.4 shows the EERs of UP-AVR for different numbers of partitions (N) and resampling (R); when $R = 0$, UP-AVR is reduced to UP. Evidently, for small

[1]Our preliminary investigation on several speakers suggest that when the number of speaker-class supervectors is greater than 40, about half of the supervectors are not support vectors.



(a)



(b)

No. of Target-Speaker GSVs	5	9	13	17	21	29	33	36	41
1	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**	**0.000**
5	-	0.292	0.098	0.129	0.916	0.148	0.320	0.110	0.532
9	-	-	0.650	0.682	0.185	0.719	0.960	0.602	0.648
13	-	-	-	0.958	0.063	1.000	0.678	0.876	0.396
17	-	-	-	-	**0.002**	1.000	0.590	0.957	0.334
21	-	-	-	-	-	0.068	0.199	0.045	0.394
29	-	-	-	-	-	-	0.628	0.916	0.378
33	-	-	-	-	-	-	-	0.510	0.712
36	-	-	-	-	-	-	-	-	0.233

(c)

Figure 8.3: (a) EER and (b) minimum DCF versus number of speaker-class supervectors used for training the speaker-dependent SVMs in NIST'02. The supervectors were obtained by utterance partitioning with acoustic vector resampling (UP-AVR, $N = 4$). (c) p-values of McNemar's tests [2] on the pairwise differences between the EERs in (a). For each entry, $p < 0.005$ means that the difference between the EERs is statistically significant at a confidence level of 99.5%.

number of partitions (e.g., $N = 2$ and $N = 4$), UP-AVR ($R \geq 1$) performs better than UP ($R = 0$), suggesting that resampling can help create better GMM–SVM speaker models. However, when the number of partitions increases (e.g, $N = 8$), the advantage of resampling diminishes. This result agrees with our earlier argument in Chapter 5 that when the number of partitions is too large, the length of sub-

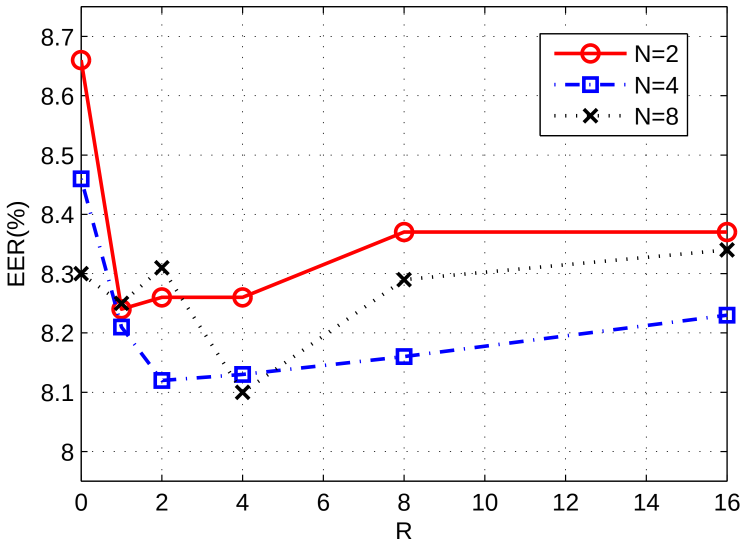


Figure 8.4: Performance of UP-AVR for different numbers of partitions (N) and resampling (R) in NIST'02. When $R = 0$, UP-AVR is reduced to UP.

utterances will become too short, causing their corresponding supervectors almost identical to that of the UBM.

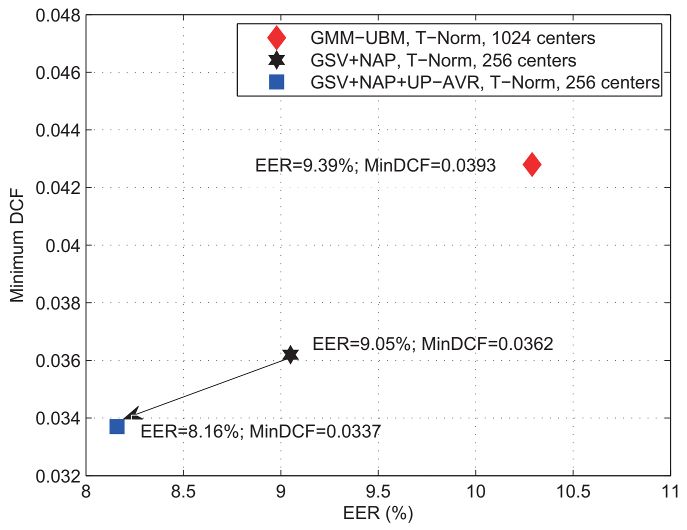
Table 8.9 shows the performance of UP-AVR in NIST'04 when the number of speaker-class supervectors increases from 5 to 201. The results also suggest that with just 5 speaker-class supervectors (UP-AVR(5)), significant reduction in EER can be obtained. However, adding extra speaker-class supervectors can only reduce the EER slightly, which again confirms our earlier argument that it is not necessary to generate excessive number of speaker-class supervectors. The p-value of McNemar's test [2] between System A and System B in Table 8.9 is 7×10^{-9}. Because the p-value is significantly smaller than 0.005 and the EER of System B is higher than other systems that use UP-AVR, we conclude that all of the systems that use UP-AVR are significantly better than the one without using UP-AVR.

Method	EER (%)	Minimum DCF
(A) GMM–SVM+NAP+TNorm	10.42	0.0458
(B) GMM–SVM+NAP+TNorm+UP-AVR(5)	9.67	0.0421
(C) GMM–SVM+NAP+TNorm+UP-AVR(33)	9.63	0.0424
(D) GMM–SVM+NAP+TNorm+UP-AVR(61)	9.63	0.0422
(E) GMM–SVM+NAP+TNorm+UP-AVR(81)	9.57	0.0422
(F) GMM–SVM+NAP+TNorm+UP-AVR(93)	9.57	0.0424
(G) GMM–SVM+NAP+TNorm+UP-AVR(101)	**9.46**	**0.0419**
(H) GMM–SVM+NAP+TNorm+UP-AVR(201)	9.58	0.0421

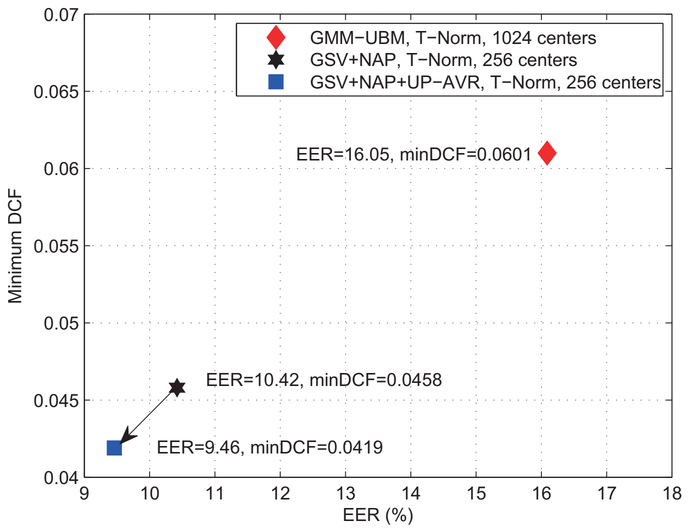
Table 8.9: Performance of GMM–SVM and GMM–SVM with utterance partitioning in NIST'04 (core test, all trials). The numbers inside the parentheses indicate the number of speaker-class supervectors used for training a speaker-dependent SVM, which include the supervectors generated by UP-AVR ($N = 4$, $R = 1, 8, 15, 20, 23, 25, 50$) and the full-length utterance.

Figure 8.5 plots the minimum DCF against the EER for various configurations. It highlights the amount of performance gain that can be obtained by UP-AVR. Figure 8.6 shows the DET curves of various systems, which suggest that GMM–SVM with utterance partitioning is significantly better than the baseline GMM–SVM and GMM–UBM systems for a wide range of decision thresholds.

Our results and other published results in the literature suggest that the EER and minimum DCF in NIST'02 and NIST'04 are higher than those achievable in more recent corpora such as NIST'08. The reason may be that recent results on NIST'08 are typically based on joint factor analysis using a large amount of background data to train the Eigenchannel and Eigenvoice matrices. For example, Dehak et al. [123] used Switchboard, NIST'04, and NIST'05 to estimate these matrices and achieved an EER of 6.55% (all trials). As the amount of data prior to NIST'04 is significantly less than the amount of data prior to NIST'08, it will be difficult to reduce the EER



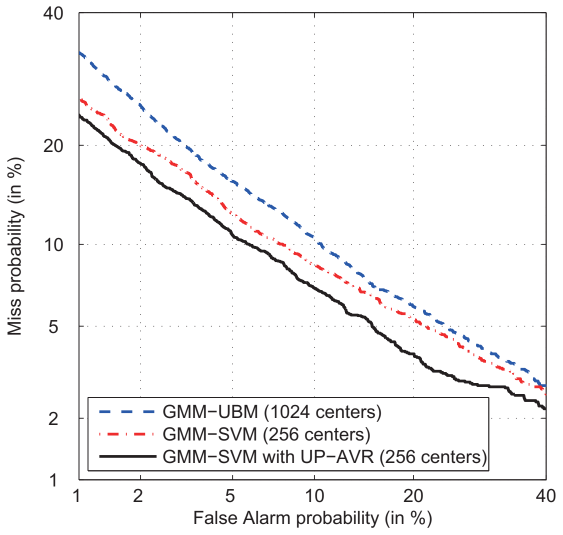
(a) NIST'02



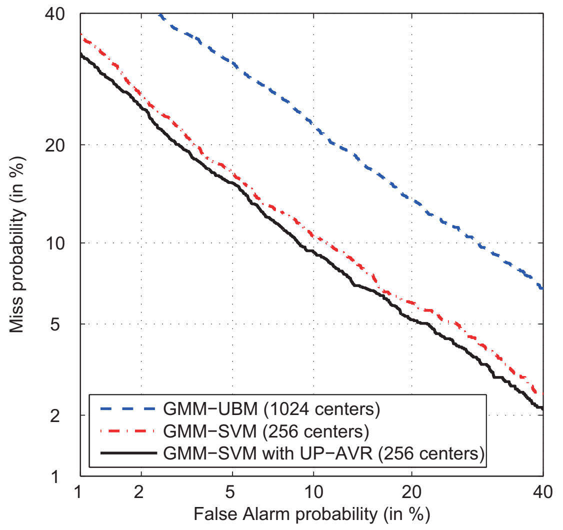
(b) NIST'04

Figure 8.5: Minimum DCF versus EER demonstrating the performance improvement obtained by the utterance partitioning approach in (a) NIST'02 and (b) NIST'04.

of NIST'04 to a level comparable to that of NIST'08.



(a) NIST'02



(b) NIST'04

Figure 8.6: The DET performance of GMM–UBM, GMM–SVM, and GMM–SVM with utterance partitioning in (a) NIST'02 and (b) NIST'04. T-Norm was applied in all cases.

8.1.7 Comparing with Other Systems

GMM-SVM with UP-AVR was also performed on NIST'10 evaluation for proving its effectiveness and comparing with speaker comparison systems and JFA systems. Table 8.7 shows that under most of the common conditions, UP-AVR helps improve the performance of GMM-SVM systems for both KL and GUMI kernels. In terms of EER, the performance of UP-AVR in GMM-SVM systems (system E) is consistently better than that of the speaker comparison systems and the JFA system (except for Common Condition 2). This further demonstrates that UP-AVR is effective in solving the imbalance data problem in GMM-SVM systems.

The scores of JFA and GMM-SVM with UP-AVR (KL kernel, T-norm only) were fused using a set of linear fusion weights that achieve the best fusion performance (in terms of minimum EER). Table 8.7 shows that the fusion improves the performance for Common Conditions 1, 2, and 4.

8.2 I-Vector Based Speaker Verification

8.2.1 Effects of Training-set Size on I-Vectors

This experiment is to analyze the effect of the number of training utterances and training speakers on the discriminative power of LDA and WCCN projected i-vectors.

The training set comprises the i-vectors of 191 male speakers in NIST'05–NIST'08, with each speaker having 10 i-vectors (sessions). For each experiment, a subset of i-vectors was extracted from this training set to train the LDA and WCCN projection matrices. More precisely, the numbers of i-vectors per speaker were set to 6, 8, and 10. For each configuration, the number of speakers S was progressively increased from 60 (80 when there are only 6 utterances per speaker)[2] to 191. The resulting LDA+WCCN matrices were then used to project two thousand 400-dim i-vectors extracted from 90 speakers in NIST 2010 SRE to i-vectors of dimensions $S-1$ or 150, whichever is less.[3] The discriminative power of the projected i-vectors was quantified by minimum DCF derived from 22,198 intra- and 1.9 million inter-speaker cosine-distance scores without score normalization.

