Universidad de Zaragoza

Departamento de Ingeniería Electrónica y Comunicaciones

Universidad Zaragoza

Tesis doctoral

Métodos discriminativos para la optimización de modelos en la Verificación del Hablante

Discriminative Methods for model optimization in Speaker Verification

Leibny Paola García Perera

Directores de Tesis

Juan Arturo Nolazco Flores, Eduardo Lleida Solano

Mayo 29, 2014

Any intelligent fool can make things bigger, more complex, and more violent.
It takes a touch of genius – and a lot of courage – to move in the opposite direction.

Albert Einstein

Acknowledgements

To every expression of love...

I am very thankful and fortunate to have been surrounded by amazing people during these years. My sincere and deepest thanks to each one of you.

Firstly, I would like to thank Doctor Juan Arturo Nolazco Flores for his support during all this time, specially through those tough years. Thank you for always having faith in me. Secondly, I would like to thank Eduardo Lleida Solano for his kindness and support every time I needed it. It has been an awesome and fruitful experience to work with both of you.

My first short stay collaboration was in Georgia Tech, where Professor Chin-Hui Lee and his research team welcomed me. I am very thankful to Professor Lee for spending time discussing my research, for sharing his knowledge and for his constant and sincere support during these years. I was very lucky to be in Gatech and learned about speech, being there helped me to foster my research skills.

During my stay in CMU, I had the opportunity to work with very smart and nice people that in many ways contributed to this work. First, I would like to thank Bhiksha Raj, for rescuing me when I was about to give up. Your help and advice marked my life. There is a before Bhiksha and after Bhiksha. Thank you Rita Singh for all your support, I learned kindness from you. You are an example to all of us. My acknowledgment extends to Richard Stern for the nice discussions and the academic and personal advice.

My gratitude goes to the people that anonymously make our work easier, taking care of the administrative procedures. I acknowledge the help of the secretaries in every place I have been, your help and patience to do the paperwork and to solve my administrative concerns is greatly appreciated. Special thanks to Sandra Dominguez Alanis, at the Tec de Monterrey.

Friendship has been a constant in my life. Usually, colleagues transform into lifelong friends that deal with you through the up and downs in research. They do deserve more than a thank you!

In Atlanta I had the opportunity to have very fruitful discussions and learn about different topics. Special thanks to Antonio Moreno for teaching me speech! Now I can not recognize what code is mine or yours. Thank you for the nice discussions and for sharing your knowledge with me and my team. I am thankful for showing me new ways of viewing problems. But most of all, for being there when I needed it. Thank you to all my friends in Atlanta, special thanks to the spouse group that gave me true friends while living in Atlanta.

Pittsburgh was a watershed in my research and personal life. First, I would like to thank Afsaneh Asaei, Affysita Preciosa, for her unconditional friendship and love. There are no words to express how much you mean to me. *Look! There is a miracle!* Apurv Tiwari, Angelito, thank you for letting me share my adventures with you. But, first, *listen... do you want to know a secret?* Akiko and Takayuki Arakawa, thank you for being around all the time. Your awesomeness and friendship always give me peace. Antonio Juarez, thank you for existing. I always need an outlier in my life that can show me that beyond the horizon there are places full of beauty. Thank you for keeping in touch, when the probability of doing so was close to zero.

Thank you Oliver Walter for sharing your time with me. You have this strange ability to give balance and peace to my surreal world. Thank you for having faith when I get lost. Life made us meet, and it has been awesome to have you around.

Tec de Monterrey has been my *home place* during the last decade. My gratitude goes to the students that have been in our lab. Special thanks to Igmar Hernández for the very nice friendship during these years. There are no words to express my gratitude. Thank you Daniel Escobar for being a true friend, for being part of my crazy and abnormal world over and over again, and virtually staying in touch no matter what. Thank you Benjamin Martínez for showing me other ways of thinking, and that research is not everything. I am very thankful to Roberto Aceves for his patience and his support with the cluster along this time. It would have been almost impossible to succeed in this without your help. I extend my gratitude to Carlos M for the wonderful moments we shared and for encouraging me to study the PhD.

Zaragoza University gave me the opportunity not only to study a PhD, but to have awesome colleagues and friends. Thank you Oscar Saz for scolding me every time I need it (still now). You have been an awesome friend during these years. Special thanks to Jesús Villalba and Rubén Cabrejas, I really enjoyed spending time with you, dancing and parting. You are just fun! Thank you Diego Castán, David Martínez and Julia Olcoz, I enjoyed having coffee with you, our discussions and all your help.

Special thanks to my unconditional friends along this time: Jeanette Russ, Gloria Massieu, Mayra González, Adriana López, Raúl Moreno, Enrique Coeto, Juan José Salas "Fuku". You are always in my thoughts, even when I am far away.

Lastly, I would like to thank my mother, Paula Perera, my father, Guillermo García and my brother Guillermo García-Perera for being always beside me along this journey. Thank you mom, for your prayers, your immense love, your tears, but most of all for your laugh and your smile. When I am down, your laugh and happiness give me courage to continue with enthutiasm and faith. There is not other phrase to say than: *"yo soy la hija de Paulita y con eso es suficiente."* Thank you Dad, for believing, for your immense love and for showing me that the word impossible is not in our dictionary. Your advices, examples, prayers and love are always with me. Thank you Memo, for everything, you are the best! Thank you for always being there, in the good, in the bad and in the very very bad. You deserve a statue somewhere. Your support and advice always make the difference in my life. Thank you grandma, wherever you are... your strength is always guiding my steps.

And for those not mentioned here, all my gratitude!!!

Thank you "music" and "poetry" just for being... If love is everything, I am sure art is the sweetest part of it.

This work has been financially supported by the National Council on Science and Technology of Mexico (CONACYT) through scholarship 160892

Abstract

The increasing need for secure authentication systems has motivated recent interest in effective algorithms for Speaker Verification (SV). Such need for high-performance algorithms, capable of delivering low error rates, has opened various research branches.

In this research, we propose to investigate, from a discriminative point of view, a set of methodologies to improve the performance of the actual state of the art of the Speaker Verification systems. In a first approach, we investigate the optimization of the training hyper-parameters to explicitly consider the tradeoff between false rejections and false acceptances. This objective can be achieved by *maximizing the area under the Receiver Operating Characteristic (ROC) curve*. We believe that the enhancement of the parameters should not be limited to a single operating point, and that a more robust strategy is to optimize the parameters according to the maximization of the *area under the curve* (AUC). We study how to optimize the parameters using a mathematical characterization of the area under the ROC curve based on the Wilcoxon Mann Whitney (WMW) statistic, and the solution using the generalized probabilistic descent (GPD) algorithm. Moreover, we analyze the effect and improvements in the performance metrics such as *detection error tradeoff curve* (DET), the *equal error rate* (EER), and the *minimum value of the detection cost function* (minDCF).

On a second approach, we investigate the speech signal as a combination of attributes that contain information of the speaker, channel and noise. Conventional speaker verification systems train a single generic model for all cases, and handle all variations of these attributes either by factor analysis, or by not considering the variations explicitly. We propose a new methodology to *partition* the *data space* according to these attributes and train separate models for each partition. The partitions may be obtained according to any chosen attribute. In this research, we will show how to effectively train the

models for each partition in a discriminative way to maximize the separation among them.

Moreover, the design of algorithms –robust to noisy conditions– plays a key role that allows SV systems to operate successfully in real conditions. We propose to extend our current methodologies to alleviate the effect of noise in such conditions. For our first approach, in a situation where noise is present, the point of operation may not be a single point, or it may be shifted in an unpredictable manner. We will show how our methodology of maximizing the area under the ROC curve can be more robust than the conventional classifiers even when the noise effect is not explicitly considered. Besides, the noise may be present at different signal to noise ratios (SNRs) that can degrade the performance of the system. Hence, we consider an effective decomposition of the speech signals that can handle the different signal attributes such as SNR, noise and channel type. We believe that instead of addressing the problem with just a unified model, a decomposition that partitions the signal space based on special attributes can provide better estimation. Those attributes can represent different channel or noisy conditions.

We have analyzed the potential of these methodologies that can improve the performance of the state of the art systems by reducing not only the error, but also by controlling the operating points and alleviate the noise effects.

Resumen

La creciente necesidad de sistemas de autenticación seguros ha motivado el interés de algorítmos efectivos de Verificación de Hablante (VH). Dicha necesidad de algorítmos de alto rendimiento, capaces de obtener tasas de error bajas, ha abierto varias ramas de investigación.

En este trabajo proponemos investigar, desde un punto de vista discriminativo, un conjunto de metodologías para mejorar el desempeño del estado del arte de los sistemas de VH. En un primer enfoque investigamos la optimización de los hiper-parámetros para explícitamente considerar el compromiso entre los errores de falsa aceptación y falso rechazo. El objetivo de la optimización se puede lograr *maximizando el área bajo la curva* conocida como ROC (Receiver Operating Characteristic) por sus siglas en inglés. Creemos que esta optimización de los parámetros no debe de estar limitada solo a un punto de operación y una estrategía más robusta es optimizar los parámetros para incrementar el *área bajo la curva*, AUC (Area Under the Curve por sus siglas en inglés) de modo que todos los puntos sean maximizados. Estudiaremos cómo optimizar los parámetros utilizando la representación matemática del área bajo la curva ROC basada en la estadística de Wilcoxon Mann Whitney (WMW) y el cálculo adecuado empleando el algorítmo de descendente probabilístico generalizado. Además, analizamos el efecto y mejoras en métricas como la curva *detection error tradeoff* (DET), el error conocido como *Equal Error Rate* (EER) y el valor mínimo de la función de detección de costo, *minimum value of the detection cost function* (minDCF) todos ellos por sus siglas en inglés.

En un segundo enfoque, investigamos la señal de voz como una combinación de atributos que contienen información del hablante, del canal y el ruido. Los sistemas de verificación convencionales entrenan modelos únicos genéricos para todos los casos, y manejan las variaciones de estos atributos ya sea usando análisis de factores o no

considerando esas variaciones de manera explícita. Proponemos una nueva metodología para *particionar el espacio de los datos* de acuerdo a estas carcterísticas y entrenar modelos por separado para cada partición. Las particiones se pueden obtener de acuerdo a cada atributo. En esta investigación mostraremos como entrenar efectivamente los modelos de manera discriminativa para maximizar la separación entre ellos.

Además, el diseño de algorítimos robustos a las condiciones de ruido juegan un papel clave que permite a los sistemas de VH operar en condiciones reales. Proponemos extender nuestras metodologías para mitigar los efectos del ruido en esas condiciones. Para nuestro primer enfoque, en una situación donde el ruido se encuentre presente, el punto de operación puede no ser solo un punto, o puede existir un corrimiento de forma impredecible. Mostraremos como nuestra metodología de maximización del área bajo la curva ROC es más robusta que la usada por clasificadores convencionales incluso cuando el ruido no está explícitamente considerado. Además, podemos encontrar ruido a diferentes relación señal a ruido (SNR) que puede degradar el desempeño del sistema. Así, es factible considerar una descomposición eficiente de las señales de voz que tome en cuenta los diferentes atributos como son SNR, el ruido y el tipo de canal. Consideramos que en lugar de abordar el problema con un modelo unificado, una descomposición en particiones del espacio de características basado en atributos especiales puede proporcionar mejores resultados. Esos atributos pueden representar diferentes canales y condiciones de ruido.

Hemos analizado el potencial de estas metodologías que permiten mejorar el desempeño del estado del arte de los sistemas reduciendo el error, y por otra parte controlar los puntos de operación y mitigar los efectos del ruido.

Contents

List of Figures

List of Tables

List of Acronyms

AUC Area Under the Curve

ASR Automatic Speech Recognition

CMS Cepstral Mean Subtraction

CMV Cepstral Mean Variance

DCT Discrete Cosine Transform

EER Equal Error Rate

EM Expectation Maximization

FAR False Acceptance rate

FA Factor Analysis

FRR False Rejection Rate

GMM Gaussian Mixture Model

GPD Generalized Probabilistic Descent Algorithm

JFA Joint Factor Analysis

log Logarithm

MFCC Mel Frequency Cepstral Coefficients

minDCF Minimum Detection Cost Functions

ML Maximum Likelihood

MLLR Maximum Likelihood Linear Regression

MVE Minimum Verification

NIST National Institute of Standards and Technology

ROC Receiver Operating Characteristic

SNR Signal to Noise Ratio

SV Speaker Verification

SVM Support Vector Machine

UBM Universal Background Model

WMW Wilcoxon Mann Whitney

Chapter 1

Introduction

To see a world in a grain of sand,
And a heaven in a wild flower,
Hold infinity in the palm of your hand,
And eternity in an hour.

- William Blake *Auguries of Innocence*

Over the last decade, advances in communication technology have leaded the research community efforts to focus more and more on secure and remote transactions over the networks [4]. User authentication has attracted major attention because of the emphasis on security issues. The first idea was to use passwords and rely on the users that these passwords would not be shared and kept secure [5, 6]. However, passwords can be stolen, shared or even forgotten. The necessity of authentication systems that can obtain information from human characteristics (biometrics) was a good solution [6]. At first, research on biometrics concentrated on fingerprint, face recognition, iris recognition, DNA and speaker authentication, among others. Nowadays, the trend is to make a fusion of them to reach the best performance.

From the variety of biometric signals that can be used for authentication, speech shows several advantages. First, it is the most natural source of human communication; the users do not need any extra device to produce speech. Second, the acquisition of the signal can be performed using well-known non-sophisticated equipment like the telephone and internet network devices (common forms of information transfer). Moreover, speech opens the possibility to perform remote transactions over those networks. Lastly, the study in areas such as *speech recognition* and lately in *speaker recognition* had consolidated the understanding of speech.

Other more sophisticated applications include forensics, surveillance and speaker personalization, all of them related to information security. In forensics, the main idea is to

identify a suspect given a speech sample [7]. In surveillance, the networks are monitored so that a big amount of data from several speakers is available. The challenge is to locate a target speaker within that collection [8]. The newest devices, such as smart phones, rely also on speaker recognition in their applications. The speaker personalization of the software results in a more efficient operation of the device [9].

From all of them, authentication is the most representative example of speaker recognition. It has attracted major attention due to the continuous increase of users connected to the networks. Speaker recognition arose in the last three decades [10, 11] as a solution to provide security. Speaker recognition is usually divided into two subtopics: speaker identification and speaker verification. In speaker identification the target user is distinguished from a set of possible users. In speaker verification the identity of a user is accepted or rejected by analyzing a claimed identity and a speech phrase. Both fields share in essence the same principles.

From these two, we will study Speaker Verification (SV), which main objective is to accept or reject a prospect user with the lowest error. The speaker verification traditional schemes are based on statistical hypothesis testing, where we establish two hypothesis: a null hypothesis H_s (accepts the speaker as legitimate) and the alternative hypothesis $H_{\bar{s}}$ (rejects him or her) [12, 13]. Their relation is expressed as a ratio between likelihoods with respect to two models: target and imposter. The output ratio is then tagged as accepted or rejected user.

Two types of error commonly occur: *false acceptances* (FA) – incorrect decision due to accept a speaker who is not actually the target speaker, and *false rejections* (FR) – incorrect rejection of target speakers. Ideally, the probability of both types of errors must be zero; in practice, the two occur and are considered in the system design. A trade-off between them is established so that both errors are reduced. The probability of false acceptance can be reduced at the cost of increasing false rejection. Thus, for any given system, the operating point can be manipulated to obtain a desired ratio between false acceptances and false rejections. The entire range of possible operating points is characterized mainly by the *Operating Receiver Characteristic* (ROC) curve and *detection error tradeoff* (DET) curve.

This thesis shows how to reduce these errors using a discriminative optimization approach from two different perspectives. First, we present a complete strategy so that the reduction of every operating point at the ROC and the DET curve are improved. Moreover, we extend our findings to the noisy condition case. Second, we searched for more specific ways to describe the speaker space based on certain attribute. We built contiguous region models that characterized a specific attribute. These region models are an aid to construct more specific target models. The discriminative optimization enhances those models and improves the results of the state of the art architectures. This strategy is also extended to the noisy condition case. Finally, averaging both methodologies improves the current results.

1.1 Thesis Motivation and Scope

The success of traditional SV systems lies in computing the adequate models that can clearly classify a target speaker from an impostor. The more suited these models are to specific scenarios or target speakers, the best results we can obtain [12, 13, 14, 15]. The *maximum likelihood* (ML) approaches using generative modeling [12, 13] and lately factor analysis [14, 15] represented a huge improvement in the systems performance. Generative modeling relies on maximizing the likelihood of a model to a current set of data. However, reduction of the *false acceptance* and *false rejection* errors is not taken as a primary objective but as an effect of an accurate modeling of the target speakers.

Discriminative training approaches, first employed in speech recognition [16, 17, 18, 19], proposed an alternative solution that included the reduction of the error by considering imposter samples in the optimization of the models. Just a few studies are found in the bibliography [20, 21, 22, 23], probably because the need of high performance computation. But nowadays, with the advances in technology, the computation is not longer and issue and the experimentation is feasible.

One of the schemes that fit the SV theory is the minimum classification approach. Its main goal is to minimize the empirical classification error regardless the distribution of the data [16, 17, 24]. Hence, we considered it to be the backbone of our research and the motivation to extend its potential to SV. In this sense, the algorithm must focus on the error reduction. Moreover, the algorithm must be designed in a way that it can obtain competitive results as the ones produced by traditional approaches. Lastly, we observed that traditional methods usually measure the improvements on a single operating point, called *Equal Error Rate* (EER), that acts as a summary of the full range of possible outcomes. In our case, the motivation is to find a methodology to reduce the error at every operating point.

Finally, the challenge nowadays is how to make the systems robust enough to different kinds of scenarios (including different environments, and noisy conditions). Hence, the motivation of this study is to deal with those scenarios and build a robust system based on discriminative training.

1.2 Thesis Statement

This thesis develops a discriminative model optimization to improve the performance of traditional SV systems. Two perspectives were explored separately and finally merged to obtain a more robust architecture.

In a first study, we propose a discriminative training paradigm that *explicitly* learns the model parameters to optimize the *entire* ROC curve. The solution we propose is to maximize the AUC; this naturally also optimizes the performance at every operating point on the ROC

curve. Moreover, the effect is also observed in the performance metrics such as the DET curve and the minDCF. To obtain an analytical solution, we use the well known Wilcoxon Mann Whitney (WMW) statistic as an equivalent to the AUC [25, 26] quantification.

In a second study, we analyze the complexity of the SV task. Then we propose an alternative approach to consider an effective decomposition of the speech signal space. Instead of addressing the problem with just a unified model (one model that can represent all possible data), a decomposition that partitions the signal space based on special attributes of the training data provides a better estimation. This thesis proposes to break down a general model into sub-region specific models. Each submodel is trained discriminatively for specificity purposes. Hence, the new optimized models are more robust and specific to the region they represent (a set of sub-models can represent, for example, channel, noise and speaker main attributes). To obtain each target submodel, the system employs non-target examples which are *likely* to be confused with that target attribute. We showed that by using this procedure, the error rate is reduced. Both techniques were examined also under noisy conditions, improving the baseline systems.

Finally, we propose a hierarchical architecture that incorporates both approaches. At first the system generates an attribute tree that decides the sub-region attributes per level. Then, the models for each level are discriminatively optimized using the AUC under the ROC curve. This approach is also naturally extended to noisy conditions scenarios, where the specificity of the *optimized models* improve the performance of the system in terms of the complete DET curve, the EER and the minDCF.

1.3 Objectives and Methodology

This section describes the objectives of this thesis. The main objectives are stated in the scientific and development objectives. The former shows the scientific knowledge acquired by using the discriminative approach under different conditions. The latter include the technical infrastructure needed and the software developed for this purpose. Finally, a special section dedicated to the methodology shows the steps followed in the analysis of the problem.

1.3.1 Scientific objectives

The first general objective is to explore the state of the art of speaker verification systems. We analyze two branches: the training process to optimize the target speaker models and the performance of such systems under noisy conditions. The training of the models include several flavors: the generative, discriminative and hybrid modeling. From them, we study their outstanding components that lead to improvements in both clean and noisy conditions.

A second general objective is to propose a procedure to improve the current systems by using a discriminative methodology. The first approach is to show that by *optimizing the area under the ROC curve* the errors at any operating point can be reduced. This is achieved by proposing an analytical solution that includes the WMW statistic and the gradient descent. A second approach is to refine the sub-spaces of the entire speaker space in order to obtain a set of sub-region specific models. According to this approach, the objective is to demonstrate that by breaking up the speech signal into specific attributes – like channel, SNR or even special characteristics of a set of speakers – we can compute more accurate speaker models that result in a reduction of the errors.

A third objective is to tackle the challenge of speaker verification under noisy conditions with our previous methodologies. For our first approach, the goal is to demonstrate that optimizing the area under the ROC curve improves the performance even if noise is present. The same applies for our second method in which we can treat the noise as a special attribute and decompose the space according to specific signal to noise ratios.

1.3.2 Development objectives

The main objective is the construction of a system from scratch that can give us the expertise on each part and the possible improvements that may be applied, see Figure 1.1. The current system is composed of two main parts, enrollment and verification. The first step of both stages is to convert speech into a vector representation known as feature extraction. Then, in the enrollment, the system trains the models for the target speakers and for impostors. The verification tests those models using non-tagged speech. The score is the final numerical outcome. Depending on its value, the non-tagged speech is accepted or rejected.

At the beginning, each part of this scheme was built from scratch, afterwards, we also used software provided by other universities, and modified it to fit our requirements.[1]

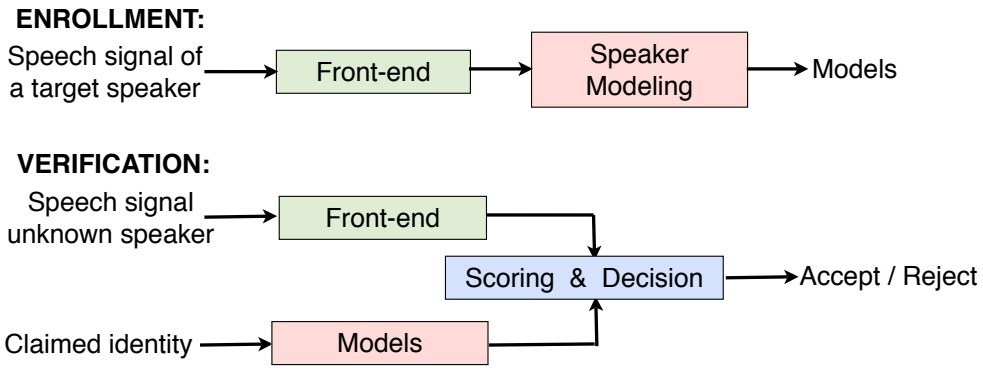


Figure 1.1: Simple Architecture of a Speaker Verification System

[1] JFA and i-vectors software provided by [27] and the voice activity detection [28]).

Our main contribution in this sense is to focus on the adequate speaker modeling. The objective is then to develop the discriminative AUC and ensemble optimization algorithms that can be included in further SV designs.

1.3.3 Methodological procedure

The methodology to follow is shown in Figure 1.2. The first step in this methodology is to explore the traditional modeling approaches in SV, so we can have the knowledge of the state of the art. The results at this stage give us a baseline system. Afterwards, we designed a discriminative optimization to be included in the current solutions: factor analysis and minimum verification error. From this optimization two branches emerged: the AUC under the ROC curve and the Ensemble approach. At first, we modified the baseline methods, Joint Factor Analysis (JFA) and the Minimum Verification Error (MVE), to follow either the AUC under the ROC curve or the Ensemble approach. Then we merged both ideas to fulfill the final goal of working under clean and noisy conditions.

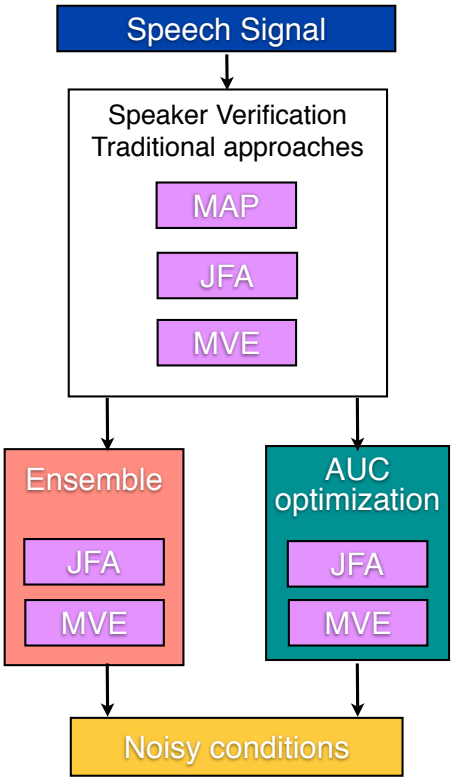


Figure 1.2: Scheme of the methodology followed by this thesis

1.4 Organization

In this section, we provide the organization of this thesis, so that the reader can choose the topic of interest. The first chapters detailed the state of the art systems. We emphasized the modeling and the optimization procedure. The next chapters describe and analyze our two approaches (Ensemble and AUC) and give the mathematical formalism of both of them. The final chapters show the results of our findings, give conclusions and open discussion for future research.

1.4.1 Classical Speaker Verification System

In this chapter, we present a brief introduction of the state of the art systems. We describe the components of the traditional SV systems and some of its variations. We describe the front-end, generative and discriminative modeling, the scoring and decision making.

1.4.2 Speaker Modeling

This chapter provides the intuition of how the modeling is performed in detail. First, we explain the *hypothesis testing theory* as the background of a simple classifier. We define what are the elements needed to have a functional classifier. Then, we describe in depth techniques such as maximum a posteriori, factor analysis and minimum verification error.

1.4.3 Robustness in Speaker Verification

This part is dedicated to algorithms that deal with noisy conditions. We first give some examples of the algorithms for speech recognition and then focus on Speaker Verification. Besides, the chapter shows how the distorted signal affects every part of the SV system.

1.4.4 Experimental Framework

In this chapter, we present the experimental setup used along this research. We give exact details of the databases employed and a brief description. Moreover, we show baseline results according to the setup and the classical approach.

1.4.5 Optimization of the *area under the ROC curve*

This chapter shows the first of our approaches which updates the model parameters by optimizing the area under the ROC curve. We give the mathematical formalism and we show how the method optimizes the parameters for each of the operating points along

the curve. The approach is also extended to noisy conditions. Finally, we present the improvements obtained with our method on real data.

1.4.6 Ensemble Modeling

In this chapter, we present the ensemble approach for speaker verification. We present the intuition behind the partition of the impostor space. Moreover, the clustering and the refinement of the models belonging to those new impostor spaces are explained. Once again, this approach can be extended to noisy conditions. Finally, this approach is applied to real databases and the improvements are presented.

1.4.7 AUC and Ensemble Integration

This chapter merges the AUC and the Ensemble approaches to produce a new methodology that can be used for both: clean and noisy conditions. In this section, we just provide the mathematical formulation of the merge of both approaches. Besides, the results and improvements on the NIST database are provided.

1.4.8 Discussion

This chapter depicts the results that support this thesis. We show the results for the AUC of the ROC curve in different databases such as MOBIO, Ahumada/Gaudi and YOHO. In the same way, we also present the results for the Ensemble and Integration approach for these databases.

1.4.9 Conclusions and Future Work

This chapter presents the conclusions of our work, emphasizing the important characteristics of our approaches. In the first case, we conclude that the AUC approach is more robust than just having a discriminative or generative approach alone. In the second case, we concluded that by segmenting the space into clusters the impostors space can be refined resulting in improvements.

1.4.10 Appendices

In this section, we include some extra tools needed to demonstrate the solution for several approaches. We cover the background for basic modeling, the most used methodologies for the factor analysis and minimum verification error approaches.

Chapter 2

Classical Speaker Verification Systems

> Me di cuenta de que tenía que revolucionar; aprender cosas nuevas para no quedarme atrás. Me di cuenta y me rebelé.
>
> - Jaime Sabines

This section presents an overview of the state of the art of the Speaker Verification systems. It details the traditional way to address SV by using a Gaussian Mixture Models (GMM) framework, joint factor analysis and i-vector or discriminative approaches like maximum mutual information estimation (MMIE), support vector machines (SVM), and minimum verification error (MVE).

As depicted in Figure 2.1, the SV systems have two main stages: enrollment (training) and verification (test). In the enrollment, the set of acoustic models are trained. In the verification, the speech trials evaluate the acoustic models and produce an accept/reject result. The first step of both stages is feature extraction. It converts the speech signal into a vectorial representation that contains specific information about the speaker.

In training, we can observe that there are several options of training algorithms available. Although they have the same final goal to obtain lower error rates, the approaches can address different aspects. We can divide them into generative, discriminative and hybrids. Generative modeling optimizes the probability density functions (pdf) so that they can describe the target speakers; examples of these algorithms are MAP and Factor analysis variants. Discriminative modeling usually minimizes the empirical error rate using samples of both the target speaker and impostors; examples of these approach are *maximum mutual information estimation*, MMIE, *support vector machines*, SVM, and MVE. Combinations of methodologies also occur; for example, for the i-vectors and the SVM approach. For the purpose of our research we embed a discriminative optimization into the current generative models so that the error rates are lowered. For the purpose of this research we will focus

only on FA, which is considered the state of the art, and MVE as examples of both possible scenarios.

In verification, the models already trained are evaluated using unknown speech, and a score (usually as likelihood ratio) is produced as an output. The score is then fed into a classifier with a predefined threshold and a decision (accept or reject a user) is made.

The SV systems, depending on the application, are divided into text-dependent and text-independent categories. The text-dependent systems require that the user utters the same set of words during the training and test phases. In some text-dependent cases, , the system can ask the speaker to utter a set of words at the test stage. Both examples have predefined acoustic models depending on the text and speaker. For the text-independent systems, the user can speak any set of words; sometimes even in other languages, making the task harder. Usually, the text-dependent systems can get better performances; and the text independent SV is still a challenge.

The next sections describe how the current systems are built based on generative modeling (MLE and MAP) approach, joint factor analysis and i-vector, and discriminative approaches like MVE.

2.1 Feature Extraction

The proper selection of relevant information contained in the voice signal is a fundamental step. Different types of feature extraction have been explored by the research community: spectral, prosodic and high-level. Spectral features, which we will address in the next sections, convey the acoustic information of the vocal tract [29, 30]. Prosodic features refer to the intonation and the stress that a person uses when speaking; they depend on several factors including, for example, emotional state and the situation the speaker is trying to communicate (sarcasm, sadness, among others) [31, 32]. The high-level features extract information about the manner in which a person speaks (the lexicon), the different topics she addresses, and the style in which she uses the words [31, 33].

In this research, we will focus on spectral feature extraction. The model used to represent the extraction process is based on a sound source – the larynx and a filter – the vocal tract [34]. The parametrizations that work under this scheme are commonly divided into filter bank (FB) cepstral analysis and linear predictive (LP) coding [35]. For the filter bank approach the most known techniques are Mel Frequency Cepstral Coefficients (MFCC) and Linear Frequency Cepstral Coefficients (LFCC). Both MFCC and LFCC features are subjected to the shape and the frequency of a filter bank. In the case of MFCC, the filter bank follows a logarithmic spacing at high frequencies [29]; LFCC adopts a linear spacing filter bank [36]. For linear predictive coding, the most representative techniques are Linear Prediction Cepstral Coefficients (LPCC) and Perceptual Linear Prediction (PLP). Both

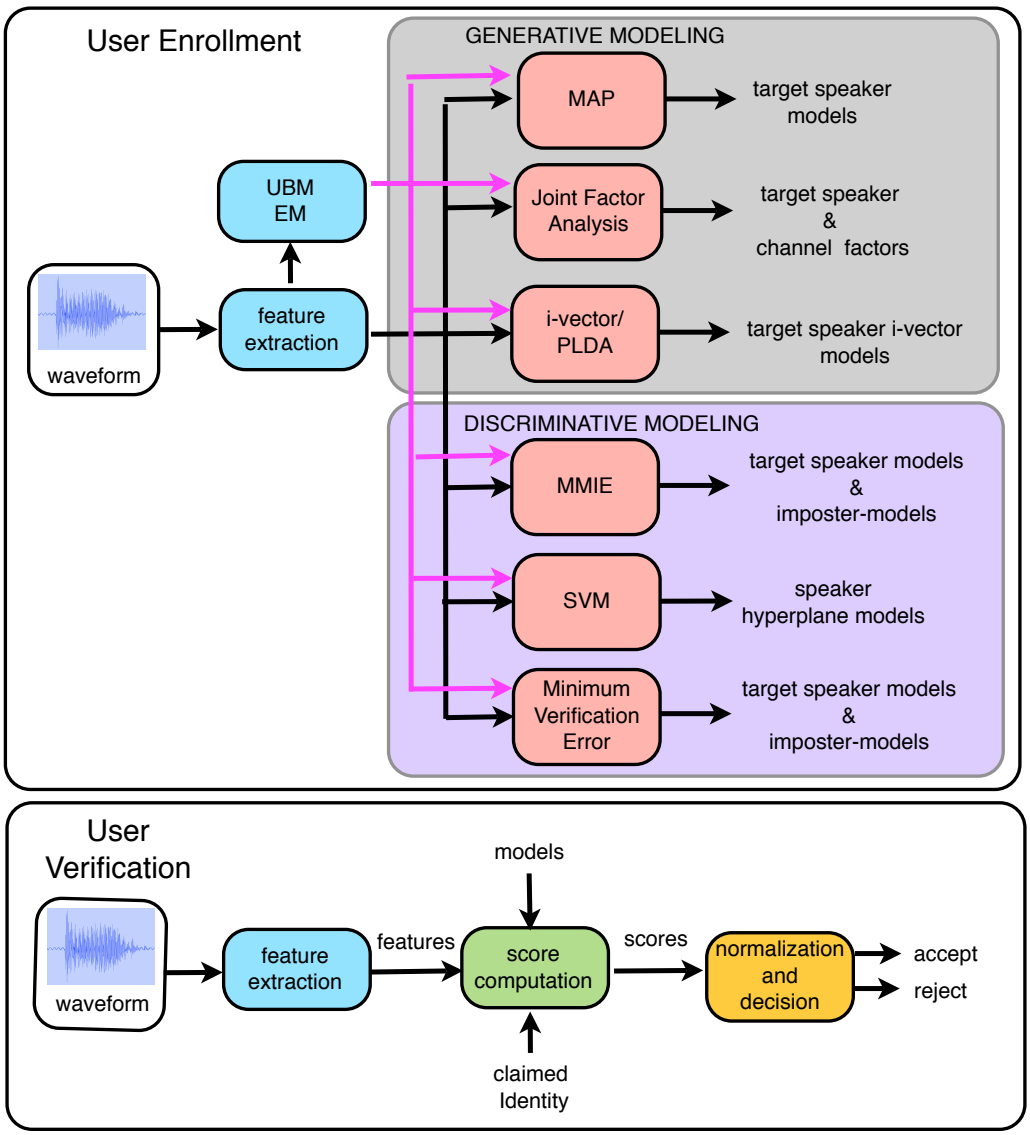


Figure 2.1: Speaker Verification architecture in detail, *The big picture*

employ a linear process that can predict the speech signal at each time given previous samples. The LPCC includes an all-pole model that can describe the spectrum with a smooth envelope [37]. Finally, the PLP uses psychoacoustic elements to model the speech using an all-pole model similar to LPC [38].

We will briefly describe the MFCC procedure; we consider it to be the most frequently employed by the research community.

2.1.1 MFCCs computation

The computation of the MFCCs is composed of several stages [29], as shown in Figure 2.2. The first stage is to pre-emphasize the speech signal and multiply by a series of overlapping hamming windows, producing the modified, signal, $S(i)$. The next step is to transform

$S(i)$ to its spectrum $S(w)$, employing short-time Fourier analysis . We can extract either the power or the magnitude of the Fourier coefficients, $S_m(w)$. Afterwards, a filterbank (FB) transforms this signal into a smooth spectrum representation (close to the envelope). The filterbank output is then converted to the log domain. Finally, we apply the DCT to decorrelate and produce the cepstral coefficients [30]. If the filterbank is linearly spaced, the resulting coefficients are named Linear Frequency Cepstral Coefficients. However, the most common used are the MFCCs. They follow the mel scale that resembles the way a person hears. To emphasize the dynamic features of the speech in time, the time-derivative (Δ) and the time-acceleration (Δ^2) are usually computed. It is common to compute 12 MFCCs, one Energy coefficient and its corresponding (Δ) and (Δ^2). However, recent studies have shown that using more than 12 MFCCs can give better results [39, 27]. [1]

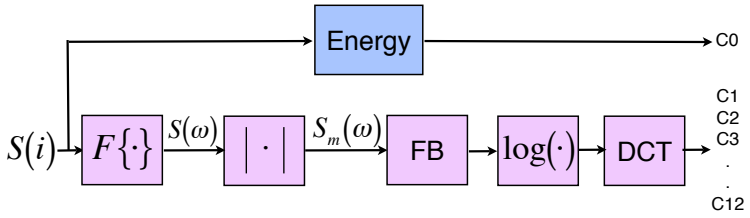


Figure 2.2: Mel Frequency Cepstral Coefficient Front End

2.1.2 Feature Normalization

Normalizations at this stage are implemented to reduce the effects of the noise and the channel distortion. For example, cepstral mean subtraction (CMS) [40] is a blind deconvolution that consists of the subtraction of the utterance mean of the cepstral coefficients from each feature. In the same way, variance normalization (CVN) [41] is also applied. Hence, the normalized features will be described by a zero mean and variance one distribution. Another well-known feature normalization is RASTA (Relative Spectra) [42]. While CMS focus on the stationary convolution of the noise due to the channel, RASTA reduces the effect of the varying channel, it removes low and high modulation frequencies. The three of them are commonly used in SV architectures.

2.1.2.1 Feature Warping

A special normalization for SV at the feature stage is feature warping. It belongs to the Gaussianization methods [43, 44]. The underlying concept in this normalization scheme is that every spectral attribute (cepstral coefficient in our case) is normally distributed across

[1] The research in [39, 27] show that from 16 to 19 MFCCs, the performance is better than with the usual 12 MFCCs.

time, but the transmission channel distorts such distribution. The task of feature warping is to undo the distortion caused by the channel by warping each attribute's scale so that the resulting attribute set has a normal distribution. Feature warping is accomplished by first assembling an empirical CDF (cumulative distribution function) from the ranked features after and before the current frame, and then performing the CDF-inverse at the current frame.

2.1.2.2 Feature Frame Removal

Frame removal is based on the idea that low energy frames do not provide information about the identity of a person. The log-energy frames of each utterance are modeled by a three-component GMM. w_1 corresponds to the highest weight of the rightmost Gaussian, w_2 to the middle Gaussian, and w_3 to the leftmost Gaussian. According to this model every frame log-energy is labelled as high if it belongs to the rightmost Gaussian; medium if it belongs to the middle Gaussian; and low if it belongs to the leftmost Gaussian [12]. The following Equation is used to determine which frames can be extracted.

$$N = w_1 + (g \times \alpha \times w_2), \tag{2.1}$$

where g is a value between 0 and 1, and α is an heuristic weighting parameter. N is the percentage (between 0 and 100) of the frames with highest energy that will be extracted. If an accurate voice activity detector (VAD) or the speech transcriptions to perform speech recognition are not available, frame removal is a suitable solution.

2.2 Speaker Modeling

After obtaining the feature vectors, the aim is to design an algorithm capable to verify if a new spoken phrase belongs or not to a specific user. To successfully achieve this goal, the next stage commonly known as training, is to extract the relevant information of each user and construct a suitable model. Hence, the system can match the new phrase with that specific model and give a decision about whether it is a speaker or not.

In the early stages of SV research, *distance-based* methods were popular. Among them, the Dynamic Time Warping (DTW) and the Vector Quantization (VQ) were extensively studied. The DTW method proposed a non-linear mapping or warping of one signal (test recording) to another (target speaker utterance) [45]. The key point is to minimize the distance between both of them and come up with a final decision. In the case of VQ, the purpose was to partition the data space for a speaker s into non-overlapping regions and obtain a codebook that represent that space [46]. When a new recording comes into the

system, a likelihood measure between the incoming features and the codebook is computed. At that stage, experiments were performed just for a few speakers and usually for text-dependent utterances, but it was difficult to extend the methodology to text-independent databases with hundreds of users. The challenge, then, became how to produce suitable models for the new scenario. The solution came along with advances in computing systems. The theoretical statistical methods, like generative and discriminative modeling, were then possible.

The tendency was to construct overlapping models that can be more robust. A way to solve such a problem was to employ a statistical approach. The *Gaussian Mixture Model* (GMM) became that answer capable to represent a speaker in terms of a distribution with just a few parameters. We will leave the GMM approach to be fully explained in detail in the following sections and chapters, including the GMM approach that signified a forward step [47, 48]. Later, it was also shown in [49], that the previous VQ is a special case of the GMM.

This thesis stands on the GMM statistical approach. We analyze SV as a classification problem that is solved using pattern recognition techniques [50]. The solution is given in terms of statistical hypothesis testing [51] and Neyman-Person Lemma [52]. The Lemma states that the relation between two competing point hypothesis models can be expressed as a ratio. The ratio is then compared to a decision threshold and an accept/reject choice is made.

The null hypothesis H_S accepts the speaker as legitimate and the alternative hypothesis $H_{\bar{S}}$ rejects him/her. Under this framework, for a set of observations $X = \chi_1, \chi_2, ..., \chi_T$, the ratio of two hypothesis (target-model and imposter-model) is defined as follows:

$$\theta(X) = \frac{p(H_S|X)}{p(H_{\bar{S}}|X)} \begin{cases} > \tau \text{ accept } H_S \\ < \tau \text{ accept } H_{\bar{S}}, \end{cases} \tag{2.2}$$

where $p(H_S|X)$ and $p(H_{\bar{S}}|X)$ are the posterior probabilities of how likely a hypothesis H_S or $H_{\bar{S}}$ is to happen given the observed data, and τ is the decision threshold.

The hypotheses H_S and $H_{\bar{S}}$ are described by models, Λ and $\bar{\Lambda}$ respectively. Including the probability distribution of Λ and $\bar{\Lambda}$ in the formulation, Equation 2.2 is defined by,

$$\theta(X) = \frac{p(H_S|X, \Lambda_S)}{p(H_{\bar{S}}|X, \Lambda_{\bar{S}})} = \frac{p(X|\Lambda_S)\,(p(H_S))}{p(X|\Lambda_{\bar{S}})\,(p(H_{\bar{S}}))}, \tag{2.3}$$

where $p(X|\Lambda_S)$ and $p(X|\Lambda_{\bar{S}})$ are the likelihood functions with respect to the target and imposter model. The likelihood denotes how probable X is an outcome of the model Λ.[2] Additionaly, recall that $p(H_S) = 1 - p(H_{\bar{S}})$. Then, Equation 2.3 can be transformed in the

[2]For further details on Bayesian modeling refer to [53, 54]

log domain,

$$\theta(X) = \log\left(p\left(X|\Lambda_S\right)\right) - \log\left(p\left(X|\Lambda_{\bar{S}}\right)\right) + C, \tag{2.4}$$

where $\log\left(p\left(X|\Lambda_S\right)\right)$ and $\log\left(p\left(X|\Lambda_{\bar{S}}\right)\right)$ are the log-likelihoods corresponding to the target model Λ_S and the imposter-model $\Lambda_{\bar{S}}$ and C is a constant.

To fulfill the Neyman-Pearson Lemma the following should be considered. Both models are defined by distributions that must be known in advance to reach optimality. However, they are not available in practice and should be estimated. We also need optimal size data sets to estimate those distributions. The more training data at one's disposal, the better estimation we can compute. However, the data sets are generally of limited size, because of the cost of collecting and organizing the utterances. Hence, estimation of those hypotheses can give approximate solutions.

The final ratio – commonly denoted as score – is given by a simplification of Equation 2.4,

$$\theta(X) = \log\left(p\left(X|\Lambda_S\right)\right) - \log\left(p\left(X|\Lambda_{\bar{S}}\right)\right) \begin{array}{l} > \tau \text{ accept } H_S \\ < \tau \text{ accept } H_{\bar{S}} \end{array}. \tag{2.5}$$

The classification problem is reduced to find optimal solutions for the distributions Λ_S and $\Lambda_{\bar{S}}$. Two approaches are commonly used to define $\Lambda_{\bar{S}}$. The first one considers a set of competing speakers $\bar{S_1}, ..., \bar{S_N}$, called cohort, for each user S [55, 56]. The second one employs just a gender-dependent set for all S target users, usually named UBM (universal background model or imposter-model). An advantage of the latter approach is that we just need one imposter model for all the target speakers, and it has been extensively used.

The next sections show various frameworks that have been effectively used to estimate the probability distributions of the speakers and imposters. To compute these models two branches are possible: generative modeling and discriminative training. Traditional SV systems focus mainly on generative modeling (on how to obtain an accurate model of the target speaker voice). However, discriminative training, which estimates more specific models for target and imposter data, has acquired attention recently for different approaches [23, 57, 58, 59]. Discriminative training addresses the problem from a different point of view, it optimizes the correctness of a model by formulating an objective function that penalizes the model parameters using positive and negative data examples. Some other successful architectures employ a combination of both [15, 60, 14]. In the next paragraphs, we will describe the classical descriptions of both of them and emphasize the outstanding characteristics.

2.2.1 Generative Model approach

The generative model is a solution to represent the distinctive characteristics of the speech of a target speaker. The only information available are the data recordings with the corresponding identity label of the speaker and a proposed statistical model to be used. The challenge is to find a distribution (model) that can describe the phenomena– a model that is able to capture the statistics of the vocal tract of each speaker. Hence, the key is to find the *maximum likelihood* between our data and the proposed model. [3]

The *Gaussian Mixture Models*, GMM, showed to be a plausible statistical model in which speech is represented just by few parameters (means, variances and weights) in a simple way.

Most of the current state of the art strategies are based, in many ways, on GMM approach to compute the desired target and imposter models [48, 62, 63]. Let's define $p(X|\Lambda)$ as a GMM distribution for a D-dimendional feature vector as,

$$p(X|\Lambda) = \sum_{i=1}^{M} w_i \mathcal{N}(X|\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i), \tag{2.6}$$

where M are the number of components of the model, w_i are the weights of each component with $\sum_{i=1}^{M} w_i = 1$, and $\mathcal{N}(X|\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)$ is the Gaussian probability density function with $\boldsymbol{\mu}_i$ mean, and $\boldsymbol{\Sigma}_i$ is the covariance diagonal matrix.

The computation of the parameters of the GMM is then the challenge to address. There is not an analytical solution to find the parameters that satisfy the maximum likelihood, but the problem solution employs an iterative algorithm, called *Expectation Maximization*, that optimizes the model parameters.

2.2.1.1 Expectation Maximization

The Expectation maximization (EM) [64] is the leading algorithm for training the GMM, under maximum likelihood criteria. The goal is to search for the maximization of the "likelihood", $p(X|\Lambda)$; *i.e.*, the probability of the realization of X from an unknown distribution, given the model Λ. In every iteration the algorithm updates the GMM model parameters, $\Lambda = \{w_i, \boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i\}$, such that, $p(X|\Lambda^*_{(n+1)}) \geq p(X|\Lambda^*_{(n)})$, where Λ^* is the model estimation at each iteration. The new model then becomes the old model until convergence is reached.

Although there are several explanations of the EM algorithm, in this section, we briefly explain the intuition behind it and leave further details to the next chapters. To obtain a probability distribution that describes our data, EM comprises two stages *expectation* and *maximization*. First, we define a "hidden variable" Z that helps to maximize $p(X;\Lambda)$. Let

[3] For further details about maximum likelihood and the algorithm used we can refer to [61]. An intuitive explanation of *maximum likelihood* is to find a model that best describes our data.

us define, $p(X|\Lambda) = \sum_z (X, Z|\Lambda)$. However, marginalizing Z becomes a difficult task. EM strategy is to update the parameters of Λ^* in two steps. In the Expectation step, we define the function g_n that is a lower bound of the objective likelihood function $\log p\,(X|\Lambda)$. The analytical solution is to compute the posterior distribution $p(Z|X, \Lambda)$. The initial model parameters at this stage are usually computed using VQ techniques. In the *maximization* step, $\Lambda^{(n+1)}$ is obtained by maximizing g_n (see Figure 2.3). In other words, the new Λ^* is computed by maximizing the joint distribution of X and Z.

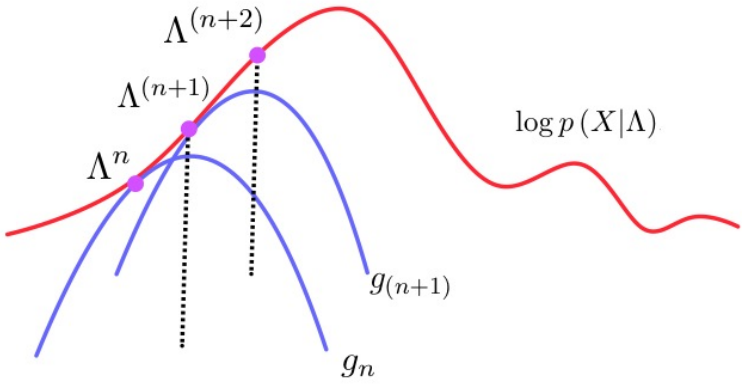


Figure 2.3: Expectation Maximization Algorithm

The purpose of the EM algorithm in SV is to produce a *general model* that embraces the characteristics of all speakers (sometimes excluding the target speakers set). The GMM model at this stage is commonly known as the imposter-model or Universal Background Model (UBM), and in most cases is gender dependent.

At this point, we have solved part of the problem by estimating the imposter or UBM model, Λ^*. The straightforward idea might be to extend the idea of EM and compute the models for each target speaker. However, the availability of the data is the main constraint. The datasets usually contain recordings of just a couple of seconds [39], not suitable for ML training. Therefore, an algorithm able to perform an *adaptation* of the current general model to a target specific model is needed (presented in the next section).

2.2.1.2 Maximum A posteriori (MAP)

EM is not able to compute accurate parameters for the target speaker models because the data available are limited. However, an algorithm proposed by [65], similar to EM, is currently used to adapt the imposter-model or UBM to each target speaker. The formulation is based on the estimation of the model parameters, Λ, that maximize the posterior, $p(\Lambda|X)$. [4] The new estimated model is denoted as $\hat{\Lambda}_{MAP} = \text{argmax}\ p(\Lambda|X)$. The solution of this *adaptation* includes the prior $p(\Lambda)$ in the formulation. It follows the rule,

[4] We recall that by Bayes rule, $p(\Lambda|X) = \frac{p(X|\Lambda)p(\Lambda)}{p(X)}$.

$$\hat{\Lambda}_{MAP} = \underset{\Lambda}{\operatorname{argmax}}\ p(X|\Lambda)p(\Lambda). \tag{2.7}$$

The prior belief, $p(\Lambda)$ is usually given by a previous estimation of a general model, in this case the UBM. The solution of MAP using GMM is intractable, hence, the algorithm is treated in a similar manner as EM as will be shown in next sections in detail.

Note that to obtain an optimal solution the system should consider: *a)* the definition of the prior models, *b)* the appropriate estimation of those priors.

2.2.2 Discriminative Training Approach

The *generative modeling* focuses on optimizing for a single class. In the case of the UBM, the training optimizes the parameters of a general model; in the case of MAP, it updates the parameters of a target speaker model. Hence, the classifier depends entirely on how good the models fit the data they represent (the better the representation, the better the classification). But these methods does not focus on the separation between the classes, target and imposter. In this sense, the *discriminative modeling* gives an alternative solution to the problem. In general, the discriminative approaches consider the opposing classes at the same time and optimize the models for both. They mainly pursue the reduction of an error, rather than improving the model likelihood with respect to the data set. However, the optimization process needs positive and negative samples for each class. The main drawback of these approaches is the amount of data needed to obtain appropriate models and the risk of over-fitting when the models become too specific to the data from which they were trained.

In SV, discriminative techniques have not been extensively used by themselves, but their properties can improve the current approaches or are competitive to the Generative Models. In this section we show some examples of them.

2.2.2.1 Maximum Mutual Information Estimation (MMIE)

One of the most representative approaches of discriminative modeling is MMIE [66, 67]. The main objective of this algorithm is to maximize the *mutual information* between the observations X (belonging to either target or imposter user) and the model tag (Λ_{tar}, Λ_{imp}). Recall that when two random variables are dependent, the mutual information is maximized. If they are independent, the *mutual information* is minimal.