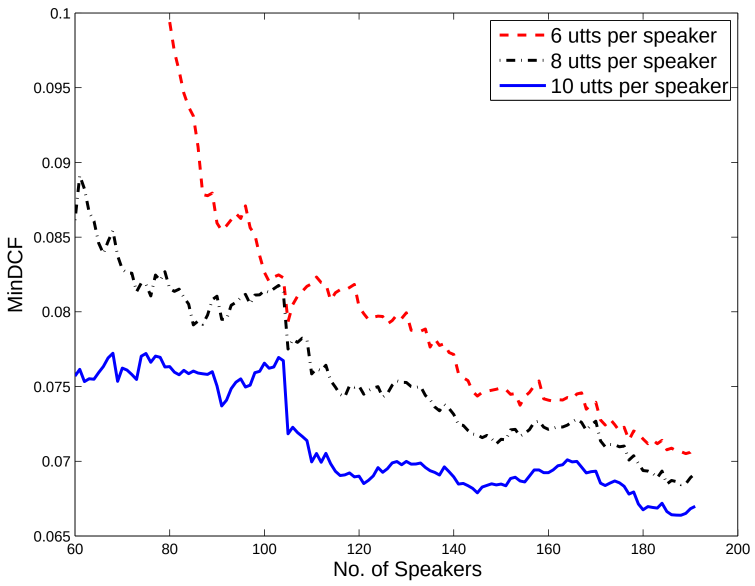


Figure 8.7: Minimum decision cost (MinDCF) versus the number of speakers used for estimating the LDA and WCCN projection matrices. Each speaker has either 6, 8, or 10 utterances for estimating the i-vectors.

Figure 8.7 shows the minimum DCF achieved by the projected i-vectors when the number of speakers and the number of utterances per speaker used for training the LDA+WCCN projection matrices increase. The results suggest that when the

[2]When the number of utterances per speaker was limited to 6 and the number of speakers is smaller than 80, the within-class covariance matrix $\mathbf{S}_w$ is close to singular, causing numerical difficulty in estimating the projection matrices.

[3]Because rank $\left(\mathbf{S}_w^{-1}\mathbf{S}_b\right) = \min\{400, S-1\}$.

number of utterances per speaker is small (≤ 8) the discriminative power of i-vectors *generally* increases when the number of speakers used for training the transformation matrices increases. The increase is more prominent when the number of utterances per speaker is very small (say 6), suggesting that more speakers are required when the number of utterances per speaker is very small. However, when the number of utterances per speaker is sufficiently large (say 10), increasing the number of speakers does not bring significant benefit until the number of speakers is larger than 105. Among the three different numbers of utterances per speaker, using 10 utterances per speaker achieves the lowest minimum DCF regardless of the number of speakers used for training the transformation matrices, suggesting that it is better to use more utterances per speaker than using more speakers but less utterances per speaker. The small fluctuation in minDCF suggests that the channel variability of some speakers in NIST'05–NIST'08 may not match the channel variability in NIST'10, causing slight performance degradation when these speakers were added to the training pool.

8.2.2 Small Sample-Size Problem in LDA and WCCN

The numerical difficulty in estimating the LDA and WCCN transformation matrices is due to insufficient rank in the within-speaker scatter matrix (Eq. 4.11) when the training set size is small. We have investigated two classical approaches to alleviating this small sample-size problem [22]. They are pseudo-inverse LDA and PCA+LDA.

1. *Pseudo-inverse LDA*. The rank deficiency problem can be avoided by replacing the inverse of the within-speaker scatter matrix by its pseudo inverse [84, 85]. The idea is that during singular value decomposition, any components with singular values smaller than a threshold will be automatically discarded by the

pseudo-inverse procedure.

2. *PCA+LDA*. We used PCA to project the training i-vectors to a lower dimension space prior to computing the within-speaker scatter matrix [82,83]. With the reduction in the i-vector dimension, the rank requirement of LDA and WCCN can be reduced to a comfortable level for reliable estimation of the LDA and WCCN transformation matrices.

Methods	EER (%)						
	M=2	M=3	M=4	M=5	M=6	M=7	$M\geq$8
Without LDA and WCCN	12.60						
LDA+WCCN	23.39	22.25	6.98	5.51	4.59	4.22	2.98
PI-LDA+WCCN	19.02	20.90	6.98	5.51	4.59	4.22	2.98
PCA+LDA+WCCN	13.37	9.05	6.29	5.14	4.32	3.86	2.98
UP-AVR(2)+LDA+WCCN	6.64	5.78	4.99	4.52	4.08	3.90	2.94
UP-AVR(4)+LDA+WCCN	**6.16**	**5.09**	**4.46**	4.05	**3.85**	3.68	**2.90**
UP-AVR(8)+LDA+WCCN	6.23	5.09	4.48	**3.88**	3.87	**3.65**	2.97
Methods	MinNDCF						
	M=2	M=3	M=4	M=5	M=6	M=7	$M\geq$8
Without LDA and WCCN	0.90						
LDA+WCCN	1.00	1.00	0.87	0.81	0.76	0.75	0.63
PI-LDA+WCCN	0.99	1.00	0.87	0.81	0.76	0.75	0.63
PCA+LDA+WCCN	1.00	0.95	0.88	0.82	0.77	**0.73**	**0.63**
UP-AVR(2)+LDA+WCCN	**0.91**	**0.87**	**0.83**	**0.79**	**0.75**	0.74	0.66
UP-AVR(4)+LDA+WCCN	0.93	0.89	0.85	0.79	0.76	0.75	0.66
UP-AVR(8)+LDA+WCCN	0.92	0.89	0.86	0.80	0.78	0.76	0.69

Table 8.10: The performance of using different methods for solving the small sample-size problem in LDA and WCCN. $M = x$ means each speaker only has x recordings for training the LDA and WCCN matrices. $M \geq 8$ means each speaker provides at least 8 recordings, with an average of 31 recordings per speaker. “LDA”: the conventional LDA; “PI-LDA”: pseudo-inverse LDA; “PCA + LDA”: perform PCA before LDA; “UP-AVR(N)”: dividing each of the full-length training utterances into N partitions using UP-AVR, with the number of re-sampling R set to 4.

Table 8.10 shows the performance achieved by different approaches to alleviating the small-sample size problem when the number of recording sessions per training speaker (M) increases from 2 to 8 or above. The performance is obtained by concatenating the scores under Common Conditions 1, 2, 4, 7, and 9 in NIST'10. The performance achieved by "Without LDA and WCCN" is considered as the baseline. For "LDA+WCCN", the performance is very poor when $M \leq 3$, because the within-speaker scatter matrix is close to singular. Only when $M \geq 4$, the benefit of LDA+WCCN becomes apparent. These observations also agree with the findings in [124].

Table 8.10 also shows the following properties:

1. when $M \leq 3$, pseudo-inverse LDA can help to avoid the singularity problem. However, this method leads to i-vectors that perform even poorer than those without LDA+WCCN projections. When the within-class scatter matrices have full rank ($M \geq 4$), the performance of pseudo-inverse LDA is the same as the classical LDA.

2. Preprocessing the i-vectors by PCA not only avoids the singularity problem but also helps the LDA to find a better projection matrix. However, when the rank of within-class scatter matrices is too low (e.g., when $M = 2$), the performance of PCA preprocessing is poorer than those without LDA+WCCN projections. Moreover, the effect of PCA diminishes when the number of recordings per training speaker is sufficient ($M \geq 8$).

3. UP-AVR is an effective way to produce more informative i-vectors from a single utterance, thus effectively avoiding the singularity problem in LDA. It also achieves the best performance among all methods investigated.

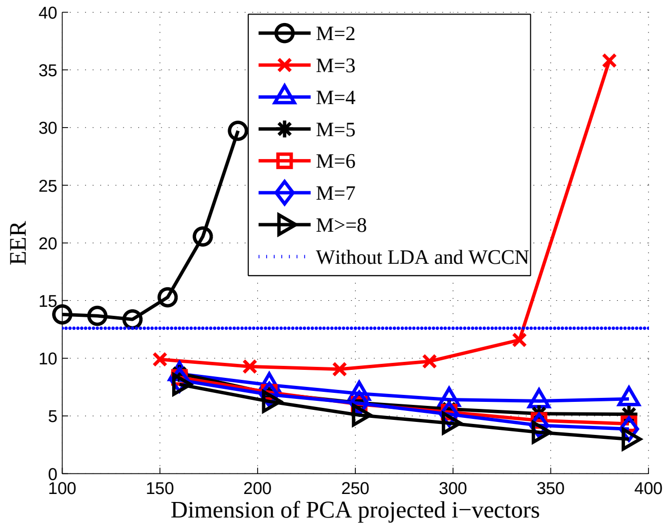


Figure 8.8: EER versus the dimension after PCA projection. $M = x$ means each speaker only has x recordings for training the LDA and WCCN matrices.

Figure 8.8 shows the effect of varying the dimension of PCA projection on the performance of PCA+LDA. The results suggest that when the number of sessions per speaker (M) is equal to two, PCA cannot help the LDA for all projection dimension. In fact, the performance is even poorer than that without LDA (dotted line). This is caused by insufficient data for training the LDA, even though PCA can help solving the singularity problem. The result also suggests that setting the PCA projection dimension close to the rank of within-class scatter matrices is not a good idea when $M \leq 3$.

8.2.3 Effectiveness of Multi-way LDA

To compare the effectiveness of PCA+LDA and multi-way LDA, we selected 63 male speakers from NIST'08 for training the LDA and WCCN projection matrices.

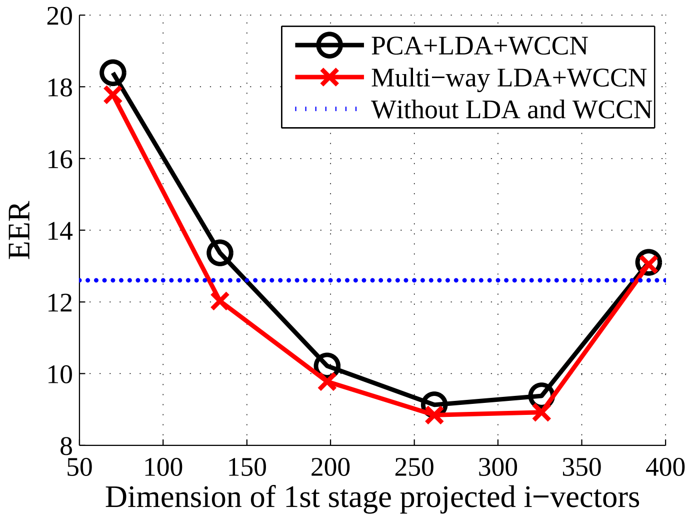


Figure 8.9: EER versus the dimension of the projected i-vectors in the first stage of PCA+LDA and multi-way LDA. The number of recordings per speaker is 8 ($M = 8$). Refer to Section 8.2.3 for the explanation of "1st stage".

Unlike the previous experiments, these speakers use the same set of microphones in the recording sessions. This arrangement allows us to arrange the training i-vectors in a grid, as explained in Section 4.2.3.

Note that both PCA+LDA and multi-way LDA divide the inter-session compensation into two stages. In the 1st stage, i-vectors are projected into a lower dimensional space via PCA or via the matrix $\mathbf{C}$ in Eq. 4.16. Then, in the 2nd stage, the dimension of the projected i-vectors is further reduced by LDA to 60.[4] Figure 8.9 shows the effect of varying the projection dimension in the first stage for both PCA+LDA and multi-way LDA. Evidently, the performance of both methods has a similar trend with respect to this dimension, with multi-way LDA always performs slightly better than PCA+LDA for all projection dimensions. Table 8.11 also

[4]Because rank $\left(\mathbf{S}_w^{-1}\mathbf{S}_b\right) = \min\{400, S-1\} = 62$, the projected dimension should be set to a value smaller than this rank.

Systems	MinNDCF		EER (%)	
	$M = 7$	$M = 8$	$M = 7$	$M = 8$
(A) Without LDA and WCCN	0.90	0.90	12.60	12.60
(B) LDA+WCCN	–	0.99	–	15.29
(C) PI-LDA+WCCN	1.00	0.99	23.33	15.29
(D) PCA+LDA+WCCN	0.97	0.96	10.27	9.13
(E) MW-LDA+WCCN	**0.92**	**0.93**	**9.97**	**8.85**

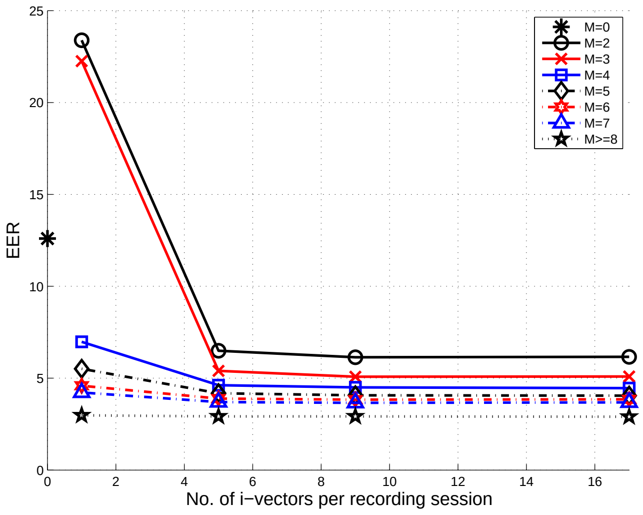
Table 8.11: The performance of Multi-way LDA and other LDA methods. *MW-LDA + LDA*: Multi-way LDA. $M = x$ means each speaker only has x recordings for training the LDA and WCCN matrices. "–" denotes the situation where singularity occurs when estimating the projection matrices.

shows that multi-way LDA outperforms PCA+LDA.

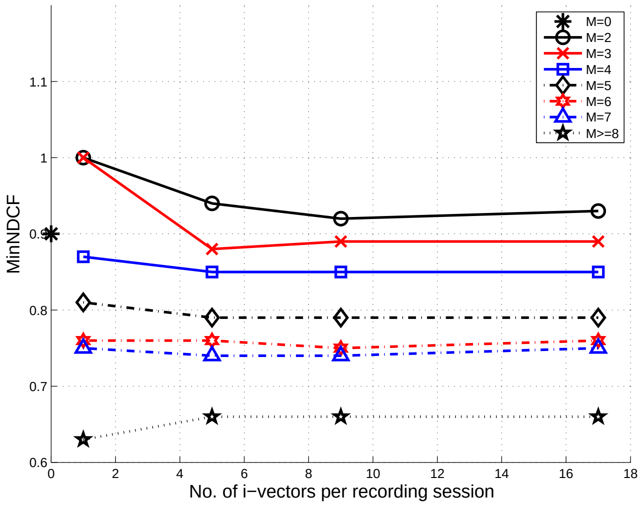
8.2.4 UP-AVR for LDA and WCCN

This experiment investigates the effectiveness of UP-AVR for solving the singularity problem in LDA. Similar to Section 8.2.2, the number of recording sessions per training speaker was increased from 2 to 8 and above. The results in Table 8.10 show that when UP-AVR is applied to increase the number of i-vectors per training speaker, the performance of LDA+WCCN improves significantly. Although many of the i-vectors produced by UP-AVR are extracted from the sub-utterances of the same recording sessions, they possess sufficient speaker-dependent information for training the LDA and WCCN projection matrices and can help LDA to find a subspace with less intra-speaker variation by alleviating the numerical problem. Nevertheless, the contribution of UP-AVR to LDA and WCCN diminishes when the number of recordings per training speaker is sufficient (over 8 per speaker in our experiments).

Figs. 8.10(a) and 8.10(b) depict the trend of EER and minimum DCF when the



(a)



(b)

Figure 8.10: (a) EER and (b) minimum DCF versus number of i-vectors per recording session for different numbers of recording session per speaker for training the LDA and WCCN matrices. The i-vectors were obtained by utterance partitioning with acoustic vector resampling (UP-AVR, $N = 4$; $R = 1, 2, 4$). M is the number of recordings per speaker used for training the matrices, $M = 0$ means without LDA and WCCN, and $M \geq 8$ means at least 8 utterances per speaker were used for training.

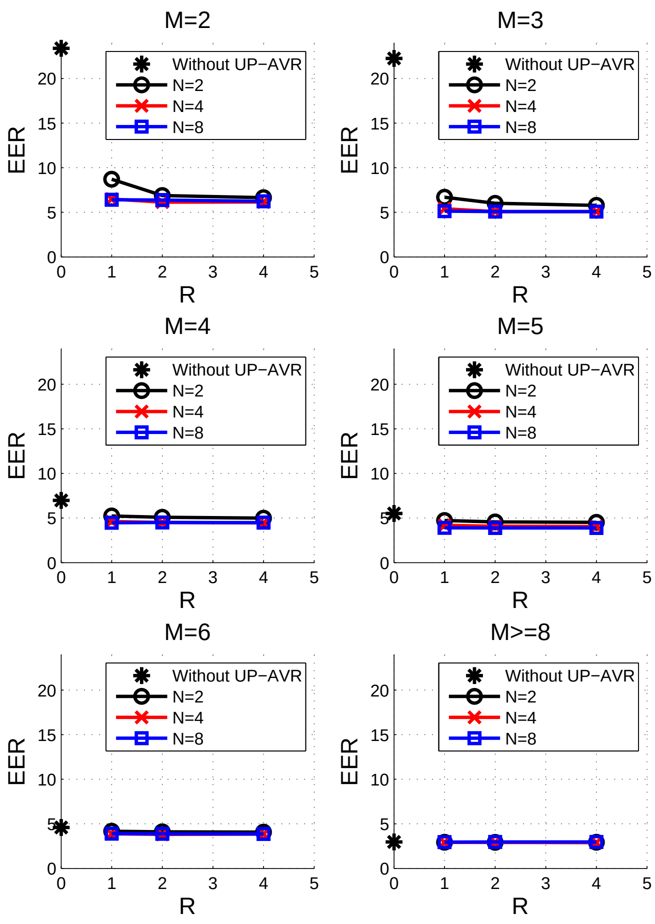


Figure 8.11: The effect of varying the number of partitions (N) and the number of resampling (R) on the performance of UP-AVR. $R = 0$ means without applying UP-AVR to the utterances.

number of recording sessions per speaker and the number of i-vectors per recording session for training the LDA and WCCN matrices increase. The results demonstrate that the most significant performance gain is obtained when the number of i-vectors per recording session increases from 1 to 5, and the performance levels off when more i-vectors are added.

Figure 8.11 shows the performances of UP-AVR for different numbers of partitions (N) and resampling (R) for different numbers of recordings per speaker. According to Figure 8.11, when the number of recordings per speaker (M) is less than five, increasing the number of partitions and resampling times can improve the performance. However, when $M \geq 6$, the effect of varying N and R diminishes, suggesting that UP-AVR is most effective for training the LDA and WCCN matrixes when the number of recording sessions per speaker is very limited.

8.2.5 UP-AVR for SVM Scoring

In this experiment, we used all of the available interview and microphone speech from NIST 2005–2008 SRE to train the LDA and WCCN matrices. The focus of the experiment is on comparing SVM scoring against cosine distance scoring.

Table 8.12 compares the performance between SVM scoring and cosine distance scoring in NIST'10. Table 8.12 shows that the performance of SVM scoring is slightly worse than that of cosine distance scoring. This may be caused by the data imbalance problem in SVM training. However, after applying UP-AVR to SVM training, the performance of SVM improves. More specifically, increasing the number of target-speakers i-vectors from one i-vector per target-speaker to 9 i-vectors per target-speaker reduces the EER of SVM scoring from 3.26% to 2.71%, which amounts to 17% relative reduction. Similarly, the method reduces the minimum DCF from 0.52 to 0.51, which amounts to 2% relative reduction. This performance improvement makes SVM scoring outperforms cosine distance scoring significantly, as evident by the results (CDS versus SVM+UP-AVR) in Table 8.12. Specifically, when UP-AVR was applied to SVM scoring, the EER and minimum DCF reduce to 2.71% and 0.51, respectively, which amount to 9% and 19% relative reduction.

Scoring Methods		EER(%)						MinNDCF					
		CC1	CC2	CC4	CC7	CC9	Mic	CC1	CC2	CC4	CC7	CC9	Mic
CDS		1.62	2.86	3.23	8.94	**1.71**	2.98	0.42	0.52	0.54	0.99	0.46	0.63
CDS+UP-AVR(1)		1.74	3.01	3.45	9.44	2.29	3.17	0.45	0.54	0.57	0.99	0.49	0.67
CDS+UP-AVR(2)		1.75	3.02	3.44	9.49	2.27	3.21	0.45	0.54	0.57	0.99	0.49	0.67
CDS+UP-AVR(4)		1.76	3.01	3.44	9.49	2.26	3.22	0.45	0.54	0.57	0.99	0.50	0.67
SVM	$C = 1$	1.82	3.08	3.32	9.50	2.56	3.26	0.32	0.49	0.50	**0.94**	0.28	0.52
	$C = 0.01$	1.85	3.26	3.47	9.49	2.56	3.31	0.32	0.48	0.46	0.99	0.36	0.55
SVM+UP-AVR(1)	$C = 1$	1.54	2.86	3.17	9.50	2.24	3.04	0.28	0.47	0.46	0.95	0.23	0.49
	$C = 0.01$	1.51	2.99	3.02	9.37	2.34	2.97	0.29	0.47	0.41	0.99	0.28	0.51
SVM+UP-AVR(2)	$C = 1$	1.57	2.84	3.02	9.50	2.16	3.04	0.27	0.47	0.45	0.96	0.23	0.49
	$C = 0.01$	1.41	2.62	**2.81**	**8.38**	2.39	2.71	0.28	0.46	**0.40**	0.99	0.32	0.51
SVM+UP-AVR(4)	$C = 1$	1.54	2.85	3.10	9.47	2.17	3.03	0.27	0.47	0.45	0.95	**0.23**	**0.49**
	$C = 0.01$	**1.31**	**2.62**	2.84	8.66	2.42	**2.71**	**0.28**	**0.45**	0.41	0.99	0.31	0.51

Table 8.12: The performance of i-vector based speaker verification in NIST'10 using different scoring methods. C is the SVM's penalty factor. "CC" denotes common condition. "Mic" represents all common conditions involving interview-style speech or microphone speech. "CDS": cosine distance scoring. "CDS+UP-AVR(R)": computing the average cosine distance score between the claimant's LDA+WCCN projected i-vector and $NR + 1$ target-speaker's i-vectors produced by UP-AVR with number of partitions $N = 4$ and number of re-sampling $R = 1, 2$, or 4. "SVM+UP-AVR(R)": SVM scoring with each SVM trained by using $NR + 1$ target-speaker's LDA+WCCN projected i-vectors and 633 background speakers' i-vectors, where the target-speaker i-vectors were produced by UP-AVR with number of partitions $N = 4$ and number of re-sampling $R = 1, 2$, or 4.

Note that UP-AVR can also be applied to cosine distance scoring. Specifically, instead of training an SVM for each target speaker, we used the RN i-vectors produced by UP-AVR together with the one estimated from the full-length enrollment utterance to represent a target speaker. During verification, given a test utterance, we computed the average cosine distance score between the i-vector of the test utterance and each of these $(RN + 1)$ target-speaker i-vectors. The rows labeled with "CDS+UP-AVR" in Table 8.12 show the performance of this strategy. Evidently, unlike the situation in SVM scoring, UP-AVR cannot improve the performance of cosine distance scoring. This result is reasonable because the discriminative power of the generated (RN) i-vectors is poorer than that derived from the full-length utterances, which has detrimental effect on the average score. On the other hand, in SVM scoring, given the $RN + 1$ target-speaker's i-vectors, the SVM training algorithm can select a more relevant subset from these target-speaker's i-vectors and the background i-vectors to form a decision boundary that best discriminate the target speaker from impostors. As the SVM score is a linear weighted sum of the cosine-distance scores of these relevant (support) i-vectors and the test i-vector, each of the target-speaker's i-vectors has different contribution to the overall score and the degree of contribution is optimized by the SVM training algorithm. The aims of UP-AVR in SVM scoring is to overcome the data-imbalance problem in SVM training. Once this data-imbalance problem can be alleviated, the SVM weights can be reliably estimated.

Results in Table 8.12 also suggest that when UP-AVR is applied, a small penalty factor C is more appropriate than a large one.[5] This is reasonable because a small C leads to more target-speaker class support vectors, which improve the influence of target-speaker class data on the decision boundary of the SVMs.

[5]As we used a scalar for the '`boxconstraint`' parameter in `svmtrain.m` provided by Mathworks, the penalty factors for speaker and impostor classes will be rescaled according to the number of training samples in these two classes.

8.3 Gaussian PLDA with Sparse Kernel Machines

8.3.1 Choice of I-Vectors for Empirical Kernel Maps

The empirical kernel maps aim to emphasize the differences between target and non-target speakers and to represent the differences in the PLDA score space. According to Eq. 6.24, the empirical kernel map comprises two parts: (1) PLDA scores between a training or test i-vector and the multiple enrollment i-vectors $\mathcal{X}_s$ of a target speaker; and (2) PLDA scores between a training or test i-vector and a subset of background i-vectors. It is of interest to investigate the roles played by these two parts by varying their size, i.e., varying H_s and B' in Eq. 6.24.

To this end, we selected 50 target speakers from NIST'12. Each of these target speakers has at least 15 enrollment utterances, which result in 15 enrollment i-vectors. A subset of these i-vectors were used for constructing the empirical kernel maps, with the subset size increases from 0 (i.e., none was used) to 15 (i.e., all of them were used). Note that when the subset size is zero, the empirical kernel maps are similar to the anchor model [106–108] where the score space is defined by the background speakers only. We also selected 704 background speakers' i-vectors (i.e., $B = 704$ in Eq. 6.22) from the utterances in NIST'05–NIST'08. Note that although the dimension of PLDA score vectors was varied in the experiments, the number of speaker-class and impostor class score vectors were fixed. This means that all of the i-vectors in $\mathcal{X}_s$ and $\mathcal{X}_b$ were used as input to the empirical kernel map to produce at least 15 speaker-class PLDA score vectors and 704 impostor-class score vectors for SVM training, regardless of the dimension of the PLDA score vectors. All of the SVMs have an RBF kernel with width parameter $\gamma = 1000$.[6]

[6]Because the LR scores range between -579.5 and 199.8 and the dimension of the LDA-projected i-vector is 200, a large value of γ is necessary.

We have also investigated the situation where each target speaker has one enrollment utterance only. The procedure is basically the same as the one above, except that the enrollment i-vectors were produced by applying UP-AVR to the single enrollment utterance.