The probability of the observation X is given by

$$p\left(X\right) = p\left(X|\Lambda_{tar}\right)p\left(\Lambda_{tar}\right) + p\left(X|\Lambda_{imp}\right)p\left(\Lambda_{imp}\right). \tag{2.8}$$

Clearly, the criterion is defined in the log domain as the sum over the posteriors of the

observations,

$$f(\Lambda) = \sum_{X \in X_{tar}} \log \frac{p(X|\Lambda_{tar})\, P(\Lambda_{tar})}{p(X)} + \sum_{X \notin X_{tar}} \log \frac{p(X|\Lambda_{imp})\, p(\Lambda_{imp})}{p(X)}. \tag{2.9}$$

The estimated Λ^* is,

$$\Lambda^* = \operatorname{argmax} f(\Lambda) \tag{2.10}$$

It is computed,

$$\Lambda^*_{n+1} = \Lambda_n + \eta \frac{\partial f}{\partial \Lambda}\bigg|_{\Lambda^*_n} \tag{2.11}$$

As shown, this method tries to maximize the class-conditional probability of the observation while the weighted sum of the competing class-conditional probabilities is minimized. Hence the separation between classes is optimized. This observation guarantees that the impostor and target distributions are optimally separated and improve the error. MMIE is the preliminary of more sophisticated discriminative modeling such as MVE discriminative modeling as will be presented in the next chapters.

2.2.2.2 Support Vector Machine (SVM)

Another discriminative approach that has attracted attention recently is SVM. The purpose of the SVM is to classify the multivariate data X into two classes tagged binary as $[1, -1]$ [68, 69]. The data vectors X, that are not linearly separable, are mapped to a higher dimensional space via a kernel function. The solution is a hyperplane with maximal margin, meaning that the distance between a tagged vector and the hyperplane is maximal. A hyperplane is trained by using data X and its tags. In the testing stage, the hyperplane determines the class of each vector.

Two ideas have mainly been developed for SVM [12]. The first one focused on the scoring. The knowledge, in the training stage, of the scores that belong to a user is employed to compute a hyperplane. In the test, the data vectors are compared to the hyperplane and a decision is produced. The final score is the average of the trial data for each target user model. The second one, is to combine the generative method GMM with the SVM [70]. Instead of evaluating each vector, the whole phrase is considered, a new input feature vector is produced for the SVM, and just a final decision is obtained. This techniques have also been successfully applied to spectral, high-level and lately to prosodic features [32, 33].

The evolution of the SVM for SV has become of interest because it can obtain competitive results compared to the generative modeling approach. However, its success is mainly due to the improvements in the kernel design and its capability to refine the modeling in other techniques such as JFA and i-vectors [14].

2.3 Decision Making

After computing models for every target and imposter speaker set (or UBM), the next step is to evaluate the system, obtain feedback from testing on a "controlled database" [5] and perform further normalizations. The target models evaluate different unlabeled recordings (trials) and from each of them we compute a *log-likelihood ratio* with respect to a specific target model. This ratio, usually named *score*, computes a numerical value of the relation between two competing hypotheses. Depending on the value, the system gives a decision (see Equation 2.5).

The score for every trial follows a hypothesis test framework; however, the classifier is not perfect. In the process of verifying whether the speech signal X belongs to a target user S, two errors may arise:

- Type I : known as false rejection, meaning that signal X is incorrectly rejected and is a target speaker.
- Type II: known as false acceptance, meaning that the signal X is incorrectly accepted and is an imposter.

In Figure 2.4, note the location of the errors and the position of the threshold, τ. The scores truly belonging to a target speaker follow a Gaussian distribution. The same is valid for the imposter distribution. Both overlapping distributions represent the classifier. The challenge now is to position the threshold. On one hand, an approach is to set τ as high as possible. However, that will produce few false acceptances and many false rejections, meaning that it will be very secure. The target user might need to perform several trials to finally get access. On the other hand, if τ is designed to be low, there will be many false acceptances and few false rejection. Many imposters might be accredited by the system. Neither of them is desirable. A good tradeoff that depends on the specific application is appropriate. The usual technique to place an effective τ between the two competing score distributions is to perform several experiments using a *development database* and harden the threshold.

2.4 Score normalization

The scores exhibit variations produced by multiple targets speakers and multiple conditions that need to be compensated. Therefore, it is convenient to scale (or normalize) the scores so that they are comparable across multiple targets. Two types of normalizations are the

[5] The *development database* from which we know that ground truth, helps us to evaluate the system in advance and tune the threshold according to specific requirements.

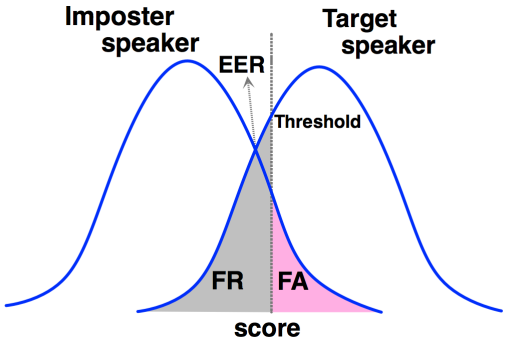


Figure 2.4: Decision Threshold of a binary classifier based on scores

most popular: with respect to the target data (Z-norm) and with respect to the test data (T-norm) [12]. In the following paragraphs we describe them.

2.4.1 Z-norm

The Z-norm [71] or *zero normalization* is performed with respect to the target speaker statistics and compensates for inter-speaker variability. Equation 2.12 shows this type of normalization.

$$\theta(X_{trial}, S)_{norm} = \frac{\theta(X_{trial}, S) - \mu(S)}{\sigma(S)}, \tag{2.12}$$

where μ and σ are the mean and the standard deviation of the scores computed for a certain speaker model S, given imposter data. X_{trial} is the test input data. One of the advantages of this normalization is that it can be performed offline.

2.4.2 T-norm

The other well-known normalization is T-norm [72] or *test normalization*. In this case, the test utterance is evaluated against imposter models as shown in Equation 2.13.

$$\theta(X_{trial}, S)_{norm} = \frac{\theta(X_{trial}, S) - \mu_{impost}(X_{trial})}{\sigma_{impost}(X_{trial})}, \tag{2.13}$$

where μ_{impost} and σ_{impost} are the mean and the standard deviation obtained by evaluating the upcoming phrase against precomputed imposter models. The computation is performed on-line and it is preferred to have separate imposter models for each speaker.

The ZT-norm is the combination of the above and can also be TZ-norm, depending on the order in which they are applied. Most of the state of art systems use one of the normalizations or a combination of them [73].

2.5 Evaluation and Performance measures

The goal of a speaker verification system is to classify the target speakers correctly and minimize the cost of the FA and miss errors. The performance of the verification systems is traditionally evaluated using the Equal Error rate (EER), the detection cost function (DCF) and detection error trade off (DET) curves. In this work, these measurements are the milestone of our research. The optimization methods discussed in this study take into a account their performance as will be explained in next chapters.

2.5.1 Equal Error Rate

The *Equal Error Rate* is an operating point in which the percentage of false positives equal the percentage of false negatives. The operating point is chosen given a threshold such that it fulfills this constraint. The EER provides a general idea of the performance and it is independent of the amount of samples for each class [74, 12, 75].

2.5.2 Detection Cost Function

As explained in previous sections, two errors occur in SV: false acceptance and false rejection. Considering not just an operating point as the EER, the system should manage a "weighted" combination of those errors according to the desire application [75, 76, 77]. For example, if the purpose is to prevent fraud, the system must give a higher penalty to the false acceptance than to the false rejection. Additionally, if the application is an access point, the false acceptance penalty may be relaxed.

In this section, to follow the NIST evaluations, we analyze the problem as a search problem; the system tries to detect the samples that truly belong to a target speaker [78]. If the system fails detecting a sample that in fact belongs to a target, it is considered a *miss*. On the other hand, if the system detects that the sample belongs to a target, but it is from an imposter, it is consider a *false alarm*.

Hence we can consider a cost function of the following form,

$$C(\kappa, \tilde{\kappa}) = \begin{cases} C_{miss} & \text{if } \kappa = 1 \text{ and } \tilde{\kappa} = -1 \\ C_{fa} & \text{if } \kappa = -1 \text{ and } \tilde{\kappa} = +1 \\ 0 & \text{otherwise} \end{cases} \tag{2.14}$$

where κ is the true class of the sample and $\tilde{\kappa}$ is the output of the classifier. $\kappa = 1$ is considered a target sample and $\kappa = -1$ is an impostor sample.

Under this new considerations, the tradeoff between the two weighted errors is established as the expected cost function [76],

$$C(C_{miss}, C_{fa}) = C_{miss} P_{miss} P_{tar} + C_{FA} P_{FA} P_{imp}, \tag{2.15}$$

where P_{miss} is the probability of a *miss* and P_{fa} is the probability of the false alarm. P_{tar} is the prior probability that the target sample truly belongs to a target speaker. $P_{imp} = 1 - P_{tar}$ is the prior probability that the sample belongs to an impostor. C_{fa}, C_{miss} and P_{tar} depend on the application and are decided in advanced by the designer.

Two CDFs are considered in this work:

Minimum Detection Cost Function

The minDCF is the minimum value that can be achieved by minimizing Equation 2.15 given the test set. In this case, it can be used as both as a measure of calibration and discrimination. It is the actual operating point of the cost function at which the optimal cost C is computed, knowing the ground truth of the test database.

Actual Detection Cost Function

This value is computed given a threshold in advance without knowing the test samples. Usually, the threshold is obtained using development data. It is uncertain how this new measure performs on test data; *i.e.*, it may be close or not to the optimum.

2.5.3 Detection Error Trade off Curve

The detection error trade off curve encompasses every operating point for every miss and FA pair [75]. It includes the EER, the CDF and the minDCF. This curve is very similar to the ROC curve, but it presents a non-linear warping at the edges that make the curves look linear.

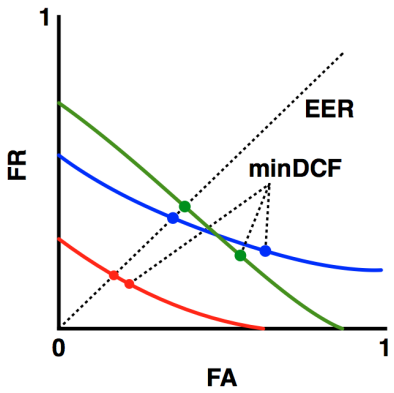


Figure 2.5: Detection Error Tradeoff curve

In this sense, we can compare two or more systems in the same plane over all the operating points as is shown in Figure 2.5.

2.6 Summary

This chapter presented the traditional schemes in SV, giving a survey that goes from feature extraction to decision making. Although every part of the system is important, the core of SV is the modeling: the better the model, the better classification you can achieve. Among the classification methods, discriminative and generative modeling give an interesting idea on how to build a reliable classifier for SV. We leave the state-of-the-art schemes to be explained in depth in the following chapters.

Chapter 3

Speaker Modeling

> The pursuit of truth and beauty is a sphere of activity in which we are permitted to remain children all our lives.
>
> - Albert Einstein

Speaker modeling is the core of the SV system. After computing the feature vectors, the challenge is to build a speaker model that truly represents the speaker's voice even under adverse conditions. For this purpose, let us first recall *Hypothesis Testing* theory and in later sections detail the algorithms of interest.

Hypothesis testing is usually performed through a likelihood-ratio test. A parametric model with parameters Λ_S is defined for the data distribution from S, and an *imposter* model with parameters $\Lambda_{\bar{S}}$ is specified for the class of imposters: the aggregate of all speakers who are not in S. The difference between the log-likelihood of χ given by Λ_S and $\Lambda_{\bar{S}}$ is compared to a threshold to choose the right hypothesis, as shown in Equation 3.1.

$$\begin{aligned} \theta(\chi) &= \log\left(p\left(\chi|\Lambda_S\right)\right) - \log\left(p\left(\chi|\Lambda_{\bar{S}}\right)\right) \\ \text{if} \quad \theta(\chi) > \tau: &\quad H_0 \\ \text{otherwise}: &\quad H_1 \end{aligned} \tag{3.1}$$

The problem of *training* a reliable SV system consists of learning the parameters Λ_S and $\Lambda_{\bar{S}}$ of an appropriately chosen parametric form of the probability distributions.

We first note that each recording χ actually consists of a *sequence* of feature vectors, typically mel-frequency cepstral vectors augmented by their velocity and acceleration. Thus $\chi = \{X_1, X_2, \cdots, X_T\}$, where X_i is the i^{th} vector in the sequence.

The most common form of probability distribution for χ treats the individual X_i vectors as IID with a probability density function given a *Gaussian mixture model*, *i.e.*

$$P(\chi;\Lambda_C) = \prod_i \sum_k w_k^C \mathcal{N}(X_i|\mu_k^C, \Sigma_k^C), \tag{3.2}$$

where C is either S or $\bar{S}$, and w_k^C, μ_k^C and Σ_k^C are the mixture weights, means and covariances (usually assumed to be a diagonal matrix) of the k^{th} Gaussian in the mixture. Thus, $\Lambda_C = \{w_k^C, \mu_k^C, \Sigma_k^C \quad \forall k\}$. In addition, each d-dimensional Gaussian density function, $\mathcal{N}(X_i|\mu_k^C, \Sigma_k^C)$, is defined as,

$$\mathcal{N}(X_i|\mu_k, \Sigma, k) = \frac{1}{(2\pi)^{D/2}\Sigma_k^{1/2}} \exp\{-\frac{1}{2}(X_i-\mu_k)^T\Sigma_k^{-1}(X_i-\mu_k)\}. \tag{3.3}$$

To make the computation easier, we consider all the covariances to be diagonal. Training the SV system is hence equivalent to learning GMM parameters for both S and $\bar{S}$. This research focuses mainly on the estimation of the parameters. In the next sections we describe the main algorithms to model the parameters.

3.1 Expectation Maximization

Learning GMM parameters can generally be performed by the *Expectation Maximization* algorithm, given a suitably large amount of training data. The intuition behind EM was explained in Section 2.2. In this section, we describe the theory behind EM [79, 80] (for detailed information see A.1).

EM is a machine learning tool that works for point estimation. Then, for a set X_t where $t = 1...T$ and a hidden (latent) variable Z, we can estimate the parameters Λ. Let us define the following,

$$\begin{aligned} \ell(\Lambda) &= \log p(x|\Lambda) = \log \sum_z p(x,z|\Lambda) \\ &>= \sum_z Q(z|x,\Lambda) \log \frac{p(x,z|\Lambda)}{Q(z|x,\Lambda)} \equiv F(Q,\Lambda), \end{aligned} \tag{3.4}$$

where $Q(z|x,\Lambda)$ is the density of z. Knowing the inequality, the task is to find the lower bound of $F(Q,\Lambda)$. Hence, $\mathbf{E-step}\ \mathrm{Q}^{\mathrm{t+1}} = \arg\max_{\mathrm{Q}} \mathrm{F}(\mathrm{Q},\Lambda^{\mathrm{t}})$ and $\mathbf{M-step}\ \Lambda^{\mathrm{t+1}} = \arg\max_{\Lambda} \mathrm{F}(\mathrm{Q}^{\mathrm{t+1}},\Lambda)$. Moreover, let us consider $Q^{t+1} = p(z|x,\Lambda^t)$ to force the distribution Q to just depend on Λ^t. Under this argument,

$$\ell(\Lambda) >= \sum_z Q(z|x,\Lambda)\log p(x,z|\Lambda) - \sum_z Q(z|x,\Lambda)\log Q(z|x,\Lambda) \quad (3.5)$$

$$>= Q(\Lambda|\Lambda^t) + S(Q). \quad (3.6)$$

Note that maximizing $F(Q,\Lambda)$ is equivalent to maximizing the expected complete likelihood. Hence, we can compute suitable expressions for the *expectation* and *maximization* steps as follows,

$$\begin{aligned} \mathrm{E}-\mathrm{step} \quad & Q(\Lambda|\Lambda^t) = E[\log p(x,z|\Lambda)] \\ \mathrm{M}-\mathrm{step} \quad & \Lambda^{t+1} = \arg\max_{\Lambda} E[\log p(x,z|\Lambda)]. \end{aligned} \quad (3.7)$$

For the purpose of this research, we applied Equations 3.7 to the special case of GMM. We obtain expressions for each $\Lambda = \{w_k, \mu_k, \Sigma_k\}$. Algorithm 3.1 shows the steps that the system follows. Once we obtained a suitable Universal Model, we need a model for each target speaker.

3.2 Maximum a posteriori adaptation

For SV systems, however, while an arbitrarily large amount of training data may be available to learn the counter model $\Lambda_{\bar{S}}$, the training data available to learn the model Λ_S for any speaker is usually limited. The common solution therefore is to learn a robust counter model $\Lambda_{\bar{S}}$, which in this context is usually referred to as a Universal Background Model (UBM), from a large collection of speech from a large number of speakers, and to *adapt* the UBM to the training data from the speaker to obtain Λ_S.

In the ideal case where the training data from the speaker are recorded over the same kind of channels that will be employed to record test data that must be verified, *maximum a posteriori* (MAP) adaptation assuming conventional conjugate priors for the mixture weights, means and variances has been found to be a good solution for estimating Λ_S [65]. MAP has shown to be a powerful speaker adaptation technique when just a few data samples are available. It takes advantage of the prior information of the model as shown in Equation 3.8. For each speaker S,

$$\hat{\Lambda}_S(\chi) = \arg\max_{\Lambda} p(\Lambda|\chi) = \arg\max_{\Lambda} p(\chi|\Lambda_S)p(\Lambda_S), \quad (3.8)$$

where $\hat{\Lambda}$ are the estimated parameters for the distribution of Λ. Since we have no information about $p(\Lambda)$ it is common practice to use the UBM, *i.e* the UBM is adapted to the training

data of speaker S. Equation 3.8 is also solved under the ML estimation criterion using the same steps as in EM. The details are shown in Algorithm 3.2.

Algorithm 3.1. Expectation Maximization Algorithm

Expectation Step

1. Initialize the parameters of the Gaussian: $w_k, \mu_k, \Sigma_k \ \ \forall k$ and the log-likelihood ℓ_0.
2. For a set of training X_t where $t = 1...T$, compute the total likelihood,
$$\ell_t = \sum_k w_k \mathcal{N}(X_t|\mu_k, \Sigma_k), \tag{3.9}$$
3. Find the responsibilities
$$\nu_{k,t} = \frac{w_k \mathcal{N}(X_t|\mu_k, \Sigma_k)}{\ell_t}, \tag{3.10}$$

Maximization Step

Define,
$$N_k = \sum_{t=1}^{T} \nu_{k,t} \ . \tag{3.11}$$
Then, we can compute:

1. weights
$$\hat{w}_k = \frac{N_k}{N}, \tag{3.12}$$
2. means
$$\hat{\mu}_k = \frac{\sum_{t=1}^{T} \nu_{k,t} X_t}{N_k}, \tag{3.13}$$
3. covariances
$$\hat{\Sigma}_k = \frac{\sum_{t=1}^{T} \nu_{k,t}(x_t - \hat{\mu})(x_t - \hat{\mu})'}{N_k}. \tag{3.14}$$

Check convergence

1.
$$\ell_{new} = \sum_{t=1}^{T} ln \left\{ \sum_k w_k \mathcal{N}(X_t|\mu_k, \Sigma_k) \right\}, \tag{3.15}$$
Compare ℓ_{new} with ℓ_0 and decide whether the criterion is satisfied or not, if not return to Expectation Step 2.

According to ML, it is possible to update the parameter means and covariances. However, it is a common practice to optimize just the means of the parameters and leave the

covariances fixed, no effective improvements are considered by using ML approach [62]. Figure 3.1 shows how the real models are built if just the mean or both mean and covariance are updated.

Another observation about these algorithms is that neither MAP nor EM take into account the variability that might occur due to channel mismatch. For those non-ideal cases, as we will see in the next sections, other powerful methods such as *joint factor analysis* [15, 60] can take into account the variability of the channel and speakers obtaining good results. But first, we briefly explain some discriminative approaches that will aid us to understand the next chapters.

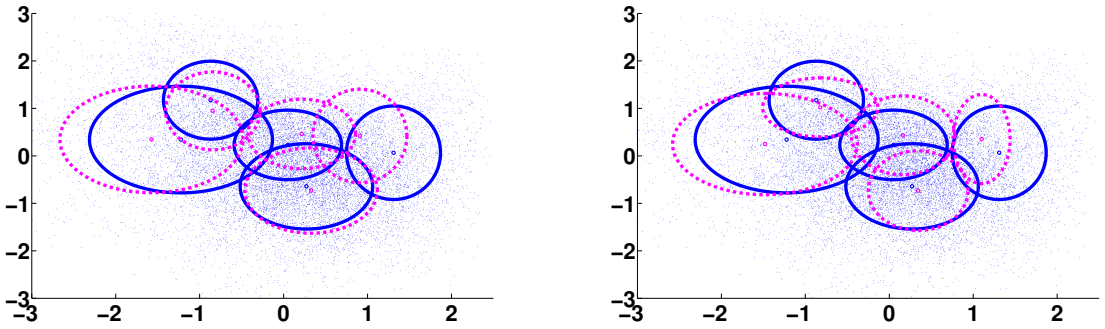


Figure 3.1: Comparison of MAP model optimization: a) only means, b) means and covariance

3.3 Minimum Verification Error

The theory behind minimum verification error was first conducted to solve automatic speech recognition tasks [17, 24] and was called *minimum classification error* (MCE). The algorithm was first conceived as a solution for a classifier optimization problem proposing a methodology to minimize the probability of misclassification[1] [24, 17, 83]. The structured formulation under *Bayes Decision Theory* and the competitive results compared to the state of the art classifiers made it attractive. The algorithm provided an original answer to the optimization of the model parameters, regardless the nature of the distributions and focusing only on the minimization of the error. Moreover, it established an estimation criterion which leads to the update of competitive models in an iterative mode. Afterwards, the approach was embedded into the *Hidden Markov Model* (HMM)[2] theory.

[1] MCE was studied as an alternative solution to the novel classifiers such as Artificial Neural Networks [81] and Vector Quantization algorithms [82].

[2] The complete explanation of the classical ASR process can be found in [84, 85] and is beyond the scope of this thesis. However, we would like to emphasize the influence of ASR techniques and advances that are used in SV with the appropriate modifications.

Algorithm 3.2. Maximum A posteriori Algorithm

Expectation Step

1. Initialize the parameters using the UBM: $w_k, \mu_k, \Sigma_k \quad \forall k$, decide the log-likelihood ℓ_0.
2. For a set of training X_t where $t = 1...T$ and for a specific user s, let's compute the likelihood,

$$\ell_t = \sum_k w_k \mathcal{N}(X_t|\mu_k, \Sigma_k), \tag{3.16}$$

3. Find the responsibilities

$$\nu_{k,t} = \frac{w_k \mathcal{N}(X_t|\mu_k, \Sigma_k)}{\ell_t}. \tag{3.17}$$

Maximization Step

Define,

$$\hat{N_k} = \sum_{t=1}^{T} \nu_{k,t} \ . \tag{3.18}$$

Then we can compute:

1. weights

$$\tilde{w_k} = \left[\frac{\alpha_k \hat{N_k}}{T} + (1-\alpha) N_k \right] \gamma, \tag{3.19}$$

2. means

$$\tilde{\mu_k} = \alpha_k \left(\frac{\sum_{t=1}^{T} \nu_{k,t} X_t}{N_k} \right) + (1-\alpha)\, \hat{\mu}_k, \tag{3.20}$$

3. covariances

$$\tilde{\Sigma_k} = \alpha \left(\frac{\sum_{t=1}^{T} \nu_{k,t} (x_t - \tilde{\mu})(x_t - \tilde{\mu})'}{N_k} \right) + (1-\alpha_k) \left(\hat{\Sigma}_k + \hat{\mu}_k \right) - \tilde{\mu}_k, \tag{3.21}$$

γ ensures that the sum of all weights is equal to one. α_k is considered a regularization coefficient; it is defined as

$$\alpha_k = \frac{\hat{N_k}}{\hat{N_k} + r} \ . \tag{3.22}$$

Check convergence

1.

$$\ell_{new} = \sum_{t=1}^{T} ln \left\{ \sum_k \tilde{w_k} \mathcal{N}(X_t|\tilde{\mu_k}, \tilde{\Sigma_k}) \right\}, \tag{3.23}$$

Compare ℓ_{new} with ℓ_0 and decide whether the criterion is satisfied or not, if not return to Expectation step 2.

The purpose was to optimize the parameters of acoustic unit models such as phones or words [24]. For example, a specific phone could become a target and the remaining ones the imposter set. Then, an iterative process would take place for that specific phone model, and also for its counter imposter model. Later in [86], MCE and MVE were applied for the combined task of utterance verification and SV. In this work, a unified scheme for both tasks was adopted for a two-pass verification system. Recently, another study presented by [87] proposed an improvement to characterize the alternative hypothesis following an optimal discrimination between the target speaker and imposters' set.

From the SV point of view, the estimation of the distributions of S and $\hat{S}$ as computed by EM and MAP, can also be tackled using the binary version of the algorithm, known as MVE (Minimum Verification Error). Under this approach, the empirical error rate is minimized using an appropriate objective function and positive (target speaker features) and negative (imposter speaker features) examples. The procedure performs several iterations while updating the distribution parameters until reaching convergence or a specific threshold.

For minimum classification, supposing that several classes are possible, the prime problem is to correctly classify χ data in C_n classes.[3] The following piecewise-loss function is an appropriate option to measure the performance of such classifier.

$$e_{n,l} = \begin{cases} 0 & n = l \\ 1 & n \neq l \end{cases}, \tag{3.24}$$

where $n, l = \{S, \bar{S}\}$. This function assigns a penalty of zero for correct classification and one for incorrect classification. Moreover, we can define the conditional loss as follows,

$$\Re(C_n|\chi) = e_{l,n}\ p(C_l|\chi). \tag{3.25}$$

If we denote $C(\chi)$ as the classifier decision, based on the observation χ, then for every χ the classifier has to be designed to achieve,

$$\Re(C(\chi)|\chi) = \min_n \Re(C_n|\chi). \tag{3.26}$$

The above equations imply that,

$$\Re(C_n|\chi) = P(C_l|\chi) = 1 - p(C_n|\chi). \tag{3.27}$$

Then, the optimal classifier that solves for the minimum loss is stated as,

$$C(\chi) = C_n \ \text{ if } p(C_n|\chi) = \max_l p\left(C_l|\chi\right). \tag{3.28}$$

[3] In future chapters we will use this approach to classify among attributes such as different SNRs. For now, we follow this formulation and its simplification to a SV scenario.

Equation 3.28, based on Bayes theory, clearly transforms the classification problem into a distribution estimation problem. This Equation can be solved by MAP approaches if we have complete knowledge of the distributions and enough data for each speaker. In most practical tasks this is not the case.