Figure 8.12 and Figure 8.13 show the EER and minimum DCF with respect to H_s in Eq. 6.21. Both figures show that using more enrollment i-vectors for constructing empirical kernel maps improves performance, but the improvement becomes insignificant when the number of enrollment i-vectors exceeds five. Figure 8.12 and Figure 8.13 also suggest that the performance of Config 1 is much better than that of Config 2, which means that using the i-vectors estimated from full-length utterances for constructing empirical kernel maps and training SVMs is better than using the i-vectors estimated from sub-utterances. This result is reasonable because full-length utterances contain much more information than partitioned sub-utterances.

Table 8.13 shows the effect of varying the number of background speakers' i-vectors B' for constructing the empirical kernel maps on the performance of CC2. To this end, the number of target speaker's enrollment i-vectors H_s is fixed and 704 background speakers' i-vectors were used for training speaker-dependent SVMs.[7] The RBF parameter γ was fixed to 1800 for all cases. Table 8.13 suggests that it is not necessary to incorporate a lot of background speakers' i-vectors to build the empirical kernel maps.

8.3.2 *Choice of Non-Targets for SVM Training*

NIST'12 allows systems to use the known non-target speakers for each verification trial. It is of interest to investigate the benefit of using known non-target speakers for

[7] $B' \leq 704$, i.e., only a subset of background speakers' i-vectors is used for constructing the empirical kernel maps.

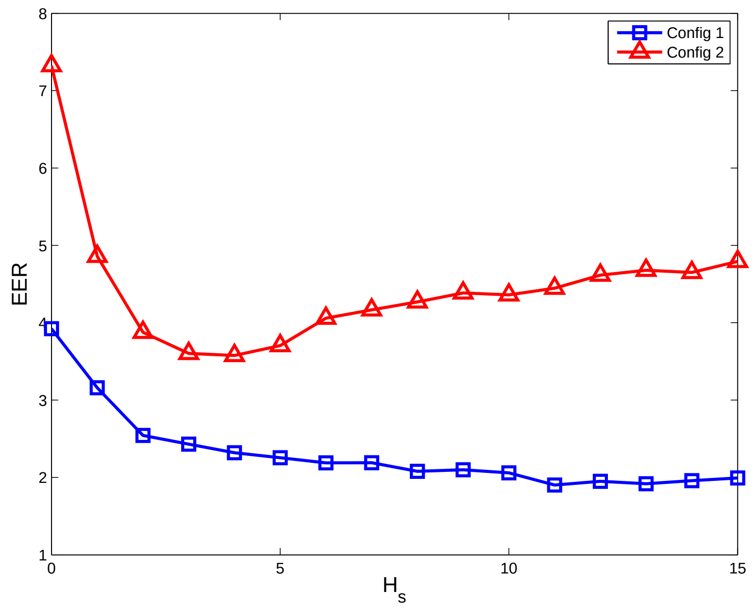


Figure 8.12: Equal Error Rate versus the number of enrollment i-vectors per target speaker (H_s in Eq. 6.21) for constructing the empirical kernel maps. *Config 1*: Each target speaker has at least 15 enrollment utterances and the i-vectors were obtained from these utterances without applying UP-AVR. *Config 2*: Each target speaker has one enrollment utterance only and all enrollment i-vectors (for SVM training and empirical kernel map construction) were generated by applying UP-AVR on this utterance.

SVM training. To this end, the number of utterances per known non-target speaker (J) was varied. Specially, for each target-speaker SVM that uses Empirical Kernel Map I (Eq. 6.23), $722 \times J$ known non-target utterances were used as imposter-class data for training. Note that not all target speakers contain multiple enrollment utterances, and some target speakers only contain one enrollment utterance. Therefore, UP-AVR ($N = 4, R = 4$) was performed to produce more sub-utterances for these target speakers.

Table 8.14 shows the performance of using unknown non-target speakers and known non-target speakers for training the SVMs. The results in Table 8.14 sug-

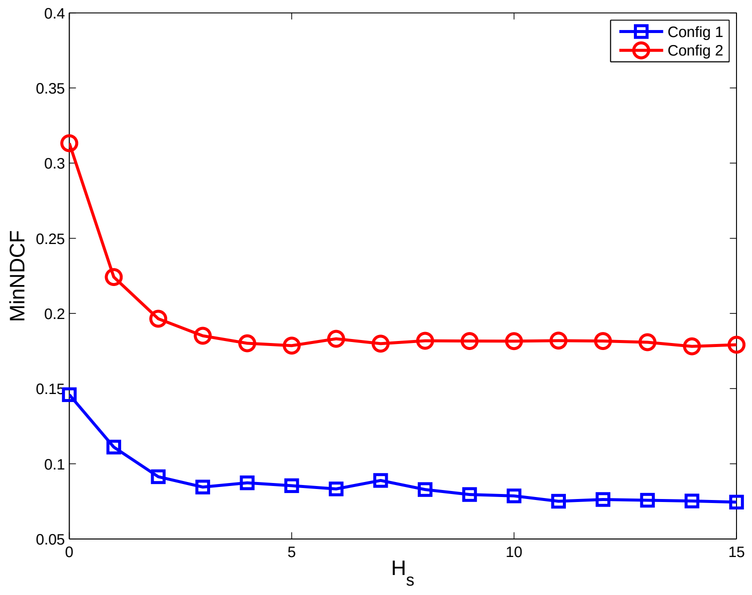


Figure 8.13: Minimum normalized DCF versus the number of enrollment i-vectors per target speaker (H_s in Eq. 6.21) for constructing the empirical kernel maps. See the caption in Figure 4 for *Config 1* and *Config 2* in the legend.

gest that using either known non-targets or unknown non-targets achieves similar performance.

8.3.3 Use of Noisy Speech for PLDA Training

In NIST'12, the test utterances have different SNRs under different common evaluation conditions. For example, the test utterances in CC2 is rather clean and have high SNR, whereas some of the test utterances in CC4 is fairly noisy because noise was artificially added to the clean speech files and the test utterances in CC5 were recorded in noisy environments. Figure 8.14 shows the distributions of the SNR of the test utterances in CC2, CC4, and CC5. Obviously, CC4 is the most difficult condition for speaker verification systems because the range of SNR is large.

No. of Bkg. I-vectors (B')	EER (%)	MinNDCF(2012)
0	1.87	0.309
50	1.87	0.319
100	1.87	0.314
150	**1.84**	0.301
200	1.87	0.297
250	1.94	0.295
300	1.97	0.293
350	2.05	**0.289**
400	2.04	0.292
450	2.05	0.299
500	2.01	0.299
550	1.96	0.302
600	2.01	0.302
650	2.01	0.300
700	1.98	0.299
704	2.01	0.299

Table 8.13: Performance of varying the number of background speakers' i-vectors for constructing empirical kernel maps under CC2 of NIST'12. The first column represents the number of background speakers' i-vectors for constructing the empirical kernel maps.

If the operating environment of a speaker verification system is known a prior, it is possible to use multi-condition training to improve performance [110,122,125–128]. In this work, we compared clean-condition training with multi-condition training and investigated different ways of applying multi-condition training for different common conditions in NIST'12. To this end, we created two sets of noise contaminated files by adding HAVC or crowd noise (from [117]) to the original clean utterances at 6dB and 15dB SNR. One set of noise files has mean SNR at 6dB and another set has mean SNR at 15dB. The clean and noisy files were used for training the PLDA models.

We have three ways of using the training data.

Source of Imposter Class for Training SVMs	No. of Utts. per Known Non-target	EER (%)	MinNDCF (2012)
Unknown Non-targets	—	1.80	0.312
Known Non-targets	1	1.83	0.324
	2	**1.79**	0.310
	3	1.84	0.306
	4	1.90	0.302
	5	1.91	0.303
	6	1.91	0.300
	7	1.87	0.299
	8	1.87	0.300
	9	1.87	0.299
	10	1.87	**0.298**

Table 8.14: Performance (CC2 in NIST'12) of PLDA-SVM scoring using different types of non-target speakers for training the SVMs.

- *Clean-Condition Training*: Use clean training utterances for all common conditions.

- *Multi-Condition Training I*: Use the original (clean) training utterance plus the sets of noise contaminated utterances, where the SNR of the noise contaminated utterances matches that of the test utterances. This multi-condition training imposes strict requirements and assumptions on the operating environment. It is expected that if the operating environment deviates significantly from the assumption, performance will suffer.

- *Multi-Condition Training II*: Use the original (clean) training utterance plus two sets of noise contaminated utterances (6dB and 15dB) for all common conditions. This multi-condition training uses all of the available (clean plus noisy) data regardless of the distribution of test utterances' SNR. When compared to Multi-Condition Training I, this mode of multi-condition training has

Training Methods	EER (%)			MinNDCF(2012)		
	CC2	CC4	CC5	CC2	CC4	CC5
Clean-Condition Training	2.40	4.14	2.79	0.333	0.410	0.343
Multi-Condition Training I	**2.40**	**3.24**	**2.72**	0.333	**0.333**	**0.313**
Multi-Condition Training II	2.54	3.24	3.11	**0.304**	0.333	0.325

Table 8.15: Performance (core set, male speakers) of different PLDA training methods for different common conditions of NIST'12.

a relaxed assumption on the operating environment. Because all the available environment-dependent training data have been used for training the PLDA models, the performance of the resulting system will not suffer significantly even if the operating environment exhibits a wide range of SNR.

Table 8.15 shows the performance of these three training methods. It shows that the performance of CC4 can be improved significantly by multi-condition training. Recall that CC4 involves test utterances contaminated with noise. It is therefore reasonable that multi-condition training is beneficial. Table 8.15 also shows that Multi-Condition Training I performs the best for CC5. Recall from Figure 8.14 that the SNR of CC5 ranges from 15dB to 50dB. As a result, Multi-Condition Training I excludes the 6dB utterances when training the PLDA model. On the other hand, Multi-Condition Training II uses the 6dB utterances as well, resulting in mismatches in the SNR of the training and test environments. The same reason also explains the poorer EER performance of Multi-Condition Training II on CC2, because the majority of the test utterances in this common condition are fairly clean.

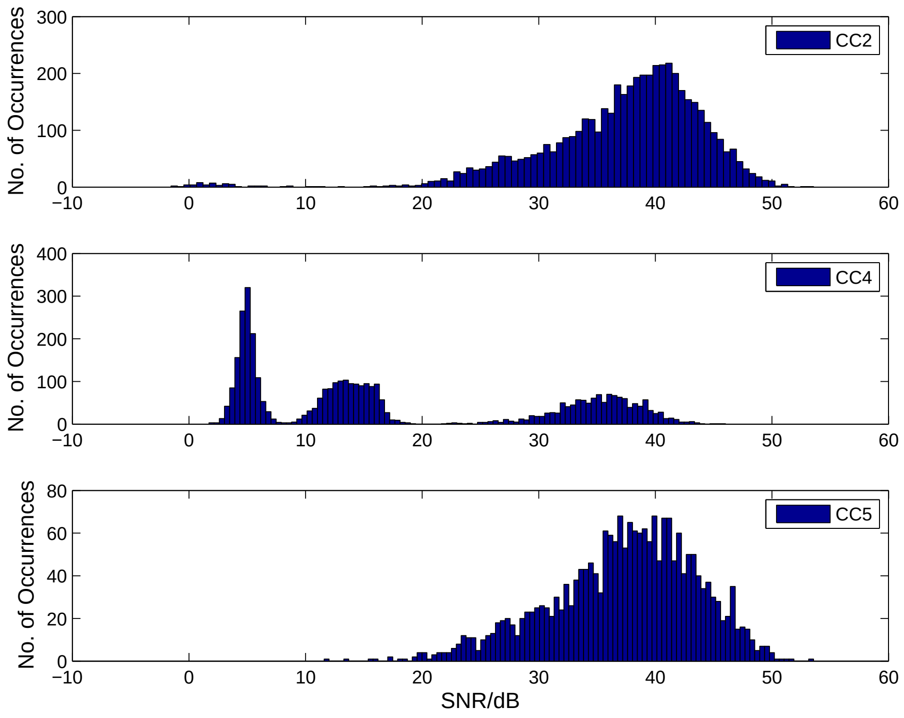


Figure 8.14: SNR distribution of test utterances in CC2, CC4, and CC5 of NIST'12.

8.3.4 Comparison of Scoring Methods

Table 8.16 shows the performance of different scoring methods. The methods in Table 8.16 can be categorized into i-vector avg. (Eq. 6.14), score avg. (Eq. 6.16), and PLDA–SVM scoring with empirical kernel maps (Eq. 6.25). Because Multi-Condition Training I achieves the best performance, the results reported in this subsection are based on this training method. The methods in Table 8.16 are named by the processes applied to the i-vectors for computing the verification scores. For example, *PLDA+UP-AVR+SVM-II* means that UP-AVR has been applied to create target-speaker i-vectors for training SVMs that use Empirical Kernel Map II (Eq. 6.27 and Eq. 6.24). Note that because some target speakers have one enrollment

Scoring Methods	EER (%)			MinNDCF(2012)		
	CC2	CC4	CC5	CC2	CC4	CC5
PLDA (I-vector Avg.)	2.40	3.53	2.99	0.328	0.335	0.307
PLDA (Score Avg.)	2.40	3.24	2.72	0.333	0.333	0.313
PLDA+UP-AVR (Score Avg.)	2.43	3.21	2.67	0.327	0.334	0.315
PLDA+UP-AVR+SVM-I	**1.84**	**2.83**	**2.34**	0.311	**0.293**	0.284
PLDA+SVM-II	2.12	3.06	2.74	0.304	0.307	0.270
PLDA+UP-AVR+SVM-II	1.94	2.85	2.34	**0.300**	0.295	**0.260**

Table 8.16: Performance of scoring methods for the multiple enrollment utterances in NIST'12 under the common conditions that involve telephone recordings.

utterance only, it is impossible to apply empirical kernel map I without UP-AVR. Therefore, no results for PLDA+SVM-I are reported. The RBF parameter γ was fixed to 1500 for all SVM-based scoring methods in Table 8.16.

Table 8.16 shows that the performance of score averaging is slightly better than the performance of i-vector averaging and that the performance of PLDA–SVM scoring is much better than that of score averaging and i-vector averaging. This demonstrates the advantage of incorporating discriminative information in the PLDA score space. Figure 8.15 shows the DET curves of the scoring methods. The curves further demonstrate the excellent performance of PLDA–SVM scoring.

Table 8.16 and Figure 8.15 also show that scoring methods PLDA+UP-AVR+SVM-I and PLDA+UP-AVR+SVM-II achieve a similar performance, which agrees with the conclusion in Figure 8.12 and Figure 8.13 that the target speaker part of the PLDA score vectors plays more important role for constructing the empirical kernel maps than the non-target speaker part. It suggests that using the Empirical Kernel Map I is sufficient for improving the performance of PLDA scoring. Another advantage of Empirical Kernel Map I is that it produces score vectors with a lower dimension.

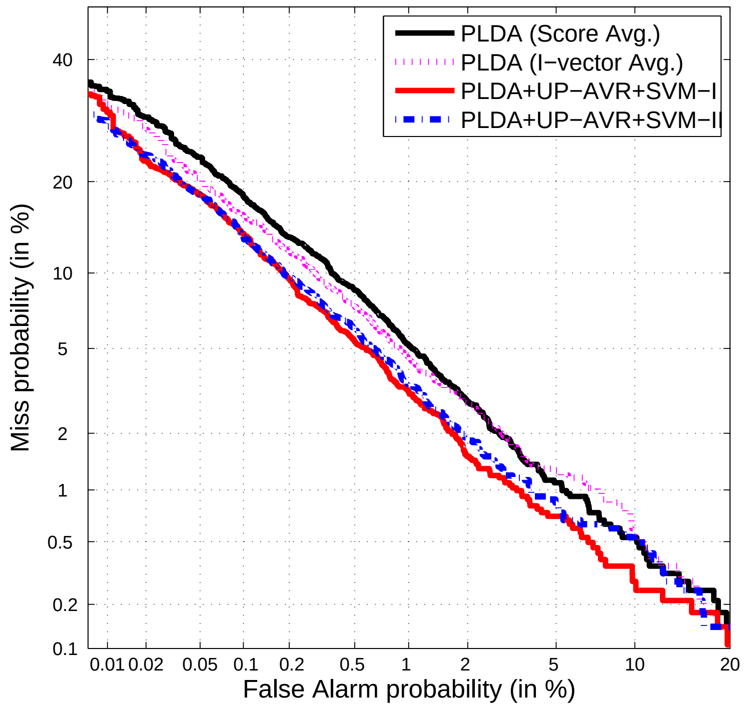


Figure 8.15: The DET performance of PLDA (I-vector Avg.), PLDA (Score Avg.), and PLDA–SVM scoring under CC2 in NIST'12. See Chapter 8.3.4 for the nomenclature of methods in the legend.

8.3.5 Importance of UP-AVR for SVM and LR Scoring

Figure 8.16 and Table 8.16 show that UP-AVR is very important for SVM scoring. After applying UP-AVR, the performance of SVMs scoring improves significantly and is much better than PLDA scoring. UP-AVR not only helps to alleviate the data-imbalance problem in SVM training, but also enriches the information content of the empirical scoring vectors by increasing the number of LR scores derived from the target speaker. However, UP-AVR is not beneficial to LR scoring, as evident by the performance of *PLDA (Score Avg.)* and *PLDA+UP-AVR (Score Avg.)* in Table 8.16.

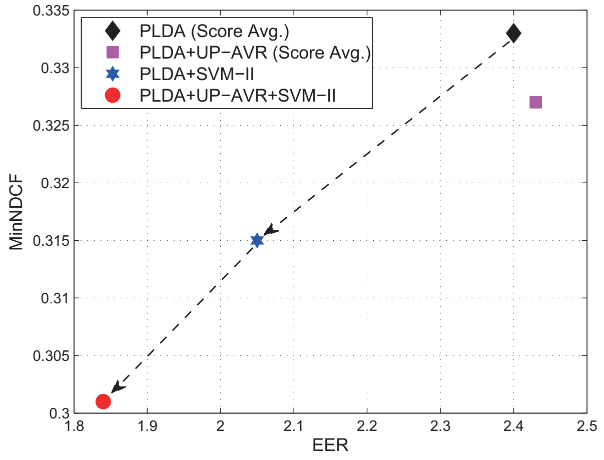
8.3.6 Property of Empirical Kernel Maps in SVM and RVM

The experiments on RVM were based on the Matlab code from Tipping [129]. For investigating the property of empirical kernel maps in SVM and RVM, we extracted 108 target speakers with true-target trials and imposter trials from NIST'12. Using these trials, the equal error rates (EERs) achieved by the SVMs and RVMs and their corresponding number of support vectors (SVs) and relevant vectors (RVs) were averaged across the 108 speakers. An RBF kernel $\mathbb{K}(\mathbf{x}, \mathbf{y}) = \exp(-\frac{\|\mathbf{x}-\mathbf{y}\|^2}{2\gamma^2})$ was adopted, where the RBF parameters γ was varied from 500 to 2500.[8]

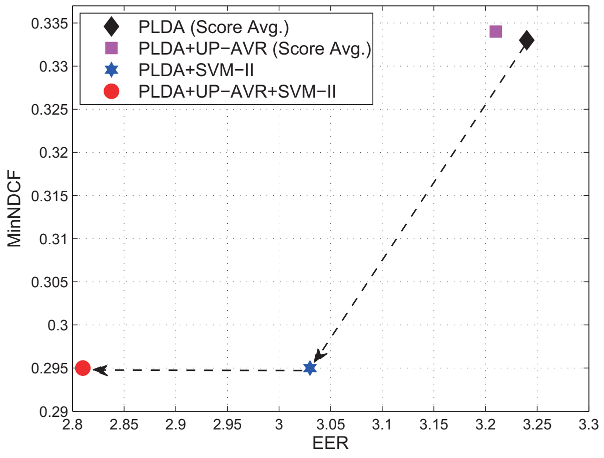
The top panel of Figure 8.17 plots the average EER against the average number of support vectors and relevance vectors in the SVMs and RVMs. It clearly shows that when the number of SVs increases, the performance of SVMs becomes poor. On the other hand, while the performance of RVMs is poor when the number of RVs is very small, their performance is fairly stable and is better than that of the SVMs once the number of relevance vectors is sufficient.

The middle panel of Figure 8.17 shows that when the RBF parameter γ increases, the number of SVs decreases first and then gradually increases. On the other hand, the number of RVs monotonically decreases when γ increases. More importantly, for a wide range of γ, there are more RVs than SVs, suggesting that for this dataset, the RVMs (under the regression mode) is less sparse than the SVMs. This phenomenon is attributed to the fact that the structural risk minimization attempts to find a small number of SVs that lie on the margin or the wrong side of it, whereas the

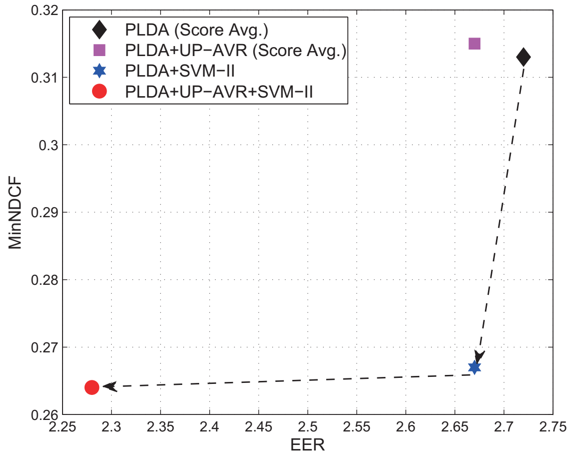
[8]Because the LR scores have range between -579.5 to 199.8 and the dimension of the LDA-projected i-vector is 150, a large value of γ is necessary.



(a) CC2



(b) CC4



(c) CC5

Figure 8.16: Minimum Norm DCF (2012) versus EER demonstrating the performance improvement obtained by the utterance partitioning approach in CC2, CC4, and CC5.

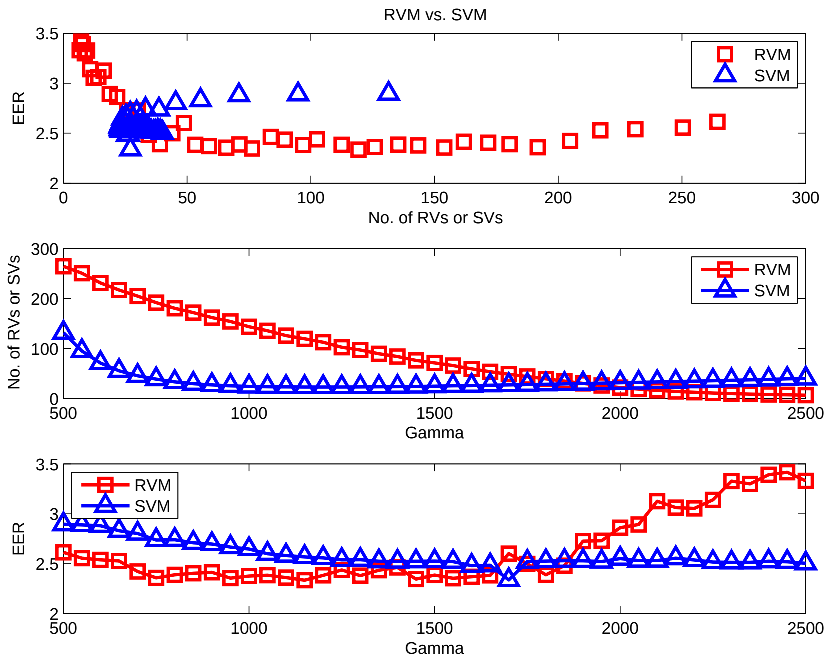


Figure 8.17: The property of empirical kernel maps in SVMs and RVM regressions. Gamma is the RBF parameter γ.

Bayesian relevance learning attempts to find a set of weights that maximize the likelihood in Eq. 6.29.

The middle and bottom panels of Figure 8.17 suggest that there is a lower limit on the number of RVs for the RVMs to be effective. In our experiments, this limit is around 50. Below this value, the performance of RVMs deteriorates rapidly and becomes significantly inferior to the SVMs. However, once the RVMs have sufficient RVs, their performance can be better than that of the SVMs.

8.3.7 Optimization of RBF Parameter Gamma

Because the performance of RVMs depends on RBF parameter γ, an optimization procedure was developed to find an appropriate γ for each target-speaker's RVM.

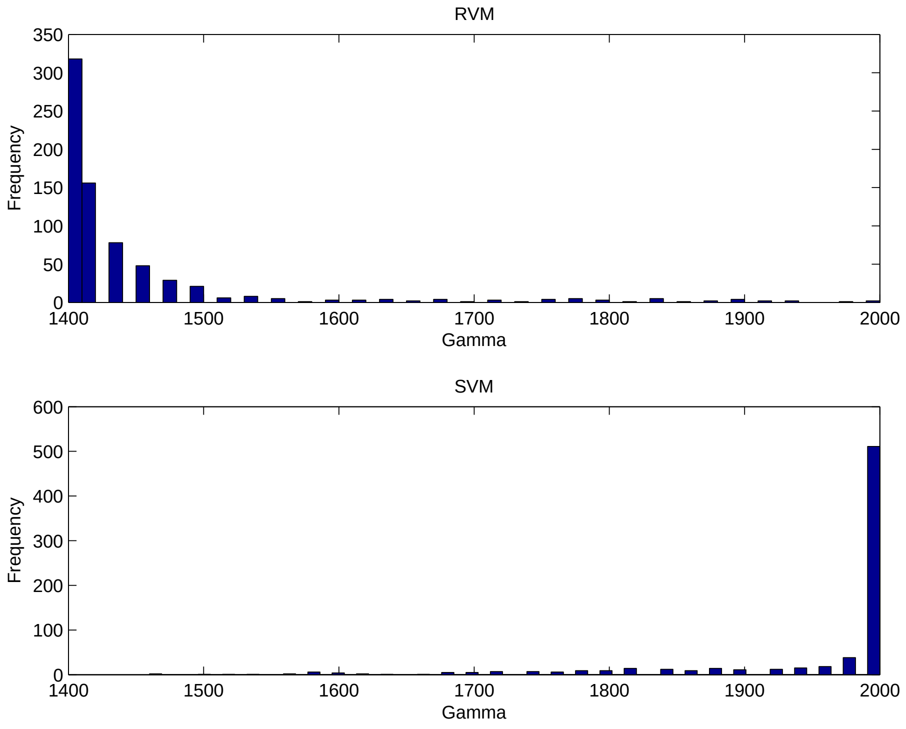


Figure 8.18: Histograms of the RBF parameter γ in target-speakers' RVMs (top panel) and SVMs (bottom panel).