A useful solution is to employ a *smooth* representation of the objective function that is suitable for both the minimization of the error and the optimization. In [24, 17, 83], an optimization criteria is proposed based on likelihood functions and a classifier that operates with the following decision rule,

$$C(\chi) = C_n \text{ if } g_n(\chi;\Lambda) = \max_l g_l(\chi;\Lambda). \tag{3.29}$$

To accomplish this objective three elements are needed and will be discussed in the next paragraphs. Let us first define a set of discriminant functions, such as log-likelihood functions that can be plugged into the objective function. Let $g_n(\chi;\Lambda) = \log(p(\chi;\Lambda_n))$ evaluate the log-likelihood function for class C_n by observing χ. Second, for any speaker we define the misverification measure as,

$$d_n(\chi) = -g_n(\chi;\Lambda) + G_n(\chi;\Lambda), \tag{3.30}$$

$G_n(\chi;\Lambda)$ is considered the log-likelihood average of the competing classes. $G_n(\chi;\Lambda)$ is defined using the following expression,

$$G_n(\chi;\Lambda) = \log\{\frac{1}{M-1}\sum_{n\neq l}\exp[\eta g_l(\chi;\Lambda)]\}^{\frac{1}{\eta}}, \tag{3.31}$$

η is considered a slack factor to control the contribution of the discriminant functions.

For the special case of SV, $M = 2$ (two classes are possible: target or imposter). Furthermore, if $M = 2$, then $\eta = 1$. Equation 3.31 can be reduced to

$$G_n(\chi;\Lambda) = \log g(\chi;\Lambda_{\bar{S}}). \tag{3.32}$$

Third, let's define the new loss function $\ell_n(\chi;\Lambda) = \ell(d_n)$, a sigmoid function that makes a soft decision centered at a threshold:

$$\ell(d_n(\chi)) = \frac{1}{1+\exp(-\gamma d_n(\chi)+\theta)}, \tag{3.33}$$

where $\gamma \geq 1$ and usually $\theta = 0$. If $d_n(\chi)$ is less than zero, it means that not error occurred; if $d_n(\chi)$ is positive, an error happened and it is penalized.

With these three elements we can accomplish a smooth representation of a composite objective function. Finally, we use the indicator function $1(\cdot)$ to sum the losses over the two

classes (target and imposter) and review the performance:

$$\ell(\chi;\Lambda) = \sum_n \ell_n(\chi;\Lambda)1(\chi \in C_n). \tag{3.34}$$

The optimization of the parameters by minimizing the loss function is solved using the generalized probabilistic descendent (GPD) algorithm [24] as shown in Equation 3.35,

$$\Lambda_{t+1} = \Lambda_t - \epsilon\nabla\ell(\chi;\Lambda), \tag{3.35}$$

where ϵ is the learning step, and $\nabla\ell(\cdot)$ denotes the gradient. The above procedure is applied to every element of the model parameters $\Lambda = \{\mu_k^C, \Sigma_k^C, w_k^C\}$.

We perform the optimization dimension by dimension. Then, for target speaker C_S and for dimension d, for example, for μ_S, to avoid bias, let $\mu = \sigma\,\tilde{\mu}$. We compute the following (note that we dropped out all subindices to make the formulation clear),

$$\tilde{\mu}(t+1) = \tilde{\mu}(t) - \epsilon\frac{\partial\ell(\chi;\Lambda)}{\partial\tilde{\mu}}|_{\Lambda=\Lambda_t}. \tag{3.36}$$

By the chain rule we compute,

$$\frac{\partial\ell_(\chi;\Lambda)}{\partial\tilde{\mu}_k} = \frac{\partial\ell(\chi;\Lambda)}{\partial d}\frac{\partial d}{\partial\tilde{\mu}_k}. \tag{3.37}$$

The first element is then,

$$\frac{\partial\ell(X;\Lambda)}{\partial d} = \gamma\ell(d)(1-\ell(d)). \tag{3.38}$$

The misverification measure for the two types of errors:

$$\frac{\partial d}{\partial\tilde{\mu}_k} = \begin{cases} -\frac{\partial g(X;\Lambda)}{\partial\tilde{\mu}_k} + \frac{\partial G(X;\Lambda)}{\partial\tilde{\mu}_k} & X \in C \quad \text{(Type I)} \\ \frac{\partial g(X;\Lambda)}{\partial\tilde{\mu}_k} - \frac{\partial G(X;\Lambda)}{\partial\tilde{\mu}_k} & X \notin C \quad \text{(Type II)} \end{cases} \tag{3.39}$$

$$\frac{\partial g(\chi;\Lambda)}{\partial\tilde{\mu}_k} = \frac{1}{p(\chi;\Lambda)}\frac{\partial\, p(\chi;\Lambda)}{\partial\tilde{\mu}_k}, \tag{3.40}$$

$$\frac{\partial G(\chi;\Lambda)}{\partial\tilde{\mu}_k} = \frac{1}{p(\chi;\bar{\Lambda})}\frac{\partial\, p(\chi;\bar{\Lambda})}{\partial\tilde{\mu}_k}. \tag{3.41}$$

Then, consider for a specific k, dimension d, the final equation to compute the mean is,

$$\frac{\partial p(\chi;\Lambda)}{\partial\tilde{\mu}_k} = \frac{w_k\,\mathcal{N}(\chi|\tilde{\mu}_k,\sigma_k)}{\sum_{k'} w_{k'}\,\mathcal{N}(\chi|\tilde{\mu}_{k'},\sigma_{k'})}\left(\frac{\chi}{\sigma_k} - \tilde{\mu}_k\right). \tag{3.42}$$

The computation for Σ and w_k^C adopt the same procedure (for clarifications and further details refer to Appendix C.1) Once again, let $\tilde{\sigma} = \log(\sigma)$, Then the final Equation to compute σ is

$$\frac{\partial p(\chi;\Lambda)}{\partial \tilde{\sigma}_k} = \frac{w_k \mathcal{N}(\chi|\mu_k, \tilde{\sigma}_k)}{\sum_{k'} w_{k'} \mathcal{N}(\chi|\mu_{k'}, \tilde{\sigma}_{k'})} \left\{ \left(\frac{\chi - \mu_k}{\tilde{\sigma}_k} \right)^2 - 1 \right\}. \tag{3.43}$$

Finally, for the set of weights w_k and to ensure that $\sum_{k=1}^{K} w_k = 1$, let $w = \frac{\exp(\tilde{w})}{\sum_{k=1}^{K} \exp(\tilde{w})}$,

$$\frac{\partial p(\chi;\Lambda)}{\partial \tilde{w}_k} = \frac{\mathcal{N}(\chi|\mu_k, \sigma_k)}{\sum_{k'} \tilde{w}_{k'} \mathcal{N}(\chi|\mu_{k'}, \sigma_{k'})}. \tag{3.44}$$

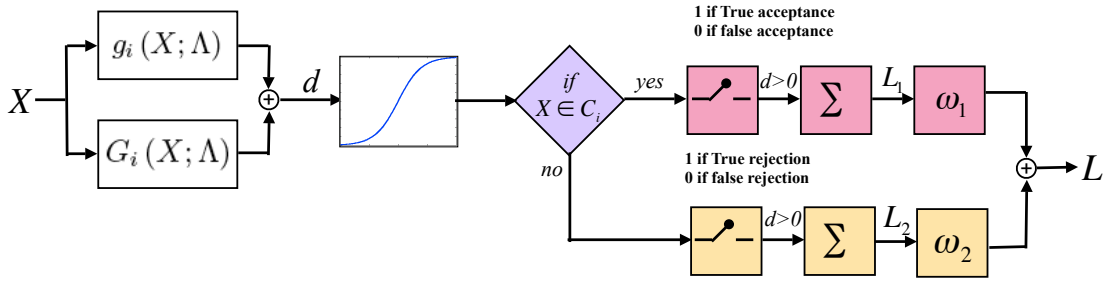


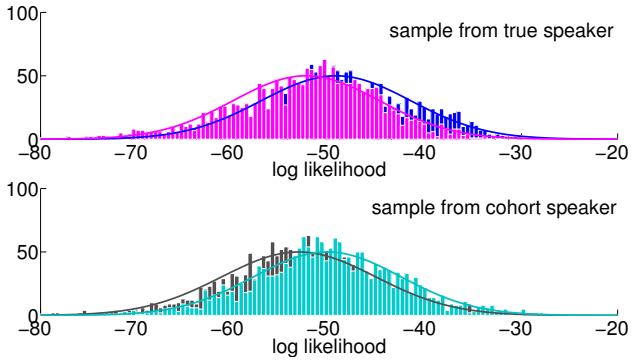
Figure 3.2: Minimum Verification Error architecture

Figure 3.2 describes the formulation in detail. For the training stage and for an i-th target speaker, MVE depends upon an initial target model to evaluate the likelihood $g_i(X;\Lambda)$, an imposter model to compute $G_i(X;\Lambda)$ and a set of target and imposter or cohort speech tokens[4]. Each initial target model can be obtained from the adapted model estimated by MAP or EM. The initial imposter and target models are evaluated using the data-vector χ_t and Equation 3.31. Afterwards, we compute the misverification distance d and map it onto the loss function (sigmoid). Since the system compasses the ground truth of the data, it is able to count for the errors and penalize accordingly for FA and FR. Hence, we construct a loss function that depends on the errors ℓ. The optimization of the model parameters with respect to the minimization of the loss function is the final calculation.

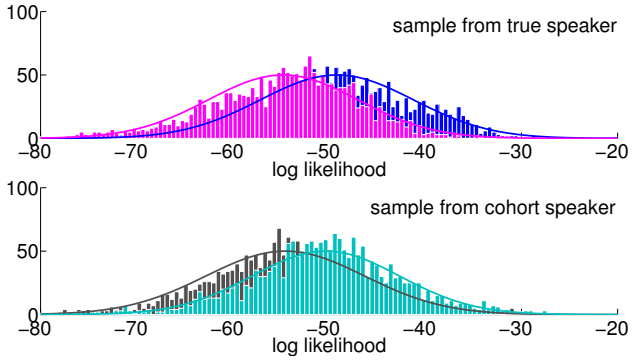
The final outcomes for an i-th speaker are the optimized target and imposter models under the MVE framework, as described in Algorithm 3.3. In the test stage, the system evaluates the unlabeled recording employing the new model parameters for target and imposter. Although MVE is a method per se that can handle random initialization models, it has shown that if combined with techniques based on ML estimation, the models get refined and the classification more accurate [89, 87].

[4]The cohort is commonly defined according to the application [88, 56]; for example, the selection can be based on the likelihood ratio (at the score level), or in terms of similarity between two models as an alternative. There is no agreement on which is the best option [56], but the approach can be successful if built for specific conditions.

It is possible to analyze the effect of MVE in terms of likelihood (see Figure 3.3). The plot depicts the output histograms of the discriminative functions $g_n(\chi;\Lambda)$ and $G_n(\chi;\Lambda)$ before and after optimization. In this example, we chose a scenario where a target user and an imposter show a similar likelihood values (as depicted in Figure 3.3.a). The top histograms show the discriminant functions with respect to a target speaker, purple for target user and pink for imposter. The next graph shows histograms with respect to the cohort or UBM. The green plot describes the distribution of the cohort likelihoods and the gray histogram belongs to the target speaker. We clearly observe the overlapping of the histograms. After 10 iterations using MVE, Figure 3.3.b shows how the histograms broaden as they also separate. This effect aids the accuracy of the classifier, improving the results.



(a) Discriminant function histogram before optimization



(b) Discriminant function histogram after optimization

Figure 3.3: Comparison of target and imposter likelihood histograms

3.4 From models to supervectors

For the traditional approaches and until this point in this thesis, the trend has been to build models from sets of data. Unseen data are classified according to those models. An alternative solution is to find a representation of each utterance using a single vector, usually called *supervector*.

Algorithm 3.3. Minimum Verification Error I

Initialization

1. Train the imposter or cohort model (usually employing EM) and the target speaker model (using MAP)

$$\bar{\Lambda} = \{\bar{W}_k^C, \bar{\mu}_k^C, \bar{\Sigma}_k^C\}, \quad \Lambda = \{W_k^C, \mu_k^C, \Sigma_k^C\}. \tag{3.45}$$

2. Compute the misverification distance and the combined loss function for the FA and FR errors using,

$$d_n(\chi) = -g_n(\chi;\Lambda) + G_n(\chi;\Lambda), \tag{3.46}$$

and

$$\ell(\chi;\Lambda) = \sum_n \ell_n(d(\chi;\Lambda))1(\chi \in C_n). \tag{3.47}$$

Parameter Estimation

For a specific class C, a component k in the GMM, dimension d, and a set of training X_t where $t = 1...T$

1. Obtain the gradients of the *sigmoid loss function* for each of the model parameters: $\{w_k^C, \mu_k^C, \Sigma_k^C\}$ and plug them into:

$$\Lambda_{t+1} = \Lambda_t - \epsilon\nabla\ell(\chi;\Lambda), \tag{3.48}$$

$$\nabla\ell(\chi;\Lambda) = \gamma\ell(d)(1-\ell(d))\frac{\partial\, p(\chi;\Lambda)}{\partial\Lambda_k}. \tag{3.49}$$

2. Optimize the models for every class

   (a) For a specific k, dimension d and to avoid bias, let $\mu = \sigma\,\tilde{\mu}$. Then,

$$\frac{\partial p(\chi;\Lambda)}{\partial\tilde{\mu}_k} = \frac{w_k\,\mathcal{N}(\chi|\tilde{\mu}_k,\sigma_k)}{\sum_{k'} w_{k'}\,\mathcal{N}(\chi|\tilde{\mu}_{k'},\sigma_{k'})}\left(\frac{\chi}{\sigma_k} - \tilde{\mu}_k\right). \tag{3.50}$$

   (b) let, $\tilde{\sigma} = \log(\sigma)$, Then σ is

$$\frac{\partial p(\chi;\Lambda)}{\partial\tilde{\sigma}_k} = \frac{w_k\,\mathcal{N}(\chi|\mu_k,\tilde{\sigma}_k)}{\sum_{k'} w_{k'}\,\mathcal{N}(\chi|\mu_{k'},\tilde{\sigma}_{k'})}\left\{\left(\frac{\chi-\mu_k}{\tilde{\sigma}_k}\right)^2 - 1\right\}. \tag{3.51}$$

   (c) For w_k and to ensure that $\sum_{k=1}^K w_k = 1$, let, $w = \frac{\exp(\tilde{w})}{\sum_{k=1}^K \exp(\tilde{w})}$.

$$\frac{\partial p(\chi;\Lambda)}{\partial\tilde{w}_k} = \frac{\mathcal{N}(\chi|\mu_k,\sigma_k)}{\sum_{k'}\tilde{w}_{k'}\,\mathcal{N}(\chi|\mu_{k'},\sigma_{k'})}. \tag{3.52}$$

Algorithm 3.3. Minimum Verification Error II

Check for convergence

1. Compute the loss function with the new parameters:

$$\ell_{new}(\chi;\Lambda) = \sum_n \ell_n(d(\chi;\Lambda))1(\chi \in C_n). \tag{3.53}$$

2. Compare ℓ_{new} with ℓ if it is higher than a threshold τ iterate from *Parameter Estimation*.

An approach of this kind was first presented in [90] and then reutilized by [91]. Both showed that time-averaged features contain relevant information of the speaker that can be represented as a single vector (that acts like a model). Afterwards, a distance measure between new utterances' super vectors and the "models" is computed and a decision is taken.

Other algorithms evolved in the same direction. SVMs, for example, created a whole theory around the construction and classification of these supervectors. The *sequence kernels* are part of this effort. The key point is to map a set of vectors to a single one via a kernel and perform classification [92]. It is not under the scope of this thesis to have a deep description of these approaches. However, their importance is notable.

The GMM components used as supervectors marked a watershed in the algorithms. The k means of all the Gaussians in a GMM are concatenated into a single vector, defined as a *supervector*,

$$M = [\mu_1|\mu_2|\cdots].$$

The new supervectors are the input to different state-of-the-art systems such as SVM [93, 70], and other factor-analysis-based techniques (explained in the next sections).

3.5 Factor Analysis

To understand the state of the art of Joint Factor Analysis and i-vectors it is useful to explain first the simplest case of factor analysis (FA). Factor analysis establishes that continuous factors can control data; meaning that these factors can model the covariance structure of the high-dimensional data. So we can think that the data can be generated by first locating a point in a subspace and then adding noise. If that is the case, we can consider the following:

$$X - \mu = LZ + \epsilon, \tag{3.54}$$

where X is a random variable of dimensions $P \times 1$ (observation), μ is the data mean vector, L refers to the loading matrix of $P \times R$ dimensions, Z is the latent random variable (known as factor vector) of $R \times 1$ dimensions and ϵ is a random residue with diagonal covariance matrix ψ. For a further and detailed explanation of FA, please refer to Appendix B.

The key point of FA, in the context of SV, is to think about a supervector that belongs to a speaker which is disturbed by some kind of perturbation. Under this scheme, the speaker model remains the same for all the recordings in different conditions, represented by the speaker factors. ϵ describes the contribution of the perturbation to the model. Generally, the first term, LZ, represents the contribution of any factor, but can also be extended to include multiple factors, as shown in joint factor analysis, where the model is decomposed into speaker and channel contributions. In either case, the solution of Equation 3.54 is performed in two ways, using Maximum Likelihood Estimation or Discriminative Modeling. Both techniques will be explained in detail in the following sections.

3.5.1 Joint Factor Analysis (JFA)

The purpose of factor analysis is to reduce the channel and speaker variability by decomposing the distributions of the feature vectors into speaker and channel factors [15, 60]. In this framework, let us once again define C as the number of components of a GMM, Λ, from a speaker S, that form a *supervector* $M = [\mu_1|\mu_2|\cdots]$ (Here we have not explicitly shown the superscript representing the speaker for generality). The supervector $M_{S,H}$ representing the GMM for the distribution of data over *each channel type* H by a speaker S is assumed to be composed from a collection of factors as:

$$M_{S,H} = s + c, \tag{3.55}$$

where s is the speaker supervector and c is the channel supervector. Moreover, the distribution of s can be described by

$$s = m + Vy_S + Dz_S, \tag{3.56}$$

where m is a $CF \times 1$ supervector that represents the global mean across all speakers, V is a low rank rectangular matrix which columns are the eigenvoices representing the subspace over which speaker-specific components of $M_{S,H}$ lie, y_S is a normally distributed random vector (with 0 mean and unit variance) representing speaker factors specific to speaker S, D is diagonal matrix and z is a normally distributed random vector representing residual error.

In the same way,

$$c = Ux_{S,H}, \tag{3.57}$$

where U is a low rank rectangular matrix, which columnns are the eigenchannels representing the subspace over which the channel specific components of $M_{S,H}$ lie, and $x_{S,H}$ is a normally distributed vector representing channel factors specific to recordings of speaker S over channel H. Baum-Welch statistics for learning the loadings V, U and D, the EM procedures for learning the factors y_S, x_H, z_S, as well as the methods for classification with the resulting model in a manner that factors out the interfering contributions of the channel are well laid out in [60, 15]. The computation of the sufficient statistics such as 0-th, N, the 1-st, F, and 2-nd, $\mathfrak{S}$ are an aid to estimate la loading matrices and the factors.

For verification, we match an utterance with s; that is, we assume that the supervector has the form $s + Ux_{S,H}$. Several scoring methods have been proposed [94]. To perform the usual likelihood ratio and integrating out the channel contribution is unfeasible. From the alternative approaches, the one that has attracted more attention is the *linear scoring* that presents a simplification that reduces the evaluation to matrix multiplication of the form,

$$\theta_{s,H} = (V y_{tar} + D z_{tar})\Sigma^{-1}(F_{tst} - N_{tst}\, m - N_{tst} U\, x_{tst}), \tag{3.58}$$

where *tar* refers to target speaker and *tst* to unseen recording and N refers to the 0-th order, and F to the first order statistic according to the test set.

This technique has evolved in recent years obtaining good results [60]. It has been adapted to various circumstances; for example, different training and testing channels (interview, telephone and cross-channel scenarios).

3.5.1.1 Joint Factor Analysis step by step

Algorithm 3.4 summarizes the JFA algorithm step by step. It describes the clue steps to estimate the parameters of the loading matrices V, U and D for a speaker S, using Baum-Welch algorithm. Thereafter, the factors x, y and z are estimated in terms of their posterior means and the observations, employing the loading matrices. For clarity we use the subscript to emphasize the speaker S or channel H. For a more detailed explanation, please refer to Appendix B.

3.5.2 Frontend Factor Analysis: i-vector

This training approach is based on the idea of a new low-dimensional speaker and channel-dependent space representation using factor analysis [14]. Whereas in JFA two spaces were defined, the speaker with its eigenvoice matrix V and the channel with its eigenchannel matrix U, in this case the new space contains both in only one space. The new space is known as "total variability space" and encompasses the variabilities of the speaker and the

channel. A total variability matrix, embracing the eigenvectors with the largest eigenvalues, is obtained from the total covariance matrix. Then, the new speaker and channel model is defined as

$$M = s + T\omega, \tag{3.59}$$

Algorithm 3.4. Joint Factor Analysis Algorithm I

Train the JFA matrices

1. Assume that U and D are zero, train the loading matrix V,

$$M = m + Vy.$$

   (a) Compute the Baum and Welch Statistics 0-th, N, the 1-st, F, and 2-nd, $\mathfrak{S}$, order statistics of speaker s and Gaussian component c employing observations T.

   (b) Using Baum-Welch statistics, estimate the posterior of the hidden factor y.

   (c) Compute statistics across speakers using the information of posterior distribution of the estimated y.

   (d) Compute V and the update of the covariance.

   (e) Iterate from a to d between 15 to 25 times. In every iteration substitute the estimated V.

2. Assume D is zero, train the loading matrix U given the estimate of V,

$$M = m + Vy + Ux.$$

   (a) After estimating y for each speaker, compute the Baum and Welch statistics, N_H and F_H, for each H (conversation side or channel) of each speaker S.

   (b) Calculate the shift for each speaker using $R = m + Vy_s$ and compute its Gaussian posterior. Subtract it from the first-order statistics F. The new $F_H(s)$ depends on the channel and the speaker.

   (c) Use the new statistics to train U.

   (d) Iterate from a to c 15 to 20 times using the updated N_H and F_H.

Algorithm 3.4. Joint Factor Analysis Algorithm II

3. Train the residual D given the estimates of V and U,

$$M = m + Vy + Ux + Dz.$$

   (a) Compute the speaker shift of $m + Vy_s$, and the channel shift, $Ux_{s,H}$. Compute the Gaussian posterior of the shifts and subtract them from the first order statistics, $F_{S,H}$.

   (b) Estimate the initial factor z and update the statistics across the speakers.

   (c) Calculate D

   (d) Iterate from b to c 15 to 20 times.

Compute factors

- Estimate x (channel), y (speaker) and the residual z in terms of the posterior mean and the observations.

Scoring

- Compute the final score by a simplified version of the log-likelihood ratio. Given a new utterance (referred here as tst), the loading matrices and the factors (referred here as tar), compute the product,

$$\theta_{s,H} = (Vy_{tar} + Dz_{tar})\Sigma^{-1}(F_{tst} - N_{tst}\, m - N_{tst}U\, x_{tst}).$$

where M represents the combined speaker and channel-independent supervector extracted from the UBM. T is a low-rank rectangular matrix and ω is a random vector, composed of the identity vectors (*i-vectors*), with normal distribution $N(0, I)$. M is normally distributed with mean m and covariance matrix TT'. The procedure to train the matrix T is very similar to the training of V in JFA, but in eigenvoice training the different conversation sides of the target speakers are treated as different speakers. Hence, the new modeling is in fact a projection of an utterance onto the low-dimensional total variability space.

ω, the hidden variable can be solved using Baum-Welch statistics. If we consider ω to be defined by its posterior distribution, a sequence of L frames $\{y_1, y_2, ..., y_L\}$, and a UBM of C components in a F-dimensional space; then,

$$N_c = \sum_{t=1}^{L} p\left(c|y_t, \Omega\right), \tag{3.60}$$

$$F_c = \sum_{t=1}^{L} p\left(c|y_t, \Omega\right) y_t. \tag{3.61}$$

where c is the number of the Gaussian component, $P\left(c|y_t, \Omega\right)$ is the posterior probability of c given the realization y_t. Moreover, the centralized first-order Baum-Welch statistics, using the UBM, is as follows:

$$\hat{F}_c = \sum_{t=1}^{L} p\left(c|y_t, \Omega\right)\left(y_t - m_c\right), \tag{3.62}$$

where m_c is the mean of UBM component c. Then, the i-vector for a phrase is computed as,

$$\omega = \left(I + T'\Sigma^{-1}N(u)T\right) \cdot T'\Sigma^{-1}\hat{F}(u), \tag{3.63}$$

where $N(u)$ is a $CF \times CF$ diagonal matrix composed of $N_c I$ diagonal blocks. The super vector $\hat{F}(u)$ of dimension $CF \times 1$ is computed by concatenating all $\hat{F}_c$ for a given phrase u. Σ is the $CF \times CF$ diagonal covariance matrix that models the residual variability.

Once the new feature vectors, *i-vectors*, are computed, then next step is to build an appropriate model that can discriminate among speakers. Two main branches were explored. On one hand, SVM was used on extensive studies as shown in [93, 70][5]. On the other hand, the most popular one is PLDA [95].

3.5.2.1 Probabilistic Linear Discriminant Analysis (PLDA) model

PLDA is a generative modeling approach to handle *i-vectors*. It was first used in image recognition scenarios [95] and then extended to SV [73]. Note that the formulation is similar to the one employed by JFA.

The supervector $r_{S,H}$ representing the i-vector of a single recording over *a channel* H by a speaker S is assumed to be composed of a collection of factors,

$$r_{S,H} = m + Vy_S + Ux_{S,H} + \varepsilon, \tag{3.64}$$

where m is an offset, V is a loading matrix representing the subspace over which *speaker-specific* components of $r_{S,H}$ lie (dealig with inter-speaker variability), and U is a loading matrix representing the subspace over which the *channel-specific* components of $M_{S,H}$ lie (deals with intra-speaker variability) and ε is the residual variability. The factors $x_{S,H}$ and y_S follow normal distributions and ε follows a Gaussian with zero mean and diagonal variance. The modeling solution follows the same steps as JFA.

[5]It is beyond the scope of this thesis to describe in detail the SVM approaches. However, the methodology is very interesting. Details can be found in [74]

Finally, for scoring, we rely on a likelihood ratio of the form,

$$\theta(X) = \frac{p(r_S, r_{new}|H_1)}{p(r_S|H_0)\, p(r_{new}|H_0)}, \tag{3.65}$$

where r_{new} stands for the i-vector from an unseen phrase, H_1 is the hypothesis of both vectors belonging to the same speaker, and H_0 represents the hypothesis of both vectors coming from different speakers.

Algorithm 3.5 delineates the steps of this methodology. We omitted the parts that are similar to the JFA approach.

Algorithm 3.5. *I-vector* Algorithm

Train the Loading matrix

1. Train the loading matrix T,

$$M = m + Tw.$$

   (a) As in JFA, compute the Baum-Welch statistics 0-th, N, the 1-st, F, and 2-nd, $\mathfrak{S}$, order statistics of speaker s and Gaussian component c. Collect the phrases belonging to specific channels and speaker, treat the different conversation sides of the target speakers as different speakers.

   (b) Using Baum-Welch statistics, estimate the posterior of the hidden factor w.

   (c) Compute statistics across speakers using the information of posterior distribution of the estimated w.

   (d) Compute T and the update of its covariance.

   (e) Iterate from b to d between 15 to 25 times. In every iteration substitute the estimated T.

Channel compensation

Two approaches have shown successful results:

1. Compute the linear discriminant analysis (LDA) to reduce the dimensionality of the i-vectors. Afterwards, use Probabilistic Linear Discriminant Analysis (PLDA) as a the target trainer
2. Compute LDA of the i-vectors followed by within-class covariance normalization (WCCN).

Scoring

- Compute the cosine distance scoring (CDS) on the updated i-vectors (after channel compensation).

3.6 Summary

This chapter described the most popular modeling algorithms. We presented the theory behind the state-of-the-art approaches including generative and discriminative modeling. Moreover, we showed training examples using real data to point out the effects they may have to perform any of the optimizations. Lastly, we emphasized the MVE approach. In the next chapters we present modified versions of the current approaches, based on a discriminative approach similar to MVE, that can improve current results.

Chapter 4

Robustness in Speaker Verification

What is essential is invisible to the eyes.

- Antoine de Saint Exupéry *The Little Prince*

In real conditions the speech signal is exposed to extraneous distortions that affect system performance. Robustness in SV deals with these undesired circumstances. It has become a complex process that takes into account the adverse conditions in the environment and variability among recordings. Noise in the background is a straightforward problem that affects the speech signal. The most common types of noise are additive (background noise) and convolutive (channel noise). In the case of additive noise, the speech signal is distorted when background noise is added to a clean speech signal (this effect can be clearly observed in the spectral domain). Convolutive noise appears as a result of the multiplicative contribution of the channel to the signal in the frequency domain. Moreover, apart from noise, system performance must guarantee that variability among recordings of the same speaker is minimized and that variability among different speakers is maximized. In this chapter, we give a description of these problems and examples of their current solutions.

4.1 Inter-speaker Variability

In SV, an utterance and a claimed identity is processed through a system that had been previously trained on a set of target models. The evaluation of those spoken phrases produces a similarity value that enables the system to accept or reject the user. It is desirable that the matched trials of a target model "a" do not get a high similarity value for other models "$b, c, d, ...$" that include the imposter and other possible targets. In addition, different trials of the same user should get tied to his own model. One of the challenges then is to maximize the similarity distance between the target model and the competing classes, while minimizing the distance among the recordings of the same speaker.

The most common solution is to efficiently train the target models for each speaker. We assume that the voice of each speaker can be differentiated from the rest. Every speaker shows the property of uniqueness; *i.e.*, the speech is clearly defined by the particularities of the vocal tract. From the classical approach of SV, maximum likelihood is a sufficient solution that gives satisfactory results even though it does not contemplate the strict modeling of an imposter speaker [13]. From the ML perspective, a target model fits the data of a particular speaker (the better the model, the better the classification). Another solution is given by Factor Analysis [14, 15] which describes an utterance as a linear combination of speaker factors. In this approach, the undesired variations are marginalized out. Later, in a joint combination with channel factors, the channel contribution can be also modeled and left out of the computation, extracting the unique information of each speaker. A similar strategy applies for i-vectors.

4.2 Session Variability

Now that the SV system guarantees that classification between a target user and the imposter is successful, the next step is to determine the variability of a target speaker among different recordings. When speaking about text-independent SV system the training and the test differ in content, so the system relies only on the acoustic information contained in the data. Hence, the algorithms must consider the variability between two utterances from the same user recorded at different times and under different conditions. Those differences among recordings are known as *session variability* and have become one of the most challenging issues in SV.

In a real scenario, the conditions employed to train the models are usually different from the ones used in the test (mismatch condition). To tackle this problem, several approaches have been proposed. The early ones just dealt with one channel [96, 97], usually telephone. However, as the technology evolved, new channel conditions were investigated and with them how to manage the channel mismatch. The new channels consisted of different types of telephones (cellular, landline) and microphones. The first studies in this new scenario used a model for each channel condition [96, 39, 97]. Traditional ML and SVM approaches were sufficient to build systems that compensated for the channel contributions. Afterwards, more sophisticated algorithms were developed such as JFA and i-vectors [98, 99, 100]. The idea of describing a spoken phrase in terms of a linear decomposition of speaker and channel factors marked the new state of the art [15, 14].

4.3 Noise and its effects in Speaker Verification

In this section, we discuss how additive noise affects SV. In real conditions, noise distorts every stage of a speaker verification system: front-end, modeling and evaluation (scoring). Therefore, well-known ASR tools are able to solve the problem at each stage [40, 101], but also it is possible to view the problem from a SV perspective [102, 1].

The additive noise model in the frequency domain for an observed signal $x(k)$ composed of speech signal $s(k)$ and noise $n(k)$ is,

$$X(e^{j\omega}) = S(e^{j\omega}) + N(e^{j\omega}), \tag{4.1}$$

where X, S and N represent the spectrum of the observed signal, the speech and the noise. In the ideal case, the noise could be removed, if we had complete knowledge of the noise at every frequency. However, the process of estimating the noise and its phase is not an easy task.

4.3.1 Noise in the feature domain

In real circumstances, the effect of the noise can be observed in the spectral domain as shown in Figure 4.1. The distortion causes not just poor intelligibility, but a significant change in the spectrum. In the clean case, the vowels are well defined; while in noisy conditions, it is difficult to define the starts and ends of the acoustic units (words and phones). To make it suitable for SV application, the next issue is the impact on the VAD. An important element in SV is the correct extraction of speech that contains relevant information from a speaker without the silences. In the noise case, it is not clear which parts of the signal contain no information about the speaker (see Figure 4.1), making it difficult for the VAD to work properly.

One possible solution is to use tools from ASR to deal with the noisy conditions – clean the noisy signal and use the same SV scheme. Another possible solution, proposed in [103], is to include enhancement techniques that can improve the quality of the signal.

Once we extract the appropriate segments of speech, the next issue to consider is the aftermath in the cepstral domain. The feature vectors are usually computed using magnitude spectra and then transformed into MFCCs. An easy way to observe the impact of noise is to plot a scatter plot. The x-axis depicts a dimension of the MFCCs for clean speech; the y axis maps out the same MFCC dimension of the noisy speech. The line at 45 degrees represents the identity function $x = y$, see Figure 4.2 (also shown in [104]). Note the spread and the variability of the data with respect to the diagonal after adding 0-dB noise. The challenge is then to reduce the variability (transform the data to reach the diagonal) so that the model represents the target user.

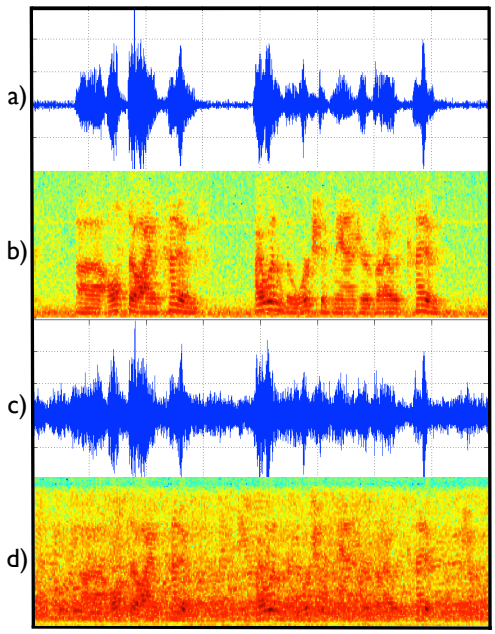


Figure 4.1: Shows the effect of the noise in the spectral domain. *a)* represents the clean speech signal in time domain. *b)* shows the clean speech signal spectrum. *c)* depicts the speech signal at 0 dB SNR. *d)* displays the spectrum of the same speech signal at 0 dB SNR.

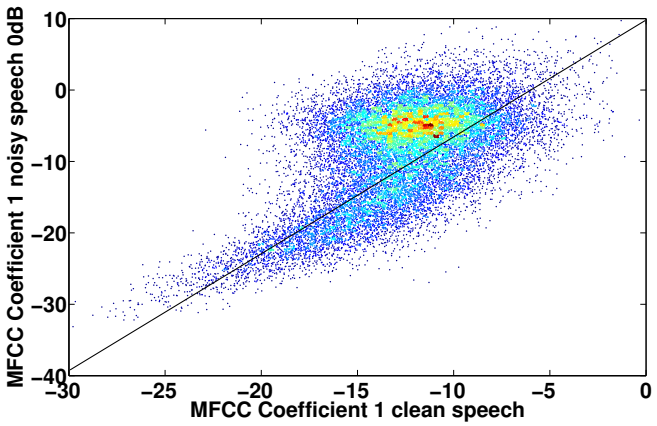


Figure 4.2: log scatter plot of the first MFCC clean speech against noisy speech at 0 dB.

4.3.2 Noise in the models

In this section, we compare the clean and noisy speech models. When noise is present the parameters of the GMM models are modified – the means are shifted, the covariances and the weights are also changed, see Figure 4.3. Depending on the type of noise (the SNR and degree of stationarity) the models updates are unpredictable.

Moreover, the new models present a shift and modifications in the parameter models $\Lambda = \{w, \mu, \Sigma\}$ (see Figure 4.4). Therefore, compensation algorithms that can handle this effect are necessary in the SV systems.

Examples of *model compensation* techniques to overcome this problems are described in

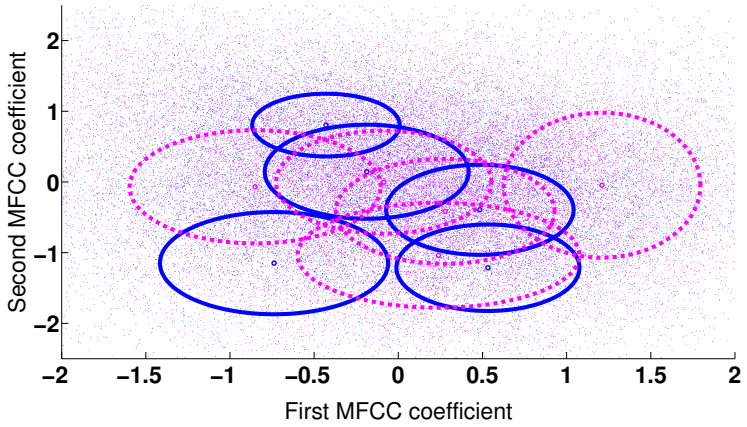


Figure 4.3: Scatter plot comparison between noisy and clean features: the continuous ellipses show the GMM component of a speaker model in clean conditions; the dotted ellipses represent the same recordings with added noise at 0 dB.

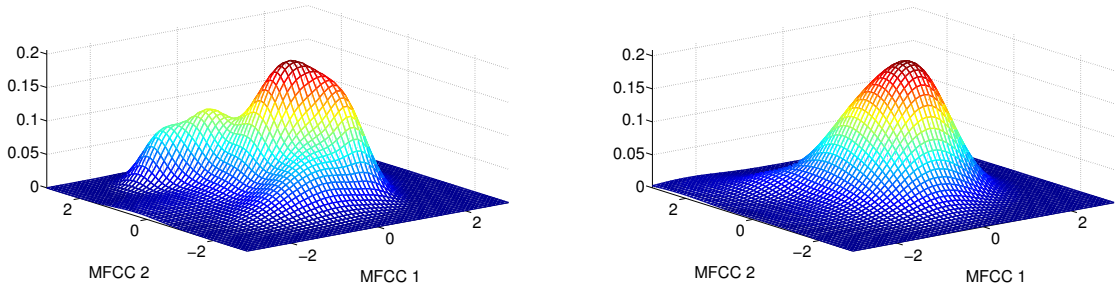


Figure 4.4: Example of model distribution for one speaker, a) speaker model in clean conditions b) speaker model in noisy conditions (0 dB SNR).

[105, 106]. The first one adapts the current GMM speaker model to the desired condition, in a similar way as feature mapping does in the feature domain [106]. A transformation function trained with prior information about the noise condition converts either the models or the features to a new space. For these two schemes, the noise-type labels of the models or features are needed to decide the most accurate transformation function. The second approach proposes a blind compensation; *i.e*, the system does not have knowledge of the noise and the noise-clusters are computed in an unsupervised manner. Even though these approaches showed improvements, the new JFA and i-vectors ensure better results due to its capability of dealing with unknown conditions (the latter approach will be explored in next sections).

4.3.3 Noise in the score domain

The noise not only exhibits its impact at a single operating point (for example the EER), but also in the score distributions and along the DET curve (see Figure 4.5). The noise

degrades the signal and manifests itself in every stage. Poor performance is clearly observed in the evaluation (last classification of the system). Targets are confused with imposters, and the opposite is likely as well. For a binary classification, an intuitive explanation is that the opposing distributions overlap, increasing the FA and FR and making the classification difficult. Also, we note that for the noise-score distributions the mean and variances differ. As a result, the DET curve is higher for noisy speech, including the EER.

Score normalization approaches such as Z-norm [71], T-norm [72] and ZT-norm [12] can alleviate the problem to some extent. Their main purpose is to eliminate the offsets caused by extraneous factors. However, the problem of low SNRs is addressed by model compensation or robust feature extraction techniques.

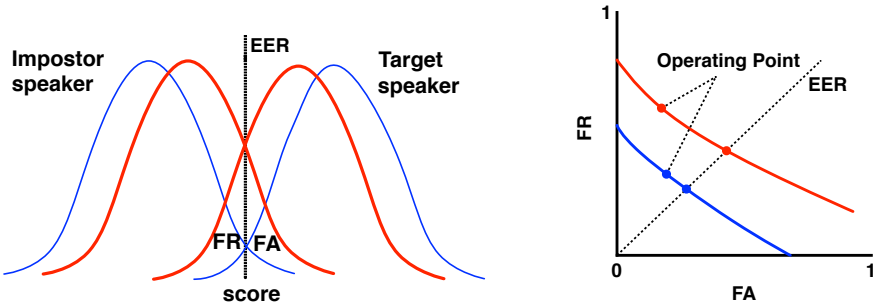


Figure 4.5: Score distributions and DET curves for two scenarios: a) blue line represents the clean condition, b) red line shows the noisy condition.

4.4 A glance of Robust solutions

The solution to the noise problem can be solved using well known Automatic Speech Recognition tools (as shown in this section). However, it is also useful to tackle the problem by employing embedded algorithms in the SV system. A few solutions have been given to address the noisy conditions in Speaker Verification that will be described in the next paragraphs.

4.4.1 Robustness in Automatic Speech Recognition

Researchers proposed to directly work on solutions similar to the ones employed for ASR. In the following paragraphs we want to give a general view of what we consider to b important algorithms that are the basis for some of the SV algorithms. The solutions that have been extensively used in ASR are divided into three main groups: speech enhancement, feature vector adaptation or normalization, and acoustic model adaptation. In this thesis we will not discuss them in depth, but we consider that the basis of the algorithms have evolved or can be used for either ASR or SV.

The objective of speech enhancement is to clean the speech signal in the feature extraction process. Examples of the algorithms are CMS (cepstral mean subtraction) [40] and SS (spectral subtraction) [101].

Feature vector adaptation or normalization occurs in the feature space. The algorithm models the distortion between clean and noisy features by a parametric function that can later be used to compensate the mismatch in the test features. A basic example of this normalization is RASTA (relative spectral amplitude) [42], which is a high-pass filtering that reduces the effect of a varying channel. Another example is CDCN (codeword-dependent cepstral normalization) [107]. In this case, the transformation is based on the normalization of the acoustic space, defining a universal codebook that can be transformed to the vectors in a phrase.

Model adaptation compensates for the mismatch by adjusting the parameters of the acoustic models. This scheme sometimes gives better results because it can model the uncertainty of the noise. One of the main disadvantages is that a certain amount of data is needed to obtain reliable statistics. Examples of these schemes are MAP (maximum a posteriori estimation) [65], MLLR (maximum likelihood linear regression) [108], PMC (parallel model compensation) [109, 110], and VTS (vector Taylor series) [111, 112]. Moreover, we emphasize the Ensemble Speaker and Speaking Environment (ESSE) modeling method. The last examples are SPLICE (stereo-based linear compensation environment) [113, 114] and MEMLIN (multi-environment model-based linear normalization) [115]. Although these methodologies have been used for ASR, they can be extended to SV.

4.4.2 Factor analysis and Robustness

Recently, the research community presented one of the first efforts to address robustness in SV. The first step was the construction of a noisy database to show the limitations and potentialities of the current state-of-the-art systems [102]. The new baseline database is an extension of the clean NIST [116], switchboard [117], and Fisher [118] databases adding noise as well as reverberation. It also includes characteristics, already treated in previous evaluations, such as channel mismatch between the train and test sets, and different degrees of vocal effort. The new database was built artificially, *i.e.*, from a set of clean data the noise and reverberation was added. The noise belongs to cocktail party noisy recordings at different SNRs (8, 15, 20 dB). The reverberation was recorded at different room types and reverb times (at 0.3, 0.5, 0.7 s).

The system uses the current state-of-the-art scheme; it includes an i-vector front-end, followed by LDA for dimensionality reduction and PLDA modeling [119]. The system shows promising EERs – of less than 3%– given by clean speech even when there is a mismatch between the test and training conditions. Moreover, the current approach was also tested for

different feature extractors: MFFCs, prosodic polynomial coefficients and MLLR (maximum likelihood linear regression). For these cases the degradation of the EER is from 3 to 14 times compared to the baseline. The results showed that the method does not depend on the feature extraction scheme. It is important to note that at this stage the training of the UBM and the i-vector is computationally expensive, so a good solution is just to add noise to the PLDA and LDA part. The first attempt was to add noise to every part of the system, but including extra noisy features to model the UBM didn't give any significant improvements. When adding noise just to the PLDA modeling, a 40% improvement was achieved. These studies showed the potentiality of the current state of the art.

4.4.2.1 Vector Taylor Series VTS

Another way to address the modeling is to define a mapping function that characterizes the mismatch between the training and testing models [111, 112]. This study suggests that the effects of the noise effects have been underestimated due to mathematical simplifications. It characterizes the statistics of the additive noise and the linear filtering in a transmission channel. This method can be applied to the feature vectors (including delta and double delta) or even to the representation of those vectors. A special case of this type of modeling is the Jacobian adaptation (JA).

An extension of the state of the art that modifies the front-end using VTS is presented in [1]. This approach tries to clean the i-vectors, but it is not used as the usual VTS, where we search for a mapping function that characterizes the mismatch between the training and testing conditions. Under this new scheme, the utterance of the GMM, similar to what it is done in JFA, is decomposed into two distributions: the clean and the noisy. The clean distribution belongs to the i-vectors. Note that by including the VTS, the system can model nonlinear effects in the GMM caused by the additive noise. The results of this approach show its benefits at low SNRs (8 and 15 dB) and it is also robust for unseen data.

Lately, in [120] the authors described a modified version of the VTS proposed by [1]. For this special case, the *unscented transform*, UT, is used to compute the first order VTS. The new nonlinear function is applied to a set of sampled points. The resulting transformed data is used to calculate the model parameters. The approach has shown improvements for low SNRs.

Table 4.1 compares the improvements obtained for 8-dB SNR. The *mean and variance normalization* approach is used as the baseline for the VTS i-vector approach. VTS is then used as the baseline for the UT transform. For clean-condition experiments, the system enrollment was performed using clean data, and the test is performed under noise conditions. For the multi-condition scenario, the enrollment phrases used the noisy data at the same SNR as the test phrases.

	VTS - ivector				*VTS - Unscented Transform*			
	clean		*multicondition*		*clean*		*multicondition*	
Approach	MVN	VTS	MVN	VTS	VTS	UT	VTS	UT
EER	15.5	5.2	5.9	3.2	6.3	5.9	4.2	3.6
Rel. Improvement	66		45		7		14	

Table 4.1: Comparison of VTS EER and relative improvement, 8 dB [1].

4.4.2.2 Acoustic Factor Analysis

Acoustic factor Analysis [2] considers the covariance information in the feature space to obtain robust models. The state-of-the-art *factor analysis* schemes employ *super-vector* space as the basis of the decomposition of the signal. Moreover FA uses a diagonal covariances of the GMM models believing that the MFCCs are uncorrelated. However, [121] shows that full covariances can also provide discriminative information of the speaker.

In contrast, this new approach shows that the MFCCs are not fully uncorrelated and that a better estimation of the parameters can be performed in the feature space instead of the super-vector space. The new solution strategy uses a *mixture probabilistic principal component analysis*, MPPCA [122], embedded into the FA approach. It follows the same idea as in FA, for $\chi = X_1, X_2, \cdots, X_T$, where X_i is the i^{th} vector in the sequence

$$X = LZ + \mu + \epsilon, \tag{4.2}$$

where X is a random variable of dimensions $P \times 1$ (the observation), μ is the mean vector, L refers to the loading matrix of $P \times R$ dimensions, Z is the acoustic factor of $R \times 1$ dimensions and ϵ is random noise with diagonal covariance matrix ψ. The latent vector Z follows a Gaussian distribution $Z \sim \mathcal{N}(0, \mathbb{I})$, $\epsilon \sim \mathcal{N}(0, \psi\mathbb{I})$ and $X \sim \mathcal{N}(\mu, \psi\mathbb{I} + LL^T)$. To capture the variations of multiple speakers and its corresponding phonemes and conditions, MPPCA is defined as,

$$P(X) = \sum_{i}^{M} w_i \mathcal{N}(\mu_i, \psi_i\mathbb{I} + L_i L_i^T), \tag{4.3}$$

where w_i, μ_i, ψ and L_i are the mixture weight, mean, noise covariance and the loading matrix of the i^{th} PPCA model and $\mathbb{I}$ is the identity matrix.

With this approach we obtain a mixture-dependent transformation for the i-vector architecture. Then, we compute the UBM using the EM algorithm and use it to compute the MPPCA. Moreover, we can decide the number of axes we require (with the optimal number still under investigation), discarding the lower dimensions. Afterwards, we can compute the loading matrices and the factors with this new scheme.

Table 4.2 shows an example of the improvements obtained with respect to two baseline systems, one with full covariance and the other with diagonal covariance. q refers to the number of dimensions retained from 60-dimensional vectors, using an i-vector/PLDA scheme. AFA shows competitive results with respect to the current baseline.

	Baseline		AFA		
	full cov	*diag cov*	$q = 36$	$q = 42$	$q = 48$
EER	2.03	2.55	2.06	1.79	1.95
Rel. Improv	-	-	-1.1	12	4.1

Table 4.2: Comparison of AFA EER and relative improvement [2].

4.4.3 Front-end modification

Modification of the frontend signifies better models and accuracy. Although, the research in ASR can be extended to SV, there are few studies that center on robustness from an SV point of view. In this section, we give an example of such research that addresses the problem by modifying the front-end to produce new features.

4.4.3.1 Regularization of all-poles

The study presented in [3] describes a modification of the traditional MFCC-based front end. The approach replaces the DFT estimation with a weighted *Linear Prediction*, LP, with regularization. This research shows a deep analysis of the traditional LP, stressing a comparison with the weighted LP. The experiments were performed under very noisy conditions (for example, under 10 dB factory noise and babble noise). For every case, this new modified front-end improves the baseline.

The main idea of the regularized LP [123] is that it gives a penalty for rapid changes in an all-pole spectral envelopes. SV benefits from the idea that the spectral models are smoothen so that mismatch between training and test is reduced.

Table 4.3 depicts an example of the results for different LP-based approaches under babble noise at 10 dB. The LP approach outperforms the FFT-baseline system. Moreover, the *stabilized weighted* LP (SWLP) and the *minimum variance distortionless response*, MVDR,[1] also improve the current baseline showing the potential of the LP approaches.

[1]Further details about this approach are described in [3]

	Baseline FFT	*LP*	*SWLP*	*MVDR*
EER	21.28	20.36	19.69	19.68
Rel. Improv	-	4.3	7.4	7.5

Table 4.3: Comparison of LP approaches EER and relative improvement, 10 dB [3].

4.5 Summary

This chapter describes the basic background and latest approaches behind robustness from an SV point of view. We observe how noise degrades the signal process in every stage, producing undesired effects in the performance. However, the research on this area is getting attention and there is an increasing number of research that tries to alleviate the effect of the noise. The noise can be treated with the usual and known solutions given by ASR research field (speech enhancement, feature vector normalization and acoustic model adaptation). But the SV community has also proposed modifications to the current system algorithms that can handle the noise. We highlight the approaches based on *factor analysis*, which have shown to improve the performance metrics.

As shown in this section, the algorithms for robust ASR motivated other studies in SV in the same directions. Our research is a fusion of both ideas, which can be seen as a vector transformation and an acoustic model adaptation. The next chapters present these ideas to improve the current known algorithms.

Chapter 5

Experimental Framework

> You're never given a dream without also being given the power to make it true.
>
> - Richard Bach *The Adventures of a Reluctant Messiah*

There are plenty of reasons to use NIST databases as the starting point of the SV research. Nowadays, the performance of the state-of-the-art systems compare the results in terms of curves and efficiency measures. Besides, the comparisons also take place at several stages of the process, for example: feature extraction, modeling and evaluation. The NIST databases provide a reliable and homogenous environment for such comparisons. Moreover, the databases have been of help to the research community when building complex systems where the constituent parts belong to different research groups.

The current chapter is entirely dedicated to the NIST databases.[1] We start by describing the current infrastructure used to develop our SV system. Secondly, we briefly describe the characteristics of the current NIST databases. Next, this chapter shows the traditional *experimental setup* that we used for the baseline experiments and that will be used for the proposed approaches along this thesis. Lastly, we give the initial baseline results for this setup.

5.1 Current infrastructure

The following functional cluster was used for such challenging project with a limited infrastructure (see Figure 5.1) composed of: Autonomous Beowulf cluster with 310 CPUs, with the following machines: Pentium 4 HT 3.2 GHz ; Xeon W3530 2.80GHz ; Xeon

[1] Most of the state of the art baseline results and comparisons with other systems are performed using this set of databases [39, 98].

E5645 2.40GHz AMD Opteron 6272 2GHz ; AMD Opteron 6276 2.6GHz ; Core2 Duo E8400 3.00GHz ; 1Gbps LAN 12TB storage. Software: SGE, Matlab 7.11.0.584, Python 2.6.6, Perl v5.10.1, x86_64 GNU-Linux.