Specifically, for each target speaker, a development set was created by applying utterance partitioning [5] on his/her enrollment utterances to generate a number of enrollment i-vectors. Then, some of these i-vectors were used for training an RVM. The remaining i-vectors were considered as true-target trials and the utterances of background speakers were considered as impostor trials. During RVM training, the value of γ was varied from 1400 to 2000 and the true-speaker scores and impostor scores were computed. The procedure continues until the difference between the mean of the true-target scores and the mean of the impostor scores is maximum.

Figure 8.18 shows the histograms of RBF parameter γ in the target-speakers' RVMs and SVMs. It shows that the preferred value of γ for RVMs is between 1400 and 1500 and that for SVMs is between 1900 and 2000. These ranges of values also agree with those in Figure 8.17.

8.3.8 PLDA-SVM vs. PLDA-RVM

Method	EER (%)	MinNDCF(2012)
PLDA	2.40	0.33
PLDA+UP-AVR	2.32	0.32
PLDA+SVM-II	2.07	0.31
PLDA+RVM-C-II	3.76	0.48
PLDA+RVM-R-II	2.32	0.28
PLDA+UP-AVR+SVM-II	1.97	0.30
PLDA+UP-AVR+RVM-C-II	3.00	0.42
PLDA+UP-AVR+RVM-R-II	**1.94**	**0.28**

Table 8.17: Performance comparison between SVM and RVM in common condition 2 of NIST'12. *RVM-C* represents relevance vector machine classification. *RVM-R* represents relevance vector machine regression. *UP-AVR* represents utterance partitioning with acoustic vector resampling [5]. The methods are named by the processes applied to the i-vectors for computing the verification scores. For example, *PLDA+UP-AVR+SVM-II* means that UP-AVR has been applied to create target-speaker i-vectors for training PLDA-SVMs (Eq. 6.27) that use empirical kernel Map II (Eq. 6.24).

Table 8.17 shows that the performance of PLDA-RVM classification with empirical kernel maps is poor and even worse than that of the baseline (Gaussian PLDA). The poor performance is caused by the severe sparsity of the RVM classification models. They are so sparse that the average number of relevance vectors per RVM is only two. In other words, each class only contains one relevance vector. Furthermore, RVM classification applies a logistic link function to compute the probabilistic outputs (posterior probabilities of the target-speaker class). While probabilistic outputs are desirable when the classification task involves one RVM only, in NIST SRE, we have one RVM per target speaker and the performance indexes (EER, minDCF, and DET) are based on the scores of all true-speaker trials and impostor attempts. This will lead to two skewed score- distributions with modes close to 1 and 0 for true-speaker trials and impostor attempts, respectively. Although these skewed distribution do not hurt the performance of SRE, we only apply the logistic sigmoid function during the training of RVM classifiers and dropped the function during scoring so that the score distribution of RVM classification is consistent with that of other methods. More precisely, Eq. 6.28 was used for computing the verification scores in the classification mode of RVMs and SVMs in our experiments.

Table 8.17 also shows that adopting empirical kernel maps in both SVM classification and RVM regression can improve performance. In addition, without performing UP-AVR, the performance of PLDA-RVM regression is comparable with PLDA-SVM. However, after performing the UP-AVR, the performance of both PLDA-RVM regression and PLDA-SVM improves and RVM regression slightly outperforms SVM. This results also agrees with the conclusion in Figure 8.17 that the performance of RVM regression will be better than the performance of SVM once the number of relevance vectors is sufficient.

The different performance of “PLDA+SVM-II” and “PLDA+UP-AVR+SVM-II” in Table 8.16 and Table 8.17 is caused by the different ways for using the RBF paramter γ. In Table 8.16, we fixed the speaker-independent RBF parameter for all SVM-based scoring methods. But in Table 8.17, the development data was used to optimize the speaker-dependent RBF parameter.

Chapter 9

CONCLUSIONS AND FUTURE WORK

9.1 Conclusions

This thesis has proposed several approaches to overcome the limitations of the state-of-the-art speaker verification systems.

For GMM-SVM speaker verification systems, one unaddressed issue is the severe imbalance between the numbers of speaker-class utterances and impostor-class utterances available for training a speaker-dependent SVM. This thesis has proposed an approach – utterance partitioning with acoustic vector resampling (UP–AVR) – to increase the number of target-speaker's supervectors in GMM-SVM speaker verification and demonstrated that a useful set of speaker-class supervectors can be generated by randomizing the sequence order of acoustic vectors in an enrollment utterance, followed by partitioning the randomized acoustic vectors. Evaluations show that the generated supervectors can alleviate the data imbalance problem and help the SVM learning algorithm to find better decision boundaries, thereby improving the verification performance. It was also found that GMM-SVM systems with UP–AVR are competitive with and sometimes superior to the state-of-the-art speaker comparison systems. The proposed resampling technique has important implications to practical implementation of speaker verification systems because it reduces the number of enrollment utterances and thereby reducing the burden and time users spent on speech recording.

For i-vector speaker verification systems, representing the i-vectors in a low-dimensional space has opened up opportunity for using machine learning techniques such as LDA, WCCN and PLDA to suppress session- and channel-variability. The key idea is to estimate the channel variability from training data and to project the target and test i-vectors to a subspace with minimal channel variability. While these techniques have achieved state-of-the-art performance in recent NIST Speaker Recognition Evaluations, they require multiple training speakers each providing sufficient numbers of sessions to train the transformation matrices. When the number of training speakers and/or number of recording sessions per speaker are insufficient, numerical difficulty or error will occur when estimating the transformation matrices, resulting in inferior performance.

This thesis has proposed two methods to solve this numerical difficulty. In the first method, UP–AVR is applied to i-vector speaker verification. Because a lot more i-vectors can be produced per training utterance, numerical difficulty arising from limited training sessions can be avoided. Our experimental results show that even if each training speaker has two recording sessions only, utterance partitioning can help to find more robust LDA and WCCN transformation matrices, leading to significant improvement in verification performance. In the second method, multi-way LDA is proposed to project the training i-vectors to a lower dimension space prior to compute the within-speaker scatter matrix. With the reduction in the dimension of i-vectors, the rank requirement of LDA and WCCN can be reduced to a comfortable level for reliable estimation of the LDA and WCCN transformation matrices. Four techniques aiming to alleviate the small sample-size problem in estimating the LDA and WCCN projection matrices in i-vector based speaker verification have been compared. It was found that UP–AVR is the most effective way to alleviate the small smaple size problem, followed by Multi-way LDA and PCA+LDA.

For i-vector/PLDA speaker verification systems, PLDA scoring only uses the information of background speakers implicitly. This thesis takes the advantage of empirical kernel maps to construct discriminative kernels for SVM/RVM scoring under the i-vector based framework and applies it to PLDA scoring for better utilizing multiple enrollment i-vectors and background speaker information. The results demonstrate that through empirical kernel maps, the discriminative information of same-speaker and different-speaker i-vector pairs can be captured in both the empirical feature space and the SVM/RVM weights.

9.2 Future Work

We plan to extend our current work in three fronts:

1. In spite of superior performance of PLDA-SVM scoring, it has one disadvantage. Compared with PLDA scoring, PLDA-SVM scoring requires a lot more computation. Further work is therefore necessary to reduce the number of nontarget speakers (B' in Eq. 6.24) for constructing the PLDA score space.

2. The idea of combining RVM with PLDA can be further explored in future work. For example, it is interesting to exploit the property that the kernel function used in RVM do not need to fulfill the Mercer's condition.

3. Gaussian PLDA with uncertainty propagation is effective for i-vector based speaker verification [130, 131]. The idea is to propagate the uncertainty of i-vectors caused by the duration variability of utterances to the PLDA model. However, a limitation of the method is the difficulty of performing length normalization on the posterior covariance matrix of an i-vector. It is interesting

to investigate methods to avoid performing length normalization on i-vectors in Gaussian PLDA modeling so that uncertainty propagation can be directly applied without transforming the posterior covariance matrices of i-vectors.

Appendix A

NOTATION

Vectors and matrices

x, X	a scalar is denoted by either italic small or capital letter
$\mathbf{x}$	a vector is denoted by a bold small letter
$\mathbf{x}^{\mathsf{T}}$	transpose of vector $\mathbf{x}$
$\Vert\mathbf{x}\Vert$	norm of vector $\mathbf{x}$
$\mathbf{1}$	column vector whose elements are all equal to one
$\mathbf{0}$	column vector whose elements are all equal to zero
$\boldsymbol{\mu}$	mean vector
$\mathbf{A}$	a matrix is denoted by a bold capital letter
$\mathbf{A}^{\mathsf{T}}$	transpose of matrix $\mathbf{A}$
$\mathbf{A}^{-1}$	inverse of matrix $\mathbf{A}$
$\vert\mathbf{A}\vert$	determinant of matrix $\mathbf{A}$
$\Vert\mathbf{A}\Vert$	norm of matrix $\mathbf{A}$
$\mathrm{diag}(\alpha_0, \alpha_1, ..., \alpha_N)$	matrix whose diagonal elements are $\alpha_0, \alpha_1, ..., \alpha_N$ and off-diagonal elements are zero
$\mathrm{Tr}(\mathbf{A})$	trace of matrix $\mathbf{A}$
$\boldsymbol{\Sigma}$	covariance matrix
$\mathbf{I}$	identity matrix

Variables, Symbols and Operations

$\equiv$	definition to
$\log_{10}(x)$	logarithm base 10 of x
$\exp(x)$	exponential of x
$\langle\cdot,\cdot\rangle$	Inner product of two vectors
$\mathrm{Cov}(\cdot,\cdot)$	covariance between two random variables
$\underset{x}{\mathrm{argmax}} f(x)$	the value of x that maximises the value of $f(x)$
$\int f(x)dx$	the integral of $f(x)$
$\sum_{i=1}^{M} x_i$	summation from 1 to M: $x_1 + x_2 + \ldots + x_M$
$\prod_{i=1}^{M} x_i$	product from 1 to M: $x_1 \times x_2 \times \ldots \times x_M$

Probability and distributions

$p(\cdot)$	probability density function
$p(\cdot\|\cdot)$	conditional probability density
$\Pr(\cdot)$	posterior probability
$P(\cdot)$	probability
$\mathcal{N}(\boldsymbol{\mu}, \boldsymbol{\Sigma})$	Gaussian distribution with mean $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$

Sets

$\mathcal{U}, \mathcal{X}, \ldots$	the sets or lists are generally denoted by Calligraphic font.
$\|\mathcal{X}\|$	the cardinality of set $\mathcal{X}$

BIBLIOGRAPHY

[1] W. M. Campbell, D. E. Sturim, D. A. Reynolds, and A. Solomonoff, "SVM based speaker verification using a GMM supervector kernel and NAP variability compensation," in *Proc. ICASSP 2006*, Toulouse, France, May 2006, vol. 1, pp. 97–100.

[2] L. Gillick and S. J. Cox, "Some statistical issues in the comparison of speech recognition algorithms," in *Proc. ICASSP 1989*, Glasgow, UK, May 1989, vol. 1, pp. 532–535.

[3] R. M. Bolle, N. K. Ratha, and S. Pankanti, "Evaluating authentication systems using bootstrap confidence intervals," in *Proc. AutoID'99*, 1999, pp. 9–13.

[4] R. M. Bolle, *Guide to biometrics*, Springer, New York, 2004.

[5] W. Rao and M. W. Mak, "Boosting the performance of i-vector based speaker verification via utterance partitioning," *IEEE Trans. on Audio, Speech and Language Processing*, vol. 21, no. 5, pp. 1012 – 1022, 2013.

[6] F. Bimbot, J. F. Bonastre, C. Fredouille, G. Gravier, I. Magrin-Chagnolleau, S. Meignier, T. Merlin, J. Ortega-Garcia, D. Petrovska-Delacretaz, and D. A. Reynolds, "A tutorial on text-independent speaker verification," *EURASIP Journal on Applied Signal Processing*, vol. 2004, no. 4, pp. 430–451, 2004.

[7] 2012 NIST Speaker Recognition Evaluation, "http://www.nist.gov/itl/iad/mig/sre12.cfm," 2012.

[8] T. Kinnunena and H. Z. Li, "An overview of text-independent speaker recognition: From features to supervectors," *Speech Communication*, vol. 52, no. 1, pp. 12–40, 2010.

[9] H. Beigi, *New Trends and Developments in Biometrics*, InTech, 2012.

[10] W. M. Campbell, D. E. Sturim, and D. A. Reynolds, "Support vector machines using GMM supervectors for speaker verification," *IEEE Signal Processing Letters*, vol. 13, no. 5, pp. 308–311, May 2006.