The code was implemented in a parallelized mode in which all the computers can perform a reliable computation.

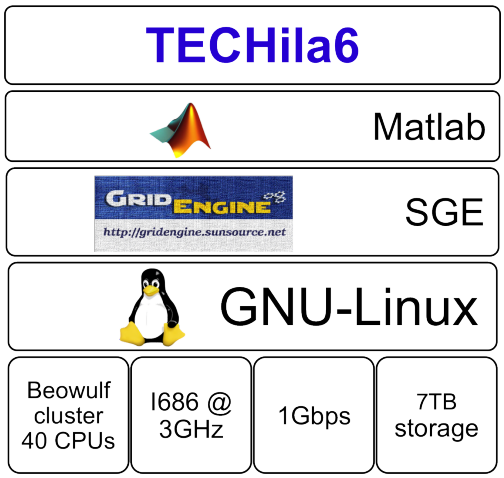


Figure 5.1: Infrastructure used by the Speaker Verification System

5.2 NIST database characteristics

A great effort has been done by the National Standard Institute and Technology to promote the research on this field [116].[2] Every two years the community organizes an evaluation among participants around the world. Each evaluation addresses the lately challenges. From the beginning, the plan was to record a database for a text-independent purpose. The first tasks included only a few trials and the challenge was to generate good models for each target speaker. Over the years, the databases have increased in target speaker and trials number. Moreover, for the mismatch conditions (different conditions for the training and test) is common to include several kinds of microphones and telephones. The latest evaluations included different types of vocal effort (low, high and medium) as well. The last one also included noise at different signal-to-noise ratios.

Before 2012 the rules were strict on not using speech signals from other target users to model a certain speaker. In the recent evaluation, the competition allowed the use of this information.

[2]Almost every new algorithm has to be tested using these databases, the organism provided to document improvements. Since 1996 the competition has tried to perfect SV systems, going from the development of better models of the target speaker progressively including a huge amount of trials (of the order of one million) and most recently encompassing noise. The tasks includes harder goals to achieve.

To have an idea of the databases, a summary of the conditions presented is shown in Table 5.1.

train condition	*test condition*	*channel*
Interview	interview	same microphone
Interview	interview	different microphone
telephone	telephone	matched condition
telephone	telephone	mismatched condition
Interview	telephone	mismatch channel condition
telephone	interview	mismatch channel condition

Table 5.1: Table presenting the common conditions

Lastly, every evaluation presents an increment in the number of trials as part of the concern of how real imposters try to access the verification systems. The last evaluation included a set of one million trials.

For the noisy conditions we employed Aurora II database [124]. This database has been widely employed for ASR, it includes the TI-digits in a clean scenario and TI-digits in noisy conditions. As part of this work, we selected babble noise to be added to the clean recordings using the noise adding tool, *FaNT* in [125]. The noisy recordings are then employed to test the baseline and our proposed techniques.

5.3 NIST database baseline

In this section, we introduce the *experimental setup*. We show further details about the actual datasets used and the specific feature extraction. We also present a compilation of the most relevant results.

We focus our research on the core-core evaluation NIST 2008, which is a defined task. We employed the NIST Speaker Evaluation 2004, 2005, 2006, 2008 and 2010 databases [39] to complete this study. The training data selection consists of building a UBM using the recordings in databases 2004, 2005, 2006 and 2010. Databases 2004 to 2006 provided sufficient information to build models for telephone conditions. Database 2010 granted different microphone and mismatch conditions to our experiments. Database 2008 tested the baseline framework and the new approaches. From this database, the target speakers belonging to the core-core evaluation were used to train specific user models. Lastly, the set of trials (from database 2008) evaluated the reliability of those models and the system.

The training and testing data sets are composed of either one two-channel telephone conversation of approximately five minutes total duration, with the (target or proposed trial) channel designated previously, or a microphone conversation segment of three to fifteen minutes involving both the interviewee (target speaker) and an interviewer. The data

also included various types of microphones (seven) and both conversational and interview sessions.[3] The common conditions used are described in Section 5.2 and Table 5.1.

For the set of experiments, we adopted the NIST evaluation restrictions (for example, neither choosing other target models as imposter models, nor using other target data to estimate the current target models).[4] Following the NIST 2008 Evaluation rules, the probability of being a target, P_{target}, is 0.01 and the probability of being a impostor, $P_{impostor}$, is 0.99.

5.3.1 Front-end

For the feature extraction, a short-time 256-pt Fourier analysis is performed on a 25-ms analysis window and 10ms frame rate. The feature vector (token) consists of 49 Mel Frequency Cepstral Coefficients (MFCCs), including delta and double delta coefficients, as shown in Table 5.2. The next step is to apply a feature warping normalization to undo the distortion caused by the channel. This warping is accomplished by first assembling an empirical CDF (cumulative distribution function) from the ranked features within 1.5 seconds after and before the current frame (3 seconds total), and then perform the CDF-inverse at the current frame. We included a frame removal criterion that encloses the concept of eliminating the low energy frames that do not provide information about the identity of the person. Low and 70% of the medium log-energy frames were simply discarded

Cep. Coeff	16
logE	1
Δ Cep	16
$\Delta\Delta$ Cep	16

Table 5.2: Table presenting the MFCC computation

5.3.2 UBM generation

A gender-dependent and target-independent 512-mixture GMM UBM model was trained from the core-core of NIST-SRE 2004, 2005, 2008 and 2010 databases.[5] Note that databases 2008 and 2010 training include several types of microphones that were used to test the system against mismatched conditions. The EM algorithm was used to obtain maximum likelihood estimates of the GMM parameters. For every iteration of EM, the system randomly polls 80% of the training tokens, corresponding to approximately 15 hours of speech. The UBM

[3]More information about the types of microphones can be found in [39] and in [116].

[4]Although in recent evaluations the rules have become more flexible, employing the target speakers as part of the imposter models, we built imposter models independently from the target data.

[5]Usually, the NIST databases must be cleaned from silent recordings or recordings with any kind of problems. This process is always performed in advance.

is first initialized using the K-means algorithm to obtain a set of 512 centroids. By using the k-means the performance of EM was simplified, however, it is always important to check that the local bounds are not very restrictive so that EM can provide a satisfactory estimation. The EM algorithm is then repeated after the model had converged (10 iterations).

5.3.3 Speaker Modeling

For each target speaker the current UBM is adapted using an MAP algorithm (using 2 to 5 iterations). For the special case of the NIST dabase 2008, a recording file for each speaker was used to perform adaptation. In the next sections, we present the results for our approaches, but MAP models are the starting point for all of them.

5.3.4 Baseline results

Table 5.3 presents the baseline results obtained using traditional techniques for database 2008. We just give an example for the simplest case using the GMM approach. 512 Gaussian components were used for this purpose. We observe that the results for mismatched conditions are worse than the ones for matched conditions. Moreover, the best results are obtained by the telephone set, probably because of the amount of telephone data used in the training.

	Female	*Male*	*Average*
1 *Interview interview same mic*	14.0	14.9	14.4
2 *Interview interview different mic*	18.3	18.2	18.2
3 *Interview tel*	17.9	18.8	18.3
4 *Tel tel*	12.5	13.0	12.7
Average	15.6	16.2	15.9

Table 5.3: Table presenting the final results (EER) on the Test set for NIST 2008

5.4 NIST database baseline - *JFA*

The starting point of JFA is definitely the traditional approach. The frontend remains the same, computing 16-dimensional MFCCs including deltas and double deltas (see Table 5.2). Next, we compute a gender-dependent and target-independent model, UBM, from a pool of raw speech (NIST Speaker Evaluation 2004, 2005, 2006 and microphone recording from 2010 core database).

For the JFA baseline, the speaker and channel factors were learned from a pool of recordings adapted to individual speakers. Apart from NIST databases, already described, other databases used for the training stage are Switchboard-1 (SW1) and Switchboard-2

(SW2) [117]. These two databases contain telephone recordings. From an extensive set, we selected a subset of 2480 files for the SW1 and 3108 for SW2. These databases were collected as part of the first efforts to have spontaneous speech for a specific speaker and for different duration time. This selection remains for the rest of our experiments, unless stated otherwise.

The detailed descriptions of the databases employed to train the loading matrices V, U, and D, are shown in Table 5.4 and 5.5.[6]

V estimation set	U estimation set
NIST 05 Tel. 2158	NIST 05 Tel. 1429
NIST 05 Mic. 600	NIST 05 Mic. 600
NIST 10 Mic. 461	NIST 10 Mic. 261
SW 1 R 2 Tel. 1240	SW 1 R 2 Tel. 660
SW 2 P-I Tel. 3108	SW 2 P-I Tel. 1636

Table 5.4: JFA training set for female

V estimation set	U estimation set
NIST 05 Tel. 2201	NIST 05 Tel. 1216
NIST 05 Mic. 600	NIST 05 Mic. 600
NIST 10 Mic. 600	NIST 10 Mic. 345
SW 1 R 2 Tel. 2480	SW 1 R 2 Tel. 1314
SW 2 P-I Tel. 3108	SW 2 P-I Tel. 1627

Table 5.5: JFA training set for male

Finally, 570,000 trials (female and male) were used to evaluate system performance.

The JFA algorithm implementation by the Speech Processing Group at the Brno University of Technology [27] was adapted to our infrastructure needs and capabilities. The basic implementation has been shown to be successful in part or whole by [99, 94, 22].

From the complete dataset shown in Table 5.4 and 5.5, we compute 600 eigenvoices and 500 eigenchannels for each gender modeling. The variability term was trained in the same way as the speaker factors. The baseline results are depicted in Table 5.6, showing an expected improvement with respect to the traditional ML approach.

	Female	*Male*	*Average*
1 *Interview interview same mic*	10.7	11.1	10.9
2 *Interview interview different mic*	13.5	13.7	13.6
3 *Interview tel*	13.5	14.1	13.8
4 *tel tel*	9.8	10.2	10.0
Average	11.9	12.2	12.07

Table 5.6: EER results on the Test set for NIST 2008 for JFA approach

[6]This configuration was selected after the study performed by [126].

5.5 NIST database baseline - *MVE*

Once again as in JFA, the starting point is the traditional approach. The system first computes the 16-dimensional MFCCs including deltas and double deltas (see Table 5.2) for all the databases. Next, the training stage calculates the UBM and the MAP-adapted model for every target speaker.

Generally, discriminative algorithms employ positive and negative examples to optimize the objective function, meaning that the optimization embraces both models: target and imposter. MVE is not an exception; several sets are constructed according to a *cohort* selection. A cohort sample is a negative sample of the current target model. Although a cohort sample is any sample that does not belong to the target data set, usually its computation is based on some other criteria such as the likelihood with respect to the target model, or the distance between the negative sample and the target model.

At first, the selection was performed randomly, as described in the following selection, but later we found that just a small set can be used to provide a reliable optimization.

5.5.1 Cohort Set selection

For every experiment, the cohorts were selected randomly, 500 phrases from NIST 2004, 2005 2006, switchboard 1 and switchboard 2; moreover, we selected 500 phrases from NIST 2010 microphone (interview). A total 3000 phrases were used as imposter samples. For every experiment, we used a different seed to select the phrases as a starting point.[7]

The results for the MVE configuration are shown in Table 5.7. There is a considerable improvement if compared to the traditional approach. However, the results are not as good as the ones presented by JFA. Once again, the best results belong to the matched conditions, performing even better for the telephone-telephone scenario.

	Female	*Male*	*Average*
1 *Interview interview same mic*	12.2	12.6	12.4
2 *Interview interview different mic*	14.8	15.0	14.9
3 *Interview tel*	15.1	15.4	15.2
4 *tel tel*	11.2	11.7	11.4
Average	13.3	13.6	13.5

Table 5.7: EER results on the Test set for NIST 2008 for the MVE approach

[7]We performed experiments with the same setup, but using a different seed for the random cohort selection. The results didn't present any considerable difference.

5.6 Summary

This chapter shows the *experimental setup* used to obtain the baseline results in the NIST database that will be used along the following chapters. The description starts from processing the audio signal up to the final step when the system outputs a performance measure (EER). We present baseline outcomes for the traditional approaches, JFA and MVE. JFA shows the best performance, having lower EER than MVE and the traditional approach. MVE improves the traditional approach but didn't obtain results as good as JFA. These set of outcomes give us a clear idea of the performance of the system and represent the starting point of our research.

Chapter 6

Optimization of the area under the ROC curve

> Knowledge can be communicated, but not wisdom. One can find it, live it, be fortified by it, do wonders through it, but one cannot communicate and teach it.
>
> - Hermann Hesse *Siddartha*

System performance in SV is measured by considering the *false acceptance* and *false rejection* errors. Both errors are closely related and a tradeoff between them is always considered; *i.e.*, the systems select an appropriate separation threshold between the distributions that describe that binary classifier. Traditional methods proposed a clever way to plot both errors in a curve called the *detection error tradeoff* (DET) curve. This performance curve is a wrapped representation of the Receiver Operating Characteristic (ROC) Curve. In this chapter, we explore the *area under the curve* of both representations: the ROC curve and the effect in the metrics such as the DET curve and the minDCF. We also show a mathematical formulation to optimize the model parameters based on those curves using a discriminative approach. First, we give a preliminary explanation about the ROC and how it relates to the DET curve. Then, we detail our algorithm and its formulation.

6.1 The Receiver Operating Characteristic Curve

For a binary classifier, the ROC curve analysis provides the tools to measure system performance [127]. First, we define two classes as in previous sections: positives (targets) and negatives (imposters). The errors are the same as in our previous definitions, but we will use a different nomenclature for clarity in this section. If a positive is classified as a positive, it is called true positive (TP). If a negative is classified as negative, it is named

true negative. If a positive is classified as a negative, it is called false negative, miss or false rejected. If a negative is classified as a positive, it becomes a false positive (FP), false alarm or false accepted.

The ROC curve, then, plots the relation between false positive rate vs true positive rate, see Figure 6.1. Let us consider the two probability distributions that define the classification of the scores, we compute each operating point by moving the score threshold from minus infinity to infinity. In fact, it plots the tradeoff between false alarm rate in the interval $[0, 1]$ and true positive percentage in the interval $[0, 1]$ as a convex function.

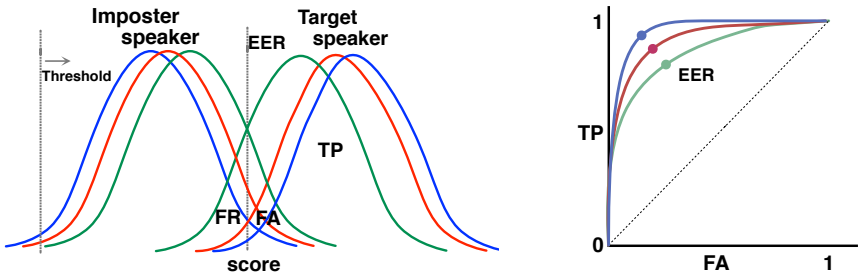


Figure 6.1: Score distributions for three different classifiers.

The EER of this graph is the summary of all the operating points, and it is where $P_{FA} = P_{miss}$. The closer the curve is to the $(0, 1)$ point the better the classifier performs.

6.2 Area under the ROC curve

A single operating point on the ROC curve is not enough to describe the efficiency of the system. The area under the ROC curve is a more sensible measure of classifier performance, as stated in [127]. Moreover, it is always useful when comparing two or more binary classifiers and want to use a single scalar, that can give information of all the operating points.[1]

The properties of the area under the ROC curve can be related to the Wilcoxon-Mann-Whitney (WMW) statistic. The AUC lies on the interval (0,1), as it is a unit square. The 45 degrees diagonal line describes a random classifier with area 0.5. Hence, no classifier should have area less than 0.5. In statistical terms, the AUC is the probability of a classifier to score a random positive sample more times than random negative samples. This statement is the bridge to the WMW statistic [25, 128].

The statistic is usually defined as,

$$\Upsilon = \frac{\sum_{\chi \in \mathcal{H}} \sum_{\hat{\chi} \in \mathcal{W}} 1(F(\chi) > F(\hat{\chi}))}{|\mathcal{H}||\mathcal{W}|}, \tag{6.1}$$

[1]However, we must not guarantee that the higher the AUC the better, there are cases where the classifier works better in certain regions (depending on the desired application or the classifier design).

where 1 is the indicator function. $\mathcal{H}$ and $\mathcal{W}$ are two sets of data belonging respectively to the target score set or impostor score set, and $F(\chi)$ determines if any data instance χ belongs once again to the target or impostor. The statistic is equivalent to the AUC of the ROC curve.

Another way to look at this statistic is as a joint probability of the target and impostor scores over a region [129] as,

$$P(F(\chi) > F(\hat{\chi})) = \int_{-\infty}^{\infty} P(F(\chi)) \int_{-\infty}^{F(\chi)} P(F(\hat{\chi}))\, dF(\hat{\chi})\, dF(\chi). \tag{6.2}$$

Both formulations are similar and present a convenient figure of merit. There are other similar approximations of the ROC curve as the ROCCH (ROC convex hull) [130]. The latter is considered an interpolation of the ROC curve operating points, building an actual *convex hull* of those points. Although this new approach deserves more attention, and has been explored in [129], it has not been examined in this thesis.

With this thesis, we will deal with the computation of the AUC that can be expressed as a metric that describes the performance of the system. An easy and direct way is just to compute Υ in Equation 6.1, or using Equation 6.2. In this research we will use Υ and give results according to it.

6.3 From ROC to DET Curve

The *Detection Error Tradeoff* (DET) curve is closely related to the ROC curve. It plots *false acceptance* probability against the *miss* probability (instead of TP as in ROC curve) to observe the errors at the same time. Moreover, it warps the curve at the edges so that it makes the curves look linear when Gaussian [76]. The warping is performed using a quantile function of the normal distribution (probit function),

$$Q(a) = \text{probit}(a) = \sqrt{2}\ \text{erf}^{-1}(2a - 1), \tag{6.3}$$

where a takes the values of P_{FA} (for the x -axis) and P_{miss} (for the y-axis) and erf^{-1} is the *inverse error function*. The interval of the ROC curve $[0, 1]$ is then transformed to $[-\infty, \infty]$. The DET curve is usually limited to a window of interest defined by the P_{miss} and P_{FA}, decided in advance.

Similar to the ROC curve, the complete DET curve is built by moving the threshold at each operating point of the FA and miss probabilities. In this sense, we can compare two or more systems in the same plane over all the operating points as is shown in Figure 6.2. Once again, the minDCF and the EER can be plotted in these curves to have a complete performance of the actual classifier.

6.4 Optimization of the area under the curve

We note that some of the traditional learning methods outlined in Sections 2.2 and Chapter 3 train the distributions of the individual classes S and $\bar{S}$ without explicitly considering the fact that the actual task is one of discrimination. Needless to say, discriminative versions of these learning algorithms also exist.

Discriminative [18] and maximum margin [131] methods for both learning and adaptation of GMM parameters have been proposed in the literature. Similarly, discriminative versions of JFA and its variants have also been proposed, *e.g.* [57]. All of these methods attempt to learn model parameters to maximize the empirical classification error on the training data provided. Consequently, they naturally optimize the performance at a *specific* operating point – that characterized by the relative (possibly weighted) proportions of positive and negative examples provided to them. This operating point may not be related at all to the actual operating point at which the system is actually eventually evaluated. Moreover, we would ideally not like to *commit* to an operating point – we would like to optimize the performance at *every* operating point. In other words, we do not merely wish to improve the performance of a single point in the ROC curve, such as the EER, and affect the DET curve (see Figure 6.2); we would like to explicitly optimize the *entire* ROC curve. The way we will do this is by explicitly optimizing the AUC – the area under the ROC curve.

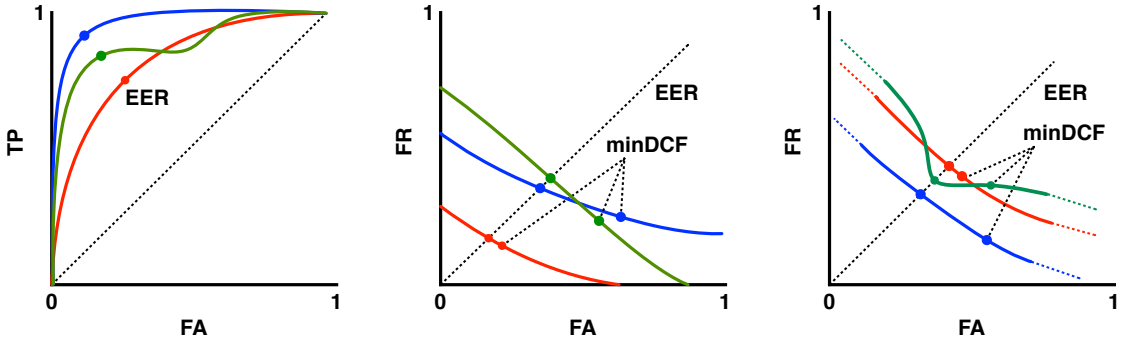


Figure 6.2: ROC and DET curves for three different classifiers.

A way to address the problem, is to optimize the AUC of the ROC curve that is described by the WMN statistic and document the improvements on the DET curve.

Consider a binary classifier that attempts to classify the data as belonging to a class C or not. Once again, let $\mathcal{H}$ and $\mathcal{W}$ be two sets of data belonging respectively to C and $\bar{C}$ (*i.e.* not in C). The empirical AUC of a classifier that computes a score $\theta(\chi)$ to determine if any data instance χ belongs to C is given by Equation 6.4.

$$\Upsilon(\Lambda) = 1.0 - \frac{\sum_{\chi \in \mathcal{H}} \sum_{\hat{\chi} \in \mathcal{W}} 1(\theta(\chi) > \theta(\hat{\chi}))}{|\mathcal{H}||\mathcal{W}|} \tag{6.4}$$

In the equation above we have written $\Upsilon(\Lambda)$ to be a function of $\Lambda = \Lambda_C \cup \Lambda_{\bar{C}}$, where Λ_C and $\Lambda_{\bar{C}}$ are the parameters of any models associated with C and $\bar{C}$, to aid us in the explanation below.

We can optimize the AUC by employing the function given in Equation 6.4. Equation 6.4, computed over the training set, is thus a new objective function to be minimized. The AUC function specified by Equation 6.4 is not smooth, however, since it is the sum of discontinuous indicator functions. In order to optimize it, a smooth, differentiable version of this objective function is needed.

To do so, we replace the indicator functions $1(a > b)$ by a sigmoid function, following an approach that is commonly used in discriminative training methods, *e.g.* [24], as in Equation 6.5,

$$R(a,b) = \frac{1}{1 + \exp\left(-\gamma \ \varphi(a,b)\right)}, \tag{6.5}$$

where γ governs the steepness of the sigmoid and controls the learning rate and φ is the distance $\varphi(a,b) = a - b$.

Thus, the modified AUC function to optimize is:

$$\Upsilon(\Lambda) = 1.0 - \frac{\sum_{\chi \in \mathcal{H}} \sum_{\hat{\chi} \in \mathcal{W}} R(\theta(\chi), \theta(\hat{\chi}))}{|\mathcal{H}||\mathcal{W}|}. \tag{6.6}$$

The modified AUC function of Equation 6.6 must be appropriately customized to the type of model being considered. It can then be optimized using the generalized probabilistic descendent (GPD) algorithm. Let $\mathbf{X}$ represent the complete set of training instances: $\mathbf{X} = \mathcal{H} \cup \mathcal{W}$. The GPD updates are performed according to the following:

$$\Lambda_{t+1} = \Lambda_t - \epsilon \nabla \Upsilon(\mathbf{X}, \Lambda), \tag{6.7}$$

$$\nabla \Upsilon(\mathbf{X}, \Lambda) = -\frac{1}{|\mathcal{H}||\mathcal{W}|} \sum_{\chi \in \mathcal{H}} \sum_{\hat{\chi} \in \mathcal{W}} \gamma R(1-R) \left[\frac{\partial \theta(\chi)}{\partial \Lambda} - \frac{\partial \theta(\hat{\chi})}{\partial \Lambda} \right]. \tag{6.8}$$

In the above Equation R is a short-hand notation for $R(\theta(\chi), \theta(\hat{\chi}))$. ϵ is a learning rate parameter.

The above formalism can generally be used in all formulations of speaker verification with appropriate customization of the objective function. Below we consider two approaches as an illustration.

6.5 Minimum Verification Error

In the minimum-verification error approach, GMM parameters for individual speakers are optimized to minimize empirical verification error on the training set [89]. Replacing the

objective function usually used in MVE training, we get the modified objective function:

$$\Upsilon(\Lambda) = 1.0 - \frac{\sum_{\chi\in\mathcal{H}}\sum_{X\in\chi}\sum_{\hat{\chi}\in\mathcal{W}}\sum_{\hat{X}\in\hat{\chi}} R(\theta(X),\theta(\hat{X}))}{\sum_{\chi\in\mathcal{H}} L_\chi \sum_{\hat{\chi}\in\mathcal{W}} L_{\hat{\chi}}}, \tag{6.9}$$

where L_χ is the number of feature vectors in χ. Note that the original AUC formulation of Equation 6.4 only considers misclassifications – both FA and FR, in keeping with the conventional formulation of MVE. The "soft" version of the AUC given by Equation 6.9 also naturally conforms to this formulation.

Using the representation

$$\sum_{\chi\in\mathcal{H}} L_\chi = |\mathcal{H}| \tag{6.10}$$

$$\sum_{\chi\in\mathcal{W}} L_\chi = |\mathcal{W}|, \tag{6.11}$$

the GPD update rule for any parameter ϕ of the distributions is now given by $\phi_{t+1} = \phi_t - \epsilon\nabla_\phi\Upsilon(\mathbf{X},\Lambda)$, where

$$\nabla_\phi\Upsilon\left(\mathbf{X},\Lambda\right) = -\frac{1}{|\mathcal{H}||\mathcal{W}|}\sum_{\chi\in\mathcal{H}}\sum_{X\in\chi}\sum_{\hat{\chi}\in\mathcal{W}}\sum_{\hat{X}\in\hat{\chi}}\gamma R(1-R)\nabla_\phi l(X,\hat{X},\Lambda), \tag{6.12}$$

and $\nabla_\phi l(X,\hat{X},\Lambda)$ is a local gradient with respect to ϕ at $\chi,\hat{\chi}$ and has the form given by Equation 6.13,

$$\nabla_\phi l(X,\hat{X},\Lambda) = -\frac{\partial\theta(X)}{\partial\phi} + \frac{\partial\theta(\hat{X})}{\partial\phi}. \tag{6.13}$$

$\frac{\partial\theta(X)}{\partial\phi}$ represents the derivative of the log-likelihood-difference given by the Gaussian mixture models for the target speaker and the universal background model for vector X with respect to ϕ. The update rules for the individual parameters w_k^S, μ_k^S and Σ_k^S are obtained by plugging in $\frac{\partial\theta(X)}{\partial w_k^S}$, $\frac{\partial\theta(X)}{\partial\mu_k^S}$ and $\frac{\partial\theta(X)}{\partial\Sigma_k^S}$ respectively into Equation 6.13. These equations are relatively straightforward to derive, (please refer to Appendix C for further details).

For the purpose of this chapter we will just include the final update equations. Once again, consider μ as an example, then, $\tilde{\mu} = \mu/\sigma$, Let the misverification measure be

$$\frac{\partial\theta(X)}{\partial\phi_k} = \left(-\frac{\partial g(X;\Lambda)}{\partial\phi_k} + \frac{\partial G(X;\Lambda)}{\partial\phi_k}\right)\mathrm{sgn}(c), \tag{6.14}$$

where $\mathrm{sgn}(c)$ is $+1$ if it is a target and -1 if it is an imposter.

Moreover,

$$\frac{\partial g(\chi;\Lambda)}{\partial \Lambda_k} = \frac{1}{p(\chi;\Lambda)} \frac{\partial\, p(\chi;\Lambda)}{\partial \Lambda_k} \quad \text{and} \quad \frac{\partial G(\chi;\Lambda)}{\partial \Lambda_k} = \frac{1}{p(\chi;\hat{\Lambda})} \frac{\partial\, p(\chi;\hat{\Lambda})}{\partial \hat{\Lambda}_k}. \tag{6.15}$$

Then for μ, a specific k, dimension d and to avoid bias, let $\mu = \sigma\,\tilde{\mu}$. Hence,

$$\nabla_\phi \Upsilon\left(\chi, \tilde{\mu}_k\right) = -\frac{1}{|\mathcal{H}||\mathcal{W}|} \sum_{\chi\in\mathcal{H}} \sum_{X\in\chi} \sum_{\hat{\chi}\in\mathcal{W}} \sum_{\hat{X}\in\hat{\chi}} \gamma R(1-R) \frac{w_k\, \mathcal{N}(\chi|\tilde{\mu}_k, \sigma_k)}{\sum_{k'} w_{k'}\, \mathcal{N}(\chi|\tilde{\mu}_{k'}, \sigma_{k'})} \left(\frac{\chi}{\sigma_k} - \tilde{\mu}_k\right). \tag{6.16}$$

For σ, let $\tilde{\sigma} = \log(\sigma)$. Then, the final equation to compute σ_k is

$$\nabla_\phi \Upsilon\left(\chi, \tilde{\sigma}_k\right) = -\frac{1}{|\mathcal{H}||\mathcal{W}|} \sum_{\chi\in\mathcal{H}} \sum_{X\in\chi} \sum_{\hat{\chi}\in\mathcal{W}} \sum_{\hat{X}\in\hat{\chi}} \gamma R(1-R) \frac{w_k\, \mathcal{N}(\chi|\mu_k, \tilde{\sigma}_k)}{\sum_{k'} w_{k'}\, \mathcal{N}(\chi|\mu_{k'}, \tilde{\sigma}_{k'})} \left\{ \left(\frac{\chi - \mu_k}{\tilde{\sigma}_k}\right)^2 - 1 \right\}. \tag{6.17}$$

Finally, for the set of weights w_k and to ensure that $\sum_{k=1}^{K} w_k = 1$,let, $w = \frac{\exp(\tilde{w})}{\sum_{k=1}^{K} \exp(\tilde{w})}$

$$\nabla_\phi \Upsilon\left(\chi, \tilde{w}_k\right) = -\frac{1}{|\mathcal{H}||\mathcal{W}|} \sum_{\chi\in\mathcal{H}} \sum_{X\in\chi} \sum_{\hat{\chi}\in\mathcal{W}} \sum_{\hat{X}\in\hat{\chi}} \gamma R(1-R) \nabla_\phi l(X, \hat{X}, \Lambda) \frac{\mathcal{N}(\chi|\mu_k, \sigma_k)}{\sum_{k'} \tilde{w}_{k'}\, \mathcal{N}(\chi|\mu_{k'}, \sigma_{k'})}. \tag{6.18}$$

Algorithm 6.1. Optimization of the ROC curve, MVE I

Initialization

1. Train the basis imposter or cohort model (usually employing EM) and the target speaker model (using MAP)

$$\bar{\Lambda} = \{\bar{W}_k^C, \bar{\mu}_k^C, \bar{\Sigma}_k^C\}, \tag{6.19}$$

$$\Lambda = \{W_k^C, \mu_k^C, \Sigma_k^C\}. \tag{6.20}$$

2. Compute the misverification distance and the combined loss function for the FA and FR errors using,

$$d_n(\chi) = -g_n\left(\chi;\Lambda\right) + G_n\left(\chi;\Lambda\right) \operatorname{sgn}(c), \tag{6.21}$$

and

$$\Upsilon(\Lambda) = 1.0 - \frac{\sum_{\chi\in\mathcal{H}} \sum_{X\in\chi} \sum_{\hat{\chi}\in\mathcal{W}} \sum_{\hat{X}\in\hat{\chi}} R(\theta(X), \theta(\hat{X}))}{\sum_{\chi\in\mathcal{H}} L_\chi \sum_{\hat{\chi}\in\mathcal{W}} L_{\hat{\chi}}}. \tag{6.22}$$

Algorithm 6.1. Optimization of the ROC curve, MVE II

Parameter Estimation

For a specific class C, a component k in the GMM, dimension d. and a set of training X_t where $t = 1...T$.

1. Obtain the gradients of the *sigmoid loss function* for each of the model parameters: $\{w_k^C, \mu_k^C, \Sigma_k^C\}$ and plugged them into:

$$\Lambda_{t+1} = \Lambda_t - \epsilon \nabla \Upsilon(\chi; \Lambda), \tag{6.23}$$

$$\nabla_\phi \Upsilon\left(\mathbf{X}, \Lambda\right) = -\frac{1}{|\mathcal{H}||\mathcal{W}|} \sum_{\chi \in \mathcal{H}} \sum_{X \in \chi} \sum_{\hat{\chi} \in \mathcal{W}} \sum_{\hat{X} \in \hat{\chi}} \gamma R(1-R) \nabla_\phi l(X, \hat{X}, \Lambda). \tag{6.24}$$

   Recall, $\nabla_\phi l(X, \hat{X}, \Lambda) = \left(-\frac{\partial g(X;\Lambda)}{\partial \phi_k} + \frac{\partial G(X;\Lambda)}{\partial \phi_k}\right) \operatorname{sgn}(c)$.

2. Optimize the models for every class:

   (a) For a specific k, dimension d and to avoid bias, let $\mu = \sigma\, \tilde{\mu}$. Then,

$$\frac{\partial p(\chi; \Lambda)}{\partial \tilde{\mu}_k} = \frac{w_k\, \mathcal{N}(\chi | \tilde{\mu}_k, \sigma_k)}{\sum_{k'} w_{k'}\, \mathcal{N}(\chi | \tilde{\mu}_{k'}, \sigma_{k'})} \left(\frac{\chi}{\sigma_k} - \tilde{\mu}_k\right). \tag{6.25}$$

   (b) let, $\tilde{\sigma} = \log(\sigma)$, Then σ is

$$\frac{\partial p(\chi; \Lambda)}{\partial \tilde{\sigma}_k} = \frac{w_k\, \mathcal{N}(\chi | \mu_k, \tilde{\sigma}_k)}{\sum_{k'} w_{k'}\, \mathcal{N}(\chi | \mu_{k'}, \tilde{\sigma}_{k'})} \left\{ \left(\frac{\chi - \mu_k}{\tilde{\sigma}_k}\right)^2 - 1 \right\}. \tag{6.26}$$

   (c) For w_k and to ensure that $\sum_{k=1}^K w_k = 1$, let, $w = \frac{\exp(\tilde{w})}{\sum_{k=1}^K \exp(\tilde{w})}$,

$$\frac{\partial p(\chi; \Lambda)}{\partial \tilde{w}_k} = \frac{\mathcal{N}(\chi | \mu_k, \sigma_k)}{\sum_{k'} \tilde{w}_{k'}\, \mathcal{N}(\chi | \mu_{k'}, \sigma_{k'})}. \tag{6.27}$$

Check for convergence

1. Compute the loss function with the new parameters:

$$\Upsilon(\Lambda)_{new} = 1.0 - \frac{\sum_{\chi \in \mathcal{H}} \sum_{X \in \chi} \sum_{\hat{\chi} \in \mathcal{W}} \sum_{\hat{X} \in \hat{\chi}} R(\theta(X), \theta(\hat{X}))}{\sum_{\chi \in \mathcal{H}} L_\chi \sum_{\hat{\chi} \in \mathcal{W}} L_{\hat{\chi}}}. \tag{6.28}$$

2. Compare ℓ_{new} with ℓ; if it is higher than a threshold τ iterate from *Parameter Estimation*.

6.6 JFA

In the case of JFA, the set of parameters to be learned are of two kinds. The *global* parameters include V, the speaker loadings, U, the channel loadings, and D, the diagonal error scaling matrix. The *specific* parameters include y_S, which is specific to a speaker S and $x_{S,H}$, which is specific to the speaker-channel combination S, H.

Hence, two distinct learning problems must be addressed: learning the global loadings from a large collection of speaker recordings over a variety of channels, and learning specific factors for individual speakers. The global parameters must learn the overall characteristics of the speaker and channel subspaces. Specific parameters must be learned to customize a model to a specific speaker given training data for the speaker.

The AUC objective function must be appropriately customized in each case. In all cases, the discriminant function $\theta()$ in Equation 3.1 is specified as in Equation 6.29.

$$\theta(\chi) = \log\ p(\chi; V, U, D, y_S(\chi), x_{H(\chi),S(\chi)}) - \log\ p(\chi; \lambda_{\bar{S}}, U, x_{H(\chi),S(\chi)}) \tag{6.29}$$

Here $S(\chi)$ represents the speaker S represented in the recording χ. $H(\chi)$ represents the recording channel in χ. The equation above explicitly indicates that the log-likelihood for the model of speaker S is computed using Gaussian parameters from the supervector

$$M = m + Vy_{S(\chi)} + Ux_{S(\chi),H(\chi)} + Dz, \tag{6.30}$$

whereas the parameters of the "imposter" model for any recording are obtained from $M' = m + Ux_{S(\chi),H(\chi)} + Dz$, which only considers the universal mean m adjusted by the channel factors which customize them to the recording χ. The global mean m is derived from the universal background model $\lambda_{\bar{S}}$.

6.6.1 Global parameters: Loadings estimation

Let $\mathbf{X}$ represent a large collection of recordings χ obtained from a large number of speakers. Let $\mathcal{S}$ be the set of all speakers represented in $\mathbf{X}$. Let $\mathbf{X}_S$ represent the subset of $\mathbf{X}$ representing recordings from speaker S and $\mathbf{X}_{\bar{S}}$ be recordings from remaining speakers, *i.e.* $\mathbf{X} = \mathbf{X}_S \cup \mathbf{X}_{\bar{S}}$. $\mathbf{X}$ can be partitioned in this manner in as many ways as there are speakers in $\mathcal{S}$.

To learn global parameters, we define the AUC objective function as given in Equation 6.31,

$$\Upsilon(\Lambda) = 1.0 - \frac{1}{|\mathbf{X}_S||\mathbf{X}_{\bar{S}}|} \sum_{S \in \mathcal{S}} \sum_{\chi \in \mathbf{X}_S} \sum_{\chi \in \mathbf{X}_{\bar{S}}} R(\theta_S(\chi), \theta_S(\hat{\chi})). \tag{6.31}$$

As before, the GPD update rule for any global parameter ϕ is given by $\phi_{t+1} =$

$\phi_t - \epsilon \nabla_\phi \Upsilon(\mathbf{X}, \Lambda)$, where

$$\nabla_\phi \Upsilon(\mathbf{X}, \Lambda) = -\sum_{S \in \mathcal{S}} \frac{\sum_{\chi \in \mathbf{X}_S} \sum_{\chi \in \mathbf{X}_{\bar{S}}} \gamma R(1-R) \nabla_\phi l(\chi, \hat{\chi}, \Lambda)}{|\mathbf{X}_S||\mathbf{X}_{\bar{S}}|}, \tag{6.32}$$

and $\nabla_\phi l(\chi, \hat{\chi}, \Lambda)$ is a local gradient with respect to ϕ at $\chi, \hat{\chi}$,

$$\nabla_\phi l(\chi, \hat{\chi}, \Lambda) = -\frac{\partial \theta(\chi)}{\partial \phi} + \frac{\partial \theta(\hat{\chi})}{\partial \phi}. \tag{6.33}$$

To derive the update rules for individual parameters, it is sufficient to obtain the derivatives $\frac{\partial \theta(\chi)}{\partial \phi}$ with respect to the corresponding parameters. This case was studied in [132] and it is extended in the equations given below to the special case.[2]

The formulation takes two stages (as in the usual JFA) to decouple estimation of V, U, Ψ. First, we will consider V estimation, maintaining U fixed, considering that $Q = VV^\top + \Psi$, where $\Psi = DD^\top$ and is diagonal.

Then for μ, a specific k, dimension d and to avoid bias[3], let $\mu_k = q_k \, \tilde{\mu}_k$, Hence,

$$\nabla_\phi \Upsilon(\chi, \tilde{\mu}_k) = -\sum_{S \in \mathcal{S}} \frac{\sum_{\chi \in \mathbf{X}_S} \sum_{\chi \in \mathbf{X}_{\bar{S}}} \gamma R(1-R)}{|\mathbf{X}_S||\mathbf{X}_{\bar{S}}|} \frac{w_k \mathcal{N}(\chi|\tilde{\mu}_k, q_k)}{\sum_{k'} w_{k'} \mathcal{N}(\chi|\tilde{\mu}_{k'}, q_{k'})} \left(\frac{\chi}{q_k} - \tilde{\mu}_k \right). \tag{6.34}$$

To estimate Q, first we present the expression for V and then Ψ; refer to Appendix C.2 that details about identities. We will also make the matrix dimensions explicit, considering that Ψ is diagonal. Then, q_k, let $\tilde{q} = \log(q)$.

Then, for V,

$$\nabla_\phi \Upsilon(\chi, \tilde{v}_{ij}) = -\sum_{S \in \mathcal{S}} \frac{\sum_{\chi \in \mathbf{X}_S} \sum_{\chi \in \mathbf{X}_{\bar{S}}} \gamma R(1-R)}{|\mathbf{X}_S||\mathbf{X}_{\bar{S}}|} \frac{w_k \mathcal{N}(\chi|\mu_k, q_k)}{\sum_{k'} w_{k'} \mathcal{N}(\chi|\mu_{k'}, q_{k'})} \left\{ [Q^{-1}(x-u)]_i [(x-u)^\top Q^{-1} V]_j - [Q^{-1}V]_{ij} \right\}. \tag{6.35}$$

Then, for ψ,

$$\nabla_\phi \Upsilon(\chi, \tilde{\psi}_{i,j}) = -\sum_{S \in \mathcal{S}} \frac{\sum_{\chi \in \mathbf{X}_S} \sum_{\chi \in \mathbf{X}_{\bar{S}}} \gamma R(1-R)}{|\mathbf{X}_S||\mathbf{X}_{\bar{S}}|} \frac{w_k \mathcal{N}(\chi|\mu_k, q_k)}{\sum_{k'} w_{k'} \mathcal{N}(\chi|\mu_{k'}, q_{k'})} \left\{ \frac{1}{2} \left(Q_{ii}^{-1} - [Q^{-1}x - \mu]_i^2 \right) \right\}. \tag{6.36}$$

Now, for U, we fix the current Q and estimate accordingly.

$$\nabla_\phi \Upsilon(\chi, \tilde{U}_k) = -\sum_{S \in \mathcal{S}} \frac{\sum_{\chi \in \mathbf{X}_S} \sum_{\chi \in \mathbf{X}_{\bar{S}}} \gamma R(1-R)}{|\mathbf{X}_S||\mathbf{X}_{\bar{S}}|} \frac{w_k \mathcal{N}(\chi|\mu_k, q_k)}{\sum_{k'} w_{k'} \mathcal{N}(\chi|\mu_{k'}, q_{k'})} \left\{ \left(\frac{\chi - \mu_k}{\tilde{U}_k} \right)^2 - 1 \right\}. \tag{6.37}$$

[2]We use the FA formulation presented in [132]; analogous to the Baum-Welch algorithm, in here we have previously derived the minimum classification approach.

[3]Our usual change of variables holds for this formulation.

Finally, for the set of weights w_k and to ensure that $\sum_{k=1}^{K} w_k = 1$, let, $w = \frac{\exp(\tilde{w})}{\sum_{k=1}^{K} \exp(\tilde{w})}$,

$$\nabla_\phi \Upsilon(\chi, \tilde{w}_k) = -\sum_{S \in \mathcal{S}} \frac{\sum_{\chi \in \mathbf{X}_S} \sum_{\chi \in \mathbf{X}_{\bar{S}}} \gamma R(1-R)}{|\mathbf{X}_S||\mathbf{X}_{\bar{S}}|} \frac{\mathcal{N}(\chi|\mu_k, q_k)}{\sum_{k'} \tilde{w}_{k'} \mathcal{N}(\chi|\mu_{k'}, q_{k'})}. \tag{6.38}$$

We would like to note that this process is replacing the current Baum and Welch estimation of the model parameters.

6.6.2 Estimating Specific Parameters

To learn the specific parameters for a particular speaker S, the AUC objective is defined simply as

$$\Upsilon(\Lambda_S) = 1.0 - \frac{\sum_{\chi \in \mathbf{X}_S} \sum_{\chi \in \mathbf{X}_{\bar{S}}} R(\theta_S(\chi), \theta_S(\hat{\chi}))}{|\mathbf{X}_S||\mathbf{X}_{\bar{S}}|}. \tag{6.39}$$

Note that unlike Equation 6.31, which includes an outer summation over all speakers, Equation 6.39 only considers a single speaker S.

As before, the GPD update rule for any specific parameter ϕ is given by $\phi_{t+1} = \phi_t - \epsilon \nabla_\phi L(\mathbf{X}, \Lambda)$, where,

$$\nabla_\phi \Upsilon(\mathbf{X}, \Lambda) = -\frac{\sum_{\chi \in \mathbf{X}_S} \sum_{\chi \in \mathbf{X}_{\bar{S}}} \gamma R(1-R) \nabla_\phi l(\chi, \hat{\chi}, \Lambda)}{|\mathbf{X}_S||\mathbf{X}_{\bar{S}}|},$$

and $\nabla_\phi l(\chi, \hat{\chi}, \Lambda)$ is defined as in Equation 6.33.

To derive the update rules for individual parameters y_S, $x_{S,H}$ and z_S, it is sufficient to obtain the derivatives $\frac{\partial \theta(\chi)}{\partial y_S}$, $\frac{\partial \theta(\chi)}{\partial x_{S,H}}$ and $\frac{\partial \theta(\chi)}{\partial z_S}$, and employ these in the GPD update rules. In practice the speaker and channel factors y_S, $x_{H,S}$ and z_S can also be estimated using conventional EM estimate rules.

Note that the estimation of global parameters also requires estimation of specific parameters, since the update rules for the former require the latter. Thus, estimation of global parameters involves estimation of both global and specific parameters for all the speakers in the training set. To learn a model for a new speaker for whom only a small amount of training data have been made available, only the specific parameters need be learned employing the already-known global parameters.

Although we have explained the above in terms of JFA, the same formulation can also be used to learn i-vector representations [14], which are essentially the same as the above without explicit separation of channel factors. AUC-optimized PLDA based representations can be derived similarly to the rules given above, with the modification that the GPD rules

will now employ partial derivatives with respect to PLDA parameters.

Algorithm 6.2. Optimization of the ROC curve, JFA I

Initialization

1. Train the imposter or cohort model (EM) and the target speaker model (MAP)

$$\bar{\Lambda} = \{\bar{W}_k^C, \bar{\mu}_k^C, \bar{\Sigma}_k^C\} \quad \Lambda = \{W_k^C, \mu_k^C, \Sigma_k^C\}. \tag{6.40}$$

2. Compute the misverification distance,

$$d_n(\chi) = -g_n\left(\chi;\Lambda\right) + G_n\left(\chi;\Lambda\right). \tag{6.41}$$

Parameter Estimation, Global Parameters

1. Compute the combined loss function for the FA and FR errors using,

$$\Upsilon(\Lambda) = 1.0 - \frac{1}{|\mathbf{X}_S||\mathbf{X}_{\bar{S}}|}\sum_{S\in\mathcal{S}}\sum_{\chi\in\mathbf{X}_S}\sum_{\chi\in\mathbf{X}_{\bar{S}}} R(\theta_S(\chi),\theta_S(\hat{\chi})). \tag{6.42}$$

2. Obtain the gradients of the *sigmoid loss function* for each of the model parameters: $\{w_k^C, \mu_k^C, \Sigma_k^C\}$ and plug them to:
$\Lambda_{t+1} = \Lambda_t - \epsilon\nabla\Upsilon(\chi;\Lambda)$,

$$\nabla_\phi\Upsilon(\mathbf{X},\Lambda) = -\sum_{S\in\mathcal{S}}\frac{\sum_{\chi\in\mathbf{X}_S}\sum_{\chi\in\mathbf{X}_{\bar{S}}}\gamma R(1-R)\nabla_\phi l(\chi,\hat{\chi},\Lambda)}{|\mathbf{X}_S||\mathbf{X}_{\bar{S}}|}. \tag{6.43}$$

Recall, $\nabla_\phi l(X,\hat{X},\Lambda) = \left(-\frac{\partial g(X;\Lambda)}{\partial\phi_k} + \frac{\partial G(X;\Lambda)}{\partial\phi_k}\right)\mathrm{sgn}(c)$.

3. Optimize the models for every class, considering $Q = VV^\top + \Psi$

   (a) For a specific class C, a component k in the GMM, dimension d. and a set of training X_t where $t = 1...T$.

$$\frac{\partial p(\chi;\Lambda)}{\partial\tilde{v}_{i,j}} = \frac{w_k\,\mathcal{N}(\chi|\mu_k,q_k)}{\sum_{k'}w_{k'}\,\mathcal{N}(\chi|\mu_{k'},q_{k'})} \left\{[Q^{-1}(x-u)]_i[(x-u)^\top Q^{-1}V]_j - [Q^{-1}V]_{ij}\right\}. \tag{6.44}$$

   (b) For Ψ,

$$\frac{\partial p(\chi;\Lambda)}{\tilde{\psi}_{i,j}} = \frac{w_k\,\mathcal{N}(\chi|\mu_k,q_k)}{\sum_{k'}w_{k'}\,\mathcal{N}(\chi|\mu_{k'},q_{k'})}\left\{\frac{1}{2}\left(Q_{ii}^{-1} - [Q^{-1}x-\mu]_i^2\right)\right\}. \tag{6.45}$$

Algorithm 6.2. Optimization of the ROC curve, JFA II

(c) For U,

$$\frac{\partial p(\chi;\Lambda)}{\tilde{U}_k} = \frac{w_k \mathcal{N}(\chi|\mu_k, q_k)}{\sum_{k'} w_{k'} \mathcal{N}(\chi|\mu_{k'}, q_{k'})} \left\{ \left(\frac{\chi - \mu_k}{\tilde{U}_k} \right)^2 - 1 \right\}. \tag{6.46}$$

(d) For w_k and to ensure that $\sum_{k=1}^{K} w_k = 1$, let, $w = \frac{\exp(\tilde{w})}{\sum_{k=1}^{K} \exp(\tilde{w})}$,

$$\frac{\partial p(\chi;\Lambda)}{\partial \tilde{w}_k} = \frac{\mathcal{N}(\chi|\mu_k, \sigma_k)}{\sum_{k'} \tilde{w}_{k'} \mathcal{N}(\chi|\mu_{k'}, \sigma_{k'})}. \tag{6.47}$$

Check for convergence

1. Compute the loss function with the new parameters:

$$\Upsilon_{new}(\Lambda) = 1.0 - \frac{1}{|\mathbf{X}_S||\mathbf{X}_{\bar{S}}|} \sum_{S \in \mathcal{S}} \sum_{\chi \in \mathbf{X}_S} \sum_{\chi \in \mathbf{X}_{\bar{S}}} R(\theta_S(\chi), \theta_S(\hat{\chi})). \tag{6.48}$$

2. Compare Υ_{new} with Υ; if it is higher than a threshold τ iterate from *Parameter Estimation*.

Parameter Estimation, Especific Parameters

For the discriminative approach we have the following:

1. Compute the combined loss function for the FA and FR errors using,

$$\Upsilon(\Lambda_S) = 1.0 - \frac{\sum_{\chi \in \mathbf{X}_S} \sum_{\chi \in \mathbf{X}_{\bar{S}}} R(\theta_S(\chi), \theta_S(\hat{\chi}))}{|\mathbf{X}_S||\mathbf{X}_{\bar{S}}|}. \tag{6.49}$$

2. Obtain the gradients of the *sigmoid loss function* for each of the model parameters: x, y and plug them into $\Lambda_{t+1} = \Lambda_t - \epsilon \nabla \Upsilon(\chi;\Lambda)$,

$$\nabla_\phi \Upsilon(\mathbf{X}, \Lambda) = -\frac{\sum_{\chi \in \mathbf{X}_S} \sum_{\chi \in \mathbf{X}_{\bar{S}}} \gamma R(1-R) \nabla_\phi l(\chi, \hat{\chi}, \Lambda)}{|\mathbf{X}_S||\mathbf{X}_{\bar{S}}|}.$$

   Recall, once again, $\nabla_\phi l(\chi, \hat{\chi}, \Lambda) = \left(-\frac{\partial g(\chi;\Lambda)}{\partial \phi_k} + \frac{\partial G(\chi;\Lambda)}{\partial \phi_k} \right) \text{sgn}(c)$.

3. Optimize for the factors x and y (discriminative form).

Check for convergence

1. Compute the loss function with the new parameters:

$$\Upsilon_{new}(\Lambda_S) = 1.0 - \frac{\sum_{\chi \in \mathbf{X}_S} \sum_{\chi \in \mathbf{X}_{\bar{S}}} R(\theta_S(\chi), \theta_S(\hat{\chi}))}{|\mathbf{X}_S||\mathbf{X}_{\bar{S}}|}. \tag{6.50}$$

2. Compare Υ_{new} with Υ; if it is higher than a threshold τ iterate from *Parameter Estimation*.

6.7 Noise case

Noise increases the inherent variability of the signal, potentially shifting it from any operating point a classifier may be optimized for, possibly in an unpredictable manner. We demonstrate that an operating-point-agnostic training paradigm that optimizes the entire ROC curve can result in classifiers that are significantly more robust to noise than conventional classifiers *even when the effect of noise is not explicitly considered*. The training formalism that optimizes the entire ROC curve by maximizing the AUC, follows the same criteria explained above.