[11] N. Dehak, P. Kenny, R. Dehak, P. Dumouchel, and P. Ouellet, "Front-end factor analysis for speaker verification," *IEEE Trans. on Audio, Speech, and Language Processing*, vol. 19, no. 4, pp. 788–798, May 2011.

[12] P. Kenny, "Bayesian speaker verification with heavy-tailed priors," in *Proc. of Odyssey: Speaker and Language Recognition Workshop*, Brno, Czech Republic, Jun. 2010.

[13] D. Garcia-Romero and C. Y. Espy-Wilson, "Analysis of i-vector length normalization in speaker recognition systems," in *Proc. of Interspeech 2011*, Florence, Italy, Aug. 2011, pp. 249–252.

[14] S.J.D. Prince and J.H. Elder, "Probabilistic linear discriminant analysis for inferences about identity," in *Proc. of 11th International Conference on Computer Vision*, Rio de Janeiro, Brazil, Oct. 2007, pp. 1–8.

[15] M. W. Mak and W. Rao, "Utterance partitioning with acoustic vector resampling for GMM-SVM speaker verification," *Speech Communication*, vol. 53, no. 1, pp. 119–130, Jan. 2011.

[16] C. M. Bishop, *Pattern recognition and machine learning*, Springer New York, 2006.

[17] A. Hatch, S. Kajarekar, and A. Stolcke, "Within-class covariance normalization for SVM-based speaker recognition," in *Proc. of the 9th International Conference on Spoken Language Processing*, Pittsburgh, PA, USA, Sep. 2006, pp. 1471–1474.

[18] J. Ye, "Characterization of a family of algorithms for generalized discriminant analysis on undersampled problems," *Journal of Machine Learning Research*, vol. 6, no. 1, pp. 483–502, 2005.

[19] L. F. Chen, H. Y. M. Liao, M. T. Ko, J. C. Lin, and G. J. Yu, "A new LDA-based face recognition system which can solve the small sample size problem," *Pattern Recognition*, vol. 33, pp. 1713–1726, 2000.

[20] 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, Jan. 2000.

[21] W. Rao and M. W. Mak, "Addressing the data-imbalance problem in kernel-based speaker verification via utterance partitioning and speaker comparison," in *Proc. of Interspeech 2011*, Florence, Aug. 2011, pp. 2717–2720.

[22] W. Rao and M. W. Mak, “Alleviating the small sample-size problem in i-vector based speaker verification.,” in *Proc. Int. Sym. on Chinese Spoken Language Processing (ISCSLP 2012)*, Hong Kong, Dec. 2012, pp. 335–339.

[23] M.W. Mak and W. Rao, “Likelihood-ratio empirical kernels for i-vector based PLDA-SVM scoring,” in *Proc. ICASSP 2013*, Vancouver, Canada, May 2013, pp. 7702–7706.

[24] W. Rao and M. W. Mak, “Construction of discriminative kernels from known and unknown non-targets for PLDA-SVM scoring,” in *Proc. ICASSP 2014*, Florence, Italy, May 2014.

[25] W. Rao and M.W. Mak, “Relevance vector machines with empirical likelihood-ratio kernels for PLDA speaker verification,” in *Proc. Int. Sym. on Chinese Spoken Language Processing (ISCSLP 2014)*, Singapore, Sep. 2014.

[26] B. Scholkopf, S. Mika, C. J. C. Burges, P. Knirsch, K. Muller, G. Ratsch, and A.J. Smola, “Input space versus feature space in kernel-based methods,” *IEEE Trans. on Neural Networks*, vol. 10, no. 5, pp. 1000–1017, Sep. 1999.

[27] H. Xiong, M.N.S Swamy, and M.O. Ahmad, “Optimizing the kernel in the empirical feature space,” *IEEE Trans. on Neural Networks*, vol. 16, no. 2, pp. 460–474, 2005.

[28] S. X. Zhang and M. W. Mak, “Optimized Discriminative Kernel for SVM Scoring and Its Application to Speaker Verification,” *IEEE Trans. on Neural Networks*, vol. 22, no. 2, pp. 173–185, 2011.

[29] S. Y. Kung, M. W. Mak, and S. H. Lin, *Biometric Authentication: A Machine Learning Approach*, Prentice Hall, Upper Saddle River, New Jersey, 2005.

[30] R. Auckenthaler, M. Carey, and H. Lloyd-Thomas, “Score normalization for text-independent speaker verification systems,” *Digital Signal Processing*, vol. 10, no. 1–3, pp. 42–54, Jan. 2000.

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

[32] G. Doddington, M. A. Przybocki, A. F. Martin, and D.A. Reynolds, “The NIST speaker recognition evaluation–overview, methodology, systems, results, perspective,” *Speech Communication*, vol. 31, no. 2, pp. 225–254, 2000.

[33] S. S. Stevens and J. Volkmann, “The relation of pitch to frequency: A revised scale,” *The American Journal of Psychology*, pp. 329–353, 1940.

[34] Douglas O’shaughnessy, *Speech communication: human and machine*, Universities press, 1987.

[35] D. Reynolds and R. Rose, “Robust text-independent speaker identification using gaussian mixture speaker models,” *IEEE Trans. on Audio, Speech, and Language Processing*, vol. 3, no. 1, pp. 72–83, 1995.

[36] G. J. McLachlan and K. E. Basford, Eds., *Mixture Models: Inference and Applications to Clustering*, Marcel Dekker, 1988.

[37] G. J. McLachlan and D. Peel, Eds., *Finite Mixture Models*, Wiley, 2000.

[38] A. Dempster, N. Laird, and D. Rubin, "Maximum likelihood from incomplete data via the EM algorithm," *Journal of the Royal Statistical Society. Series B (Methodological)*, pp. 1–38, 1977.

[39] Y. LeCun and F. J. Huang, "Loss functions for discriminative training of energy-based models," in *Proc. of the 10-th International Workshop on Artificial Intelligence and Statistics (AIStats'05)*, 2005.

[40] S. Shum, N. Dehak, R. Dehak, and J. R. Glass, "Unsupervised speaker adaptation based on the cosine similarity for text-independent speaker verification," in *Proc. Odyssey 2010*, Brno, Czech Republic, Jul. 2010, pp. 76–82.

[41] D. A. Reynolds, "Comparison of background normalization methods for text-independent speaker verification.," in *Proc. Eurospeech 1997*, Rhodes, 1997, pp. 963–966.

[42] R. Vogt, B. Baker, and S. Sridharan, "Modelling session variability in text-independent speaker verification," in *Proc. Interspeech 2005*, Spain, Sep. 2005.

[43] A. Martin, G. Doddington, T. Kamm, M. Ordowski, and M. Przybocki, "The DET curve in assessment of detection task performance," in *Proc. Eurospeech'97*, 1997, pp. 1895–1898.

[44] W. M. Campbell, "Generalized linear discriminant sequence kernels for speaker recognition," in *Proc. ICASSP 2002*, Orlando, USA, May 2002, vol. 1, pp. 161–164.

[45] V.Wan and S. Renals, "Speaker verification using sequence discriminant support vector machines," *IEEE Transactions on Speech and Audio Processing*, vol. 13, no. 2, pp. 203–210, Mar. 2005.

[46] W. M. Campbell and Z. N. Karam, "Simple and efficient speaker comparison using approximate KL divergence," in *Proc. Interspeech 2010*, Japan, 2010.

[47] C. H. You, K. A. Lee, and H. Z. Li, "GMM-SVM kernel with a Bhattacharyya-based distance for speaker recognition," *IEEE Trans on Audio, Speech and language Processing*, vol. 18, no. 6, pp. 1300–1312, 2010.

[48] A. Solomonoff, W. M. Campbell, and I. Boardman, "Advances in channel compensation for SVM speaker recognition," in *Proc. of ICASSP'05*, 2005, pp. 629–632.

[49] G. Wu and E. Y. Chang, "KBA: Kernel boundary alignment considering imbalanced data distribution," *IEEE Transactions on Knowledge and Data Engineering*, vol. 17, no. 6, pp. 786–795, 2005.

[50] Y. Tang, Y.Q. Zhang, N.V. Chawla, and S. Krasser, "SVMs modeling for highly imbalanced classification," *IEEE Trans. on System, Man, and Cybernetics, Part B*, vol. 39, no. 1, pp. 281–288, Feb 2009.

[51] W. M. Campbell, Z. Karam, and D. E. Sturim, "Speaker comparison with inner product discriminant functions," in *Advances in Neural Information Processing Systems 22*, 2009, pp. 207–215.

[52] P. Kenny, G. Boulianne, P. Ouellet, and P. Dumouchel, "Joint factor analysis

versus eigenchannels in speaker recognition," *IEEE Trans. on Audio, Speech and Language Processing*, vol. 15, no. 4, pp. 1435–1447, May 2007.

[53] P. Kenny, P. Ouellet, N. Dehak, V. Gupta, and P. Dumouchel, "A study of inter-speaker variability in speaker verification," *IEEE Trans. on Audio, Speech and Language Processing*, vol. 16, no. 5, pp. 980–988, 2008.

[54] P. Kenny, "A small foot-print i-vector extractor," in *Proc. Odyssey 2012*, 2012.

[55] G. Saon, H. Soltau, D. Nahamoo, and M. Picheny, "Speaker adaptation of neural network acoustic models using i-vectors," in *2013 IEEE Workshop on Automatic Speech Recognition and Understanding (ASRU)*, Olomouc, Dec. 2013, pp. 55–59.

[56] S. J. D. Prince, *Computer vision: models, learning and inference*, Cambridge University Press, 2012.

[57] N. Dehak, R. Dehak, P. Kenny, N. Brummer, P. Ouellet, and P. Dumouchel, "Support vector machines versus fast scoring in the low-dimensional total variability space for speaker verification," in *Proc. Interspeech 2009*, Sep. 2009, pp. 1559–1562.

[58] N. Japkowicz and S. Stephen, "The class imbalance problem: A systematic study," *Intelligent Data Analysis*, vol. 6, no. 5, pp. 429–449, Oct. 2002.

[59] Y. M. Sun, A. K. C. Wong, and M. S. Kamel, "Classification of imbalanced data: A review," *International Journal of Pattern Recognition and Artificial Intelligence*, vol. 23, no. 4, pp. 687–719, 2009.

[60] A. Napolitano J. V. Hulse, T. M. Khoshgoftaar, "Experimental perspectives on learning from imbalanced data," in *Proc. of the 24 th International Conference on Machine Learning*, Corvallis, Oregon, USA, 2007, pp. 935–942.

[61] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, "SMOTE: Synthetic minority over-sampling technique," *Artificial Intelligence and Research*, vol. 16, pp. 321–357, 2002.

[62] N. V. Chawla, A. Lazarevic, L. O. Hall, and K. W. Bowyer, "SMOTEBoost: Improving prediction of the minority class in boosting," in *Proc. of the 7th European Conference on Principles and Practice of Knowledge Discovery in Databases*, 2003, pp. 107–119.

[63] H. Han, W. Y. Wang, and B. H. Mao, "Borderline-SMOTE: A new over-sampling method in imbalanced data sets learning," in *Proc. of International Conference on Intelligent Computing. Lecture Notes in Computer Science 3644*, Hefei, China, Aug. 2005, pp. 878–887.

[64] A. Nickerson, N. Japkowicz, and E. Millos, "Using unsupervised learning to guide resampling in imbalanced data sets," in *Proc. of the 8th International Workshop on AI and Statistics*, 2001, pp. 261–265.

[65] R. C. Holte, L. E. Acker, and B. W. Porter, "Concept learning and the problem of small disjuncts," in *Proc. of the 11th Joint International Conference on Artificial Intelligence*, 1989, pp. 813–818.

[66] T. Jo and N. Japkowicz, "Class imbalances versus small disjuncts," *SIGKDD Explorations*, vol. 16, pp. 40–49, Jun. 2004.

[67] P. Kang and S. Cho, "EUS SVMs: Ensemble of under-sampled SVMs for data imbalance problems," in *ICONIP'06*, I. King, et al., Ed., 2006, vol. LNCS 4232, pp. 837–846.

[68] Z. Y. Lin, Z. F. Hao, X. W. Yang, and X. L. Liu, "Several SVM ensemble methods integrated with under-sampling for imbalanced data learning," in *Advanced Data Mining and Applications*. 2009, vol. 5678/2009 of *LNCS*, pp. 536–544, Springer.

[69] R. Akbani, S. Kwek, and N. Japkowicz, "Applying support vector machines to imbalanced datasets," in *Proc. of ECML'04*, Pisa, Italy, Sep. 2004, pp. 39–50.

[70] A. Sun, E. P. Lim, B. Benatallah, and M. Hassan, "FISA: Feature-based instance selection for imbalanced text classification," in *Proc. of PAKDD'06*, Singapore, 2006, pp. 250–254.

[71] M. McLaren, R. Vogt, B. Baker, and S. Sridharan, "Data-driven background dataset selection for SVM-based speaker verification," *IEEE Transaction on audio, speech, and language processing*, vol. 18, no. 6, pp. 1496–1506, Aug. 2010.

[72] Y. Liu, X. H. Yu, J. X. J. Huang, and A. An, "Combining integrated sampling with svm ensembles for learning from imbalanced datasets," *Information Processing and Management*, vol. 47, pp. 617–631, Jul. 2011.

[73] S. Ertekin, J. Huang, L. Bottou, and C. L. Giles, "Learning on the border: Active learning in imbalanced data classification," in *Proc. of CIKM'07*, Lisboa, Portugal, Nov. 2007, pp. 127–136.

[74] A. Sun, E. P. Lim, and Y. Liu, "On strategies for imbalanced text classification using SVM: A comparative study," *Decision Support Systems*, vol. 48, no. 1, pp. 191–201, 2009.

[75] K. Veropoulos, C. Campbell, and N. Cristianini, "Controlling the sensitivity of support vector machines," in *Proc. Int. Joint Conf. Artificial Intelligence*, 1999, pp. 55–60.

[76] Y. Lin, Y. Y. Lee, and G. Wahba, "Support vector machines for classification in nonstandard situations," *Machine Learning*, vol. 46, no. 1-3, pp. 191–202, 2002.

[77] X. Hong, S. Chen, and C. Harris, "A kernel-based two-class classifier for imbalanced data sets," *IEEE Transactions on Neural Networks*, vol. 18, no. 1, pp. 28–41, 2007.

[78] A. Fakeri-Tabrizi, S. Tollari, N. Usunier, and P. Gallinari, "Improving image annotation in imbalanced classification problems with ranking svm," in *Multilingual Information Access Evaluation Vol. II Multimedia Experiments: Proceedings of the 10th Workshop of the CrossCLanguage Evaluation Forum (CLEF 2009)*, 2010, pp. 291–294.

[79] J. P. Hwang, S. K. Park, and E. Kim, "A new weighted approach to imbalanced data classification problem via support vector machine with quadratic cost function," *Expert Systems with Applications*, pp. 8580–8585, 2011.

[80] T. R. Golub, D. K. Slonim, P. Tamayo, C. Huard, M. Gaasenbeek, J. P. Mesirov, H. Coller, M. Loh, J. R. Downing, M. A. Caligiuri, C. D. Bloomfield,

and E. S. Lander, "Molecular classification of cancer: Class discovery and class prediction by gene expression monitoring," *Science*, vol. 286, pp. 531–537, 1999.

[81] M. A. Turk and A. P. Pentland, "Face recognition using eigenfaces," in *Proc. of the Computer Society Conference on Computer Vision and Pattern Recognition*, 1991, pp. 586–591.

[82] P. N. Belhumeur, J. P. Hespanha, and D. J. Kriegman, "Eigenfaces vs. Fisherfaces: Recognition using class specific linear projection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 19, no. 7, pp. 711–720, 1997.

[83] W. Zhao, R. Chellappa, and P. Phillips, "Subspace linear discriminant analysis for face recognition," *Technical Report CAR-TR-914*, 1999.

[84] K. Fukunaga, *Introduction to Statistical Pattern Classification*, Academic Press, USA, 1990.

[85] S. Raudys and R. P. W. Duin, "On expected classification error of the Fisher linear classifier with pseudo-inverse covariance matrix," *Pattern Recognition Letters*, vol. 19, pp. 385–392, 1998.

[86] J. H. Friedman, "Regularized discriminant analysis," *Journal of the American Statistical Association*, vol. 84, pp. 165–175, 1989.

[87] T. Hastie, A. Buja, and R. Tibshirani, "Penalized discriminant analysis," *Annals of Statistics*, vol. 23, pp. 73–102, 1995.

[88] R. Huang, Q. Liu, H. Lu, and S. Ma, "Solving the small sample size problem of LDA," in *Proc. of International Conference on Pattern Recognition*, 2002, pp. 29–32.

[89] J. Ye and T. Xiong, "Computational and theoretical analysis of null space and orthogonal linear discriminant analysis," *The Journal of Machine Learning Research*, vol. 7, pp. 1183–1204, 2006.

[90] B. Efron and G. Gong, "A leisurely look at bootstrap, the jackknife, and cross-validation," *The American Statistician*, vol. 37, no. 1, pp. 36–48, 1983.

[91] B. Fauve, N. Evans, and J. Mason, "Improving the performance of text-independent short duration SVM- and GMM-based speaker verification," in *Odyssey 2008*, 2008.

[92] H.B. Yu and M. W. Mak, "Comparison of voice activity detectors for interview speech in NIST speaker recognition evaluation," in *Proc. of Interspeech 2011*, Florence, Aug. 2011, pp. 2353–2356.

[93] M. E. Tipping, "Sparse bayesian learning and the relevance vector machine," *Journal of Machine Learning Research*, vol. 1, pp. 211–244, 2001.

[94] K. A. Lee, A. Larcher, C. H. You, B. Ma, and H. Z. Li, "Multi-session PLDA scoring of i-vector for partially open-set speaker detection," in *Proc. Interspeech 2013*, Lyon, France, Aug. 2013, pp. 3651–3655.

[95] P. Rajan, A. Afanasyev, V. Hautamki, and T. Kinnunen, "From single to multiple enrollment i-vectors: Practical PLDA scoring variants for speaker verification," *Digital Signal Processing*, vol. 31, pp. 93–101, 2014.

[96] W. B. Liu, Z. D. Yu, and M. Li, "An iterative framework for unsupervised learning in the PLDA based speaker verification," in *Proc. Int. Sym. on Chinese Spoken Language Processing (ISCSLP 2014)*, Singapore, Sep. 2014.

[97] R. Saeidi, K. A. Lee, T. Kinnunen, T. Hasan, B. Fauve, P. M. Bousquet, E. Khoury, P. L. Sordo Martinez, J. M. K. Kua, C. H. You, et al., "I4U submission to NIST SRE 2012: A large-scale collaborative effort for noise-robust speaker verification," in *Proc. Interspeech 2013*, Lyon, France, Aug. 2013, pp. 1986–1990.

[98] P. Rajan, T. Kinnunen, and V. Hautamaki, "Effect of multicondition training on i-vector PLDA configurations for speaker recognition," in *Proc. of Interspeech 2013*, Lyon, France, Aug. 2013.

[99] A. Solomonoff, C. Quillen, and W. M. Campbell, "Speaker verification using support vector machines and high-level features," *IEEE Transactions On Audio, Speech, and Language Processing*, vol. 15, no. 7, Sep. 2007.

[100] S. Cumani, N. Brummer, L. Burget, and P. Laface, "Fast discriminative speaker verification in the i-vector space," in *Proc. ICASSP 2011*, Prague, May 2011, pp. 4852–4855.

[101] L. Burget, O. Plchot, S. Cumani, O. Glembek, P. Matejka, and N. Briimmer, "Discriminatively trained probabilistic linear discriminant analysis for speaker verification," in *Proc. ICASSP 2011*, Prague, May 2011, pp. 4832–4835.

[102] S. Cumani and P. Laface, "Analysis of large-scale SVM training algorithms

for language and speaker recognition," *IEEE Trans. on Audio, Speech, and Language Processing*, vol. 20, pp. 1585–1596, Jul. 2012.

[103] S. Cumani, N. Brummer, L. Burget, P. Laface, O. Plchot, and V. Vasilakakis, "Pairwise discriminative speaker verification in the-vector space," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 21, no. 6, pp. 1217–1227, Sep. 2013.

[104] S. Cumani, O. Glembek, N. Brummer, E. de Villiers, and P. Laface, "Gender independent discriminative speaker recognition in i-vector space," in *Proc. ICASSP 2012*, Kyoto, Japan, Mar. 2012, pp. 4361–4364.

[105] S. Yaman and J. Pelecanos, "Using polynomial kernel support vector machines for speaker verification," *Signal Processing Letters*, vol. 20, no. 9, pp. 901–904, Sep. 2013.

[106] D. Sturim, D. A. Reynolds, E. Singer, and J. P. Campbell, "Speaker indexing in large audio databases using anchor models," in *Proc. ICASSP 2001*, Salt Lake City, UT, May 2001, pp. 429–432.

[107] M. Collet, D. Charlet, and F. Bimbot, "Speaker tracking by anchor models using speaker segment cluster information," in *Proc. ICASSP 2006*, Toulouse, France, May 2006, pp. 1009–1012.

[108] E. Noor and H. Aronowitz, "Efficient language identification using anchor models and support vector machines," in *Odyssey 2006*, San Juan, June 2006, pp. 1–6.

[109] N. Brummer, "LLR transformation for SRE'12," in *Notes relevant to the NIST 2012 SRE*. Dec. 2012, Online: https://sites.google.com/site/bosaristoolkit/sre12.

[110] D. A.van Leeuwen and R. Saeidi, "Knowing the non-target speakers: The effect of the i-vector population for PLDA training in speaker recognition," in *Proc. ICASSP 2013*, Vancouver, BC, Canada, May 2013, pp. 6778–6782.

[111] T. Hasan, R. Saeidi, J. H. L. Hansen, and D. A. van Leeuwen, "Duration mismatch compensation for i-vector based speaker recognition system," in *Proc. ICASSP 2013*, Vancouver, BC, Canada, May 2013, pp. 7663–7667.

[112] H. W. Sun, K. A. Lee, and B. Ma, "Anti-model KL-SVM-NAP system for NIST SRE 2012 evaluation," in *Proc. ICASSP 2013*, Vancouver, BC, Canada, May 2013, pp. 7688–7692.

[113] J. Gonzalez-Rodriguez, "Evaluating automatic speaker recognition systems: An overview of the NIST speaker recognition evaluations (1996-2014)," *Loquens*, vol. 1, no. 1, 2014.

[114] 2002 NIST Speaker Recognition Evaluation, "http://www.itl.nist.gov/iad/mig//tests/sre/2002/index.html," 2002.

[115] 2004 NIST Speaker Recognition Evaluation, "http://www.itl.nist.gov/iad/mig//tests/sre/2004/index.html," 2004.

[116] 2010 NIST Speaker Recognition Evaluation, "http://www.itl.nist.gov/iad/mig//tests/sre/2010/," 2010.

[117] Freesound, “http://www.freesound.org/,” Apr. 2005.

[118] M. W. Mak and H. B. Yu, “Robust voice activity detection for interview speech in nist speaker recognition evaluation,” in *Proc. of APSIPA ASC 2010*, Singapore, 2010.

[119] B. S. Atal, “Effectiveness of linear prediction characteristics of the speech wave for automatic speaker identification and verification,” *J. Acoust. Soc. Am.*, vol. 55, no. 6, pp. 1304–1312, Jun. 1974.

[120] J. Pelecanos and S. Sridharan, “Feature warping for robust speaker verification,” in *Proc. of Odyssey: Speaker and Language Recognition Workshop*, Crete, Greece, Jun. 2001, pp. 213–218.

[121] “Joint factor analysis matlab demo,” *http://speech.fit.vutbr.cz/software/joint-factor-analysis-matlab-demo.*

[122] Y. Lei, L. Burget, L. Ferrer, M. Graciarena, and N. Scheffer, “Towards noise-robust speaker recognition using probabilistic linear discriminant analysis,” in *Proc. ICASSP 2012*, Kyoto, Japan, March 2012, pp. 4253–4256.

[123] N. Dehak, P. Kenny, R. Dehak, O. Glembek, P. Dumouchel, L. Burget, V. Hubeika, and F. Castaldo, “Support vector machines and joint factor analysis for speaker verification,” in *ICASSP*, 2009, pp. 4237–4240.

[124] M. McLaren and D. van Leeuwen, “Source-normalised LDA for robust speaker recognition using i-vectors from multiple speech sources,” *IEEE Trans. on Audio, Speech, and Language Processing*, vol. 20, pp. 755–766, 2012.

[125] T. Hasan, S. O. Sadjadi, G. Liu, N. Shokouhi, H. Boril, and J. H. L. Hansen, "CRSS system for 2012 NIST speaker recogntion evaluation," in *Proc. ICASSP 2013*, Vancouver, Canada, May 2013, pp. 6783–6787.

[126] P. Rajan, T. Kinnunen, and V. Hautamaki, "Effect of multicondition training on i-vector PLDA configurations for speaker recognition," in *Proc. Interspeech 2013*, Lyon, France, Aug. 2013, pp. 3694–3697.

[127] M. W. Mak, "SNR-dependent mixture of plda for noise robust speaker verification," in *Proc. Interspeech 2014*, Singapore, Sep. 2014, pp. 1855–1859.

[128] X.M. Pang and M.W. Mak, "Fusion of SNR-dependent PLDA models for noise robust speaker verification," in *Proc. Int. Sym. on Chinese Spoken Language Processing (ISCSLP 2014)*, Singapore, Sep. 2014.

[129] "*http://www.miketipping.com/sparsebayes.htm*," .

[130] P. Kenny, T. Stafylakis, P. Ouellet, M. J. Alam, and P. Dumouchel, "PLDA for speaker verification with utterances of arbitrary duration," in *Proc. ICASSP 2013*, Vancouver, Canada, May 2013, pp. 7649–7653.

[131] T. Stafylakis, P. Kenny, P. Ouellet, J. Perez, M. Kockmann, and P. Dumouchel, "Text-dependent speaker recogntion using PLDA with uncertainty propagation," in *Proc. Interspeech 2013*, Lyon, France, Aug. 2013.