First, we will provide some intuition about the role of the classifier. We can view it in the sense of the distributions and how they are affected by the noise. Moreover, we can also observe the behavior of the ROC curve.

As explained before, the noise can cause unpredictable effects as shown in Figure 6.3. First, we can observe random shifts on the distributions, when noise is added. The changes are more erratic when the SNR increases. The reliability of the classifier is also diminished: the EER increases (the same occurs to the DET curve) and the AUC of the ROC curve decreases.

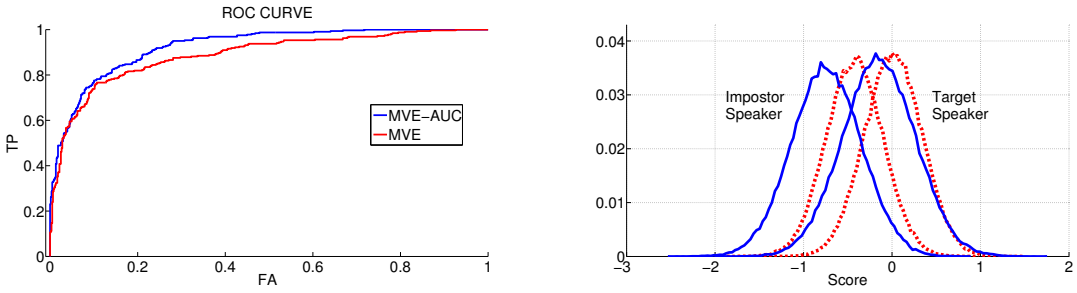


Figure 6.3: The effect of noise in the score distribution and in the ROC curve

To prevent these errors, we used the AUC approach in the usual way. For matched conditions, although the AUC approach does not take into account the noise, the models provide improvement compared to the baseline. AUC compensates for the variations induced by the noise and focus on increasing the AUC. Note that the effects are diminished as shown also in Figure 6.3.

6.8 Comparing the AUC approach with the Calibration approach

In this section, we point out some of the differences between the AUC approach and the Calibration approach described in [133, 76]. Calibration has been successfully used in recent

studies with successful results [134, 133, 135]. The idea behind calibration is to transform the scores into LLRs. Let, $\theta(\chi)$ represent a trial score for a speaker s and phrase χ. Then, $\Theta(\chi) = [\theta_1(\chi)\theta_2(\chi)...\theta_N(\chi)]$. The transformation can be represented as follows,

$$\hat{\Theta}(\chi) = R(\theta(\chi), \Lambda) = a\theta(\chi) + b, \tag{6.51}$$

where R represents a linear transformation, and Λ are the parameters a, b. When optimizing these parameters, the objective logistic regression objective function is minimized [76].

On one hand, the goal of the calibration approach is to optimize the set of Λ parameters. The LLR function employed is,

$$\theta(\chi) = \log e^{\hat{\theta}(\chi)} - \sum_{j=1}^{N} \log e^{\hat{\theta}(\chi)}. \tag{6.52}$$

On the other hand, the discriminative approach we propose maximizes the separation between target and imposter models [19, 136]. It employs positive and negative samples to optimize the GMM model parameters. But the main purpose is either to maximize the *area under the ROC curve*, the DCF or to maximize the separation between the two opponent models. The main LLR function is as follows,

$$\theta\left(\chi\right) = \log\left(p\left(\chi|\Lambda_S\right)\right) - \log\left(p\left(\chi|\Lambda_{\bar{S}}\right)\right),$$

where $\log\left(p\left(\chi|\Lambda_S\right)\right)$ and $\log\left(p\left(\chi|\Lambda_{\bar{S}}\right)\right)$ are the log-likelihoods corresponding to the target model Λ_S and the anti-model $\Lambda_{\bar{S}}$.

In this respect the goals that motivate both techniques are different. However, a study of the C_{LLR} under the discriminative approach can be for future research.

6.9 Experiments and Results

We ran two experiments to evaluate the proposed AUC-minimization approach. In the first experiment, we compared the performance of conventional MAP and JFA-based learning with JFA optimized using the AUC criterion on speech recordings, where the noise conditions in the training and test data were matched. In the second experiment, we compared the performance of AUC-minimization-based MVE against conventional methods on mismatched conditions, where the test data are noisy.

6.9.1 On the selection of the competing samples

In general, the discriminative methodologies use positive and negative examples to optimize the objective function (see Section 5.5 for details on the databases used). However, there are

negative samples (or imposter speakers) that are more suitable to compute more accurate models.

In this section, we study which samples can produce impact in the correct training of the models. The different discriminative approaches described are based on the minimum classification error approach. All of them depend on a misverification measure which is a difference of likelihoods of the form,

$$d\left(\chi_u, \Omega_u, \Lambda_u\right) = -g\left(\chi_u, \Omega_u, \Lambda_u\right) + G\left(\chi_u, \Omega_u, \Lambda_{\hat{u}}\right). \tag{6.53}$$

In this sense, the training samples that can be highly confused as imposters or target speakers represent a higher risk, see Figure 6.4.

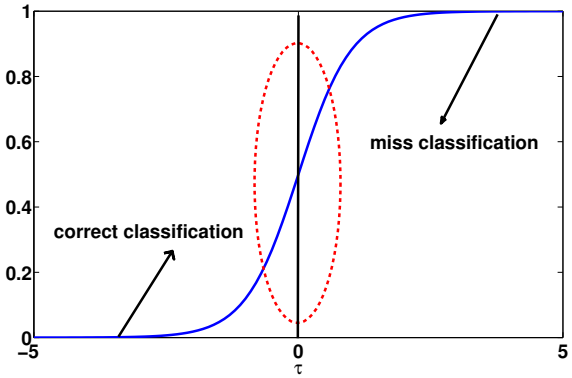


Figure 6.4: Vulnerable area in misverification measure d

In training, the designer has complete control over the system. A filtering of the samples that belong to the correct or incorrect classification; *i.e.* the flat parts of the sigmoid function, can be granted as correct or penalized accordingly. However, the samples that appear in the slope of the sigmoid function are the ones that refine the models. Hence, after a few iterations and noting convergence, it is possible to discard samples which give high and low misverification measures and focus on the central part of the sigmoid. This observation is important for both the current algorithm, where at first glance the basic approach is to perform the experiments with all the data that are available.

Table 6.1, analyzes the performance for a set of 100 users discarding 10%, 20%, 30% and 40% of the total data. By sorting the cohort sets with the lowest d, we found the sets that are more confusing for the classifier.[4] We observe that removing data samples causes the error rate to become worse as expected, but we can choose to remove from 10% to 30% and obtain competitive results.[5]

[4]The experiments are based on the experimental setup in 5.5, but selecting 100 target users and performing full-set optimizations for the cohort and target models.

[5]Special care has to be given when the EER becomes very low (below 1%).

Samples discarded	*EER*
original MAP	10.5
AUC-MVE	9.34
10%	9.36
20%	9.45
30%	9.87
40%	10.7

Table 6.1: On the selection of the number of cohort samples for discriminative training

6.9.2 Experiments on clean speech

In the first approach, all training and test recordings were noise-free, although collected over varied channels. This is the standard setup for the NIST 2008 test. We compared the baseline classification performance to that obtained with the AUC-optimized JFA. For the AUC-optimized JFA, the loading matrices V, U and D were computed to maximize the AUC. The factors y_S and $x_{H,S}$ were learned in a conventional way. For the test speakers, target users were enrolled conventionally, through MAP adaptation and computation of the factors: x_S, $y_{H,S}$ and z.

For the AUC-optimized training, we need: a) target and impostor tokens, b) an initial target model and the imposter. The initial target model was provided by the adapted models in the baseline (MAP).

Table 6.2 shows the results obtained. They are consistent with comparisons performed by other researchers: JFA outperforms both baseline MAP learning as well as MVE learning significantly. JFA uses an optimized implementation by [27]. The results have been optimized empirically and are considered competitive for the particular data setup used. All performance numbers improve as the amount of training data used to learn the base UBM increases.

More important, we note that the AUC-optimized classifier actually outperforms conventionally trained models both for MVE and JFA. In fact, over multiple runs of the experiment with different initializations and parameterizations, the trend of results in Table 6.2 were maintained. Note that for the MVE and AUC approaches we decreased the number of positive and negative examples in the iterations. For the first iteration, we used the complete cohort or UBM set. From the second iteration till the last (approximately 10) we reduced the number of training samples in 30%.

In addition, Figure 6.5 we show the results of the current AUC of the different approaches.

System	*EER*	*minDCF*
Baseline MAP	15.95	6.7
Baseline JFA	12.07	4.2
MVE	13.51	5.8
AUC-MVE	13.21	5.4
AUC-JFA	11.93	3.9

Table 6.2: AUC optimization: EER for MVE and JFA, clean condition

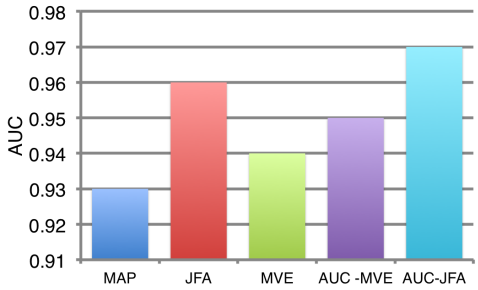


Figure 6.5: AUC optimization: AUC results for different systems, clean conditions

6.9.3 Experiments on noisy speech

In the second experiment, we compared baseline techniques to AUC-minimized training on noisy speech. Experiments were performed using speech corrupted to a variety of SNRs (10 dB, 15 dB, and a cocktail of 0-15 dB), all of them using babble noise.

The first column of Table 6.3 shows the results obtained for clean data. The results are consistent with comparisons performed by other researchers: JFA outperforms both baseline MAP learning as well MVE learning significantly. In both cases, the models learned via AUC-minimization somewhat outperform conventionally trained models. All performance numbers are noted to improve as the amount of training data used to learn the base UBM increases.

The remaining columns of Table 6.3 compares MAP, MVE, JFA, AUC-optimized JFA and AUC-optimized MVE on noisy speech of various SNRs. We observe that AUC-optimized learning consistently outperforms conventional training in all cases. Moreover, the best results are obtained with AUC-optimized MVE. The results are consistent across all noise conditions. This is in contrast to the observation for clean speech, where the best performance is obtained with JFA.

Figure 6.6 shows results for the current AUC. We can observe that for the clean scenario, JFA gets the best results; however, the MVE approaches perform better as SNR decreases.

Lastly, Figure 6.7 shows the relative improvement obtained from the baseline and by applying the AUC approach to both the MVE and the JFA approaches.

System	*clean*	*15 dB*	*10 dB*	*0-15 dB (cocktail)*
MAP	15.95	18.01	17.48	35.7
MVE	13.51	17.67	17.15	28.1
JFA	12.07	17.23	16.79	27.3
JFA-AUC	11.93	16.51	16.22	24.0
MVE-AUC	13.21	15.93	15.78	22.8

Table 6.3: AUC optimization: EER of the noisy task (babble noise).

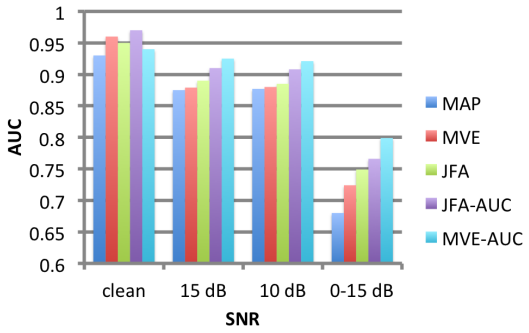


Figure 6.6: AUC optimization: AUC results for different systems in Noise Conditions

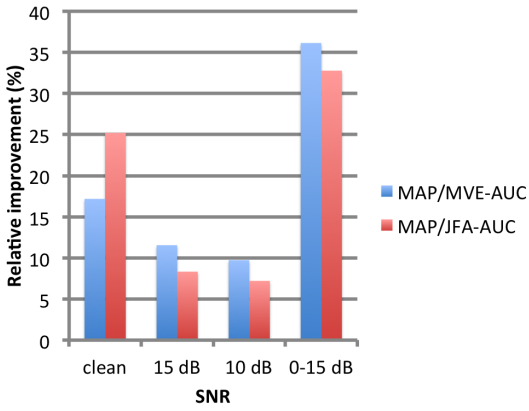


Figure 6.7: AUC optimization: relative improvements.

6.9.4 The effect of the optimization of the AUC of the ROC curve in the score distribution

In this section, we explore the aftermath of performing the area under the ROC curve optimization in the score domain. In here, we show an example of speech at 10 dB SNR (babble noise).

Figure 6.8 shows score decision distributions for this specific case. The dotted line represents baseline scores after applying the MVE approach, which shows a baseline performance. The continuous line represents the optimized AUC of the ROC curve. Note

that the means are shifted, from a -0.1 to 0.03 for the target speakers and from -0.5 to -0.7 for the imposter speaker. The variance is broadened from 0.22 to 0.31 for the target speakers and from 0.63 to 0.75. Finally, the EER goes from 17.67 to 15.93.

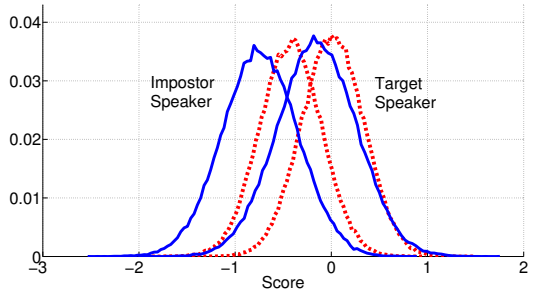


Figure 6.8: AUC Optimization: Score Distributions for 10dB SNR, babble Noise

Figure 6.9 presents the same scores but plotted in terms of a ROC curve. We clearly observe an improvement from the baseline to the curve that includes the AUC optimization. The red curve corresponds to MVE alone and the blue line to the optimized system.

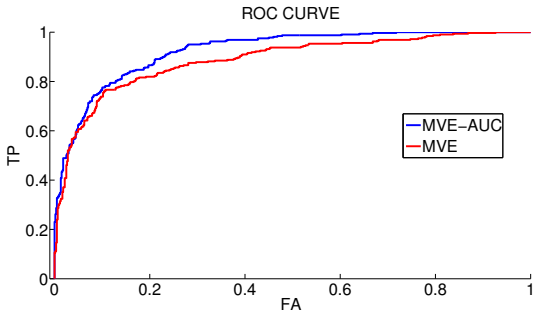


Figure 6.9: AUC optimization: ROC curve for 10dB SNR, babble Noise

Last, we show the effect in a DET curve. We can see how AUC lowers not just the EER but the error rate at every operating point along the curve. Once again, the red curve corresponds to the MVE alone and the blue line to the optimized system.

6.9.5 Number of Gaussians

We also performed experiments when the models contained less than 1024 Gaussian components. Table 6.4 shows the results for 1024, 512, 256, 128 and 64 Gaussian components for different systems. We can observe how results are better for JFA in the clean case. However, they are not as good when the number of components of the GMM are reduced. The reason behind this effect is that FA depends on the correct estimation of the distributions and the amount of the data used in the training. On the other hand, the MVE approach optimizes for the empirical verification error. In the case of few Gaussians, the MVE uses the complete feature vector space and optimizes accordingly; *i.e.,* the model

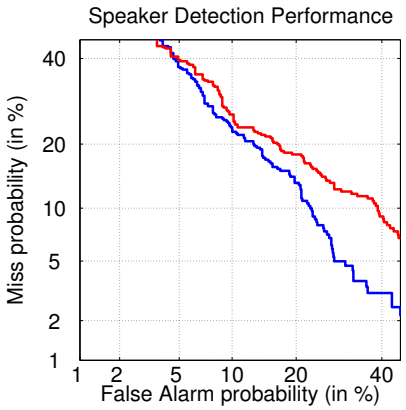


Figure 6.10: AUC optimization: DET curve for 10dB SNR, babble Noise

parameters have space to move by minimizing the objective function. Factor analysis sticks to the correct optimization based on ML. MVE shows stable results for all the cases.

System	*1024*	*512 G*	*256 G*	*128 G*	*64 G*
MAP	12.9	15.9	18.3	20.5	23.9
MVE	10.8	13.5	16.6	17.4	20.3
JFA	8.5	12.0	20.2	23.7	25.2
JFA-AUC	7.2	11.9	17.3	20.2	23.8
MVE-AUC	9.1	13.2	16.1	16.9	18.8

Table 6.4: AUC optimization: EER for different number of GMM components.

6.10 Summary

This chapter analyzes the optimization of the *area under the ROC curve*. We performed ROC maximization by using a discriminative approach, based on the MVE algorithm. The results showed that including AUC optimization improves the results for every approach: MVE and JFA. We also observed the effects of our methodology in the ROC curve, the classification distribution, the DET curve, the EER and the minDCF. Moreover, we explored the number of Gaussians for both approaches. For a small number of Gaussians, the discriminative approach obtains lower EER. However, for more than 256 Gaussians, JFA clearly provides the best improvements.

Chapter 7

The Ensemble Modeling Approach

> The highest education is that which does not merely give us information but makes our life in harmony with all existence.
>
> - Rabindranath Tagore

The previous chapters showed different forms of training. Traditional approaches focus on the optimization of target models and use a single UBM as the basis. In this chapter, we explore the idea of having multiple imposter models. We start from the premise that the imposter data which can clearly be confused with the true speaker can better enhance the discriminative property of the classifier rather than the imposter data that is not very *likely* to belong to the target speaker. If well addressed, the approach can improve the system performance.

We also consider the noise case as a natural branch of this approach. The noise conditions of the speaker verification systems can be addressed in particular ways, either by using ASR techniques or by just using the known algorithms applied to noise. Previous research had also focused on employing just a single background model that can adapt to noisy conditions. In our research, we proposed to fragment the data space and obtain multiple imposter regions for specific noise conditions, but also within them.

7.1 Cohorts

The first idea is concerned with *multiple models* for the imposter or background model is the cohort. A *cohort* is, in essence, an imposter that is very "likely" to be accredited as target speaker. The reasons are diverse, but mainly occur when the classifier fails to discriminate between two speakers whose acoustic information is similar. The cohorts are

mainly employed in the final stage of verification. Usually, they are grouped and form cohort sets that are used for normalizing the scores to remove the effects of channel mismatch and speaker variabilities [70, 137, 138]. The main goal of these cohorts is to transform the scores so that the new scores are in the same range. Thus, a unique decision threshold can be easily established for all the target speakers.

However, the cohort idea can also be applied in the training stage to obtain more specific models. There is just a small number of studies that focus on the cohort selection to form *multiple models*; some examples are [139, 58]. The main interest of these studies is the amount of data needed to model the speakers, the data that are most useful to generate the UBM or the cohort models, and the efficiency of the methods in terms of computing time.

Inspired by this trend, we present an *ensemble approach* which develops several imposter models from the cohort space for every target speaker. Moreover, we propose an *ensemble model* for noise conditions. In this case, one of the noise conditions acts as a target condition and the rest as competing classes for optimization purposes. In the next sections, we present the details of the approach.

7.2 Ensemble Model for Robust Verification

We now describe our proposed variation to the basic likelihood-ratio test based framework. Instead of estimating a general background model from all available data, we *partition* the space of background signals according to a variation-inducing factor we wish to account for. Then, we train a separate *specific* background model for each of the partitions, and corresponding to each of the background models we train a target model for the speaker. When a new recording arrives, we must choose the appropriate partition, and use the corresponding background and target models for the likelihood ratio test. Since the approach effectively employs an *ensemble* of background models, we refer to it as the *ensemble model* approach. Below we outline our procedure for training background models for each of the steps of our procedure.

We begin by assuming that we have a large collection of recordings from which to train the background models. We assume that the signal space is partitioned into P non-overlapping regions $\Omega_1, \Omega_2, \cdots, \Omega_P$, such that together these partitions cover the entire space, see Figure 7.1 (the background space was partitioned into five different regions; from each of this new data sets we obtain a model). Correspondingly, we assume that the recordings are clustered into P groups, each corresponding to signals that fall into one of these partitions. We defer the description of exactly *how* these partitions are obtained until the next section.

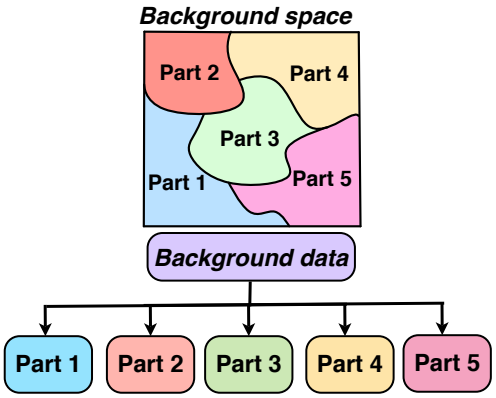


Figure 7.1: Partition ensemble Scheme

7.3 Finding the partitions

The first challenge of this approach is to obtain suitable partitions $\Omega_1, \cdots, \Omega_P$ from the signal space. It is natural to compute those partitions blindly or using some prior information. Although having prior information about the clusters may signify of great help; it is always desirable to investigate when the system has to decide autonomously. In this research we explored both branches.

7.3.1 Supervised Partitioning

Supervised Partitioning may be used, when *prior* knowledge of the factors by which we wish to partition the space is available for the background-model training data. Below we consider two mechanisms: partition by *noise* and partition by *speaker*. Partitions may similarly be obtained by other factors such as channel variations. Hierarchical partitioning strategies that consider multiple factors concurrently may also be used [140].

7.3.1.1 Environment-based partitions

We partition the signal space according to the SNR. In this research, we assume that partitions are formed based on the SNR of the signals. We divide the range of all possible SNR values into P intervals. Each interval represents a partition of the signal space. Let SNR^C_{min} and SNR^C_{max} represent the minimum and maximum SNRs associated with partition Ω_C. A signal X with *signal-to-noise-ratio*, SNR_X, is assigned to a partition C such that

$$SNR^C_{min} < SNR_X \leq SNR^C_{max}. \tag{7.1}$$

Note that partitions may also be formed based on noise type, or other known characteristics of the noise. In this research, however, we have only considered SNR.

The environment partitions may also be based on the channel. For example, we can

distinguish between microphone and telephone, and also develop subcategories of them that depend on a particular feature of the device. For training purposes in several databases (for example, NIST databases [116]) this information is available. Then, each recording associated to a certain channel type belongs to a partition Ω_C.

7.3.1.2 Speaker Partitions

When speaker identity is known for all recordings in the training set, partitions are obtained by clustering them by speaker. We first compute a universal background model (UBM) from unpartitioned data. We then use an agglomerative clustering procedure to cluster speakers. Initially, each speaker forms his or her own cluster. From all the current clusters, we select two clusters with the smallest distance and merge them.[1] The procedure continues until there is no cluster left. Agglomerative clustering iteratively merges the closest clusters until the desired number of clusters (and consequently, partitions) is obtained.

At each stage of the clustering, the UBM is adapted via MAP adaptation to learn a model Λ_C for each new cluster C. In this research, the distance between any two clusters is defined by the empirical cross entropy:

$$d(C_1, C_2) = \frac{1}{|\chi_{C_1}|} \log \frac{P(\chi_{C_1}; \Lambda_{C_1})}{P(\chi_{C_1}; \Lambda_{C_2})} + \frac{1}{|\chi_{C_2}|} \log \frac{P(\chi_{C_2}; \Lambda_{C_2})}{P(\chi_{C_2}; \Lambda_{C_1})}, \tag{7.2}$$

where χ_{C_i} is the set of all recordings in cluster C_i. Other clustering mechanisms may also be employed.

7.3.2 Unsupervised Partitions

When *a priori* knowledge about the training recordings is unavailable, partitions may be formed by clustering them using unsupervised methods.

For the purpose of this research, we employed k-means as a clustering tool. The algorithm starts first deciding an appropriate number of clusters, K. In our case, we can relate the clusters according to a specific factor. For example, if we are dealing with noisy signals at different SNR (from 0 to 20) we can decide $K = 5$, so that it accounts for five intervals. Later, the system can provide more granularity to these clusters.

Once the number of clusters is decided, we initialize K initial points, called *centroids* either by random values or by picking randomly K vectors from the pool of imposter data. Afterwards, the system assigns "memberships" to each data vector (distances from the vector to each centroid). The lowest distance, means that the vector belongs to a certain cluster. Once we account for all the data vectors, the system goes through the same process

[1] The smallest distance may be computed using criteria as log likelihood, cross-entropy, euclidean distance between the means, among others.

iteratively until the error function E does not change significantly,

$$E = \sum_{j=1}^{K} \sum_{\chi_l \in c_j} (\chi - c_j)^2 , \tag{7.3}$$

where χ is the set of all recordings in the data space, and c_j is a specific centroid.

The factor by which partitions are formed can be controlled by using an appropriate distance function. Generic clustering based on Euclidean distances or likelihoods may be used to cluster the data by a dominant factor.

7.4 Training the Ensemble Model

Corresponding to each of the partitions $\Omega_1, \cdots, \Omega_P$ we train a separate partition-specific background model. All background models are GMMs. In principle these can be trained separately for each partition using the EM algorithm. However, we require each of the background models to be highly *specific* to the partition they represent, and not generalize to other partitions. In order to do so, we train all of them together using the following *discriminative* training procedure [140].

Let Λ_C represent the model for a partition Ω_C. Let χ_C represent all (training) recordings assigned to Ω_C. For any partition Ω_C, let

$$\Omega_{\bar{C}} = \bigcup_{\bar{C}' \neq C} \Omega_{\bar{C}'} \tag{7.4}$$

represent the *complement* of Ω_C, *i.e.* the union of all partitions that are not Ω_C and i is the number of the partition.

Let $g(\chi; \Lambda_C) = \log P(\chi; \Lambda_C)$ represent the log-likelihood of any recording χ computed with the distribution for partition Ω_C. We can now define $d(\chi, \Lambda_C)$, a misclassification measure for how likely it is that a data $\chi \in \chi_C$ from Ω_C will be misclassified as belonging to $\Omega_{\bar{C}}$ as

$$d(\chi, \Lambda_C) = -g(\chi; \Lambda_C) + G(\chi, \Lambda_{\bar{C}}) , \tag{7.5}$$

$G(\chi, \Lambda_{\bar{C}})$ represents the combined score obtained from a partitions in $\Omega_{\bar{C}}$,

$$G(\chi, \Lambda_{\bar{C}}) = \log \left\{ \frac{1}{|\Omega_{\bar{C}}|} \sum_{C' : \Omega_{C'} \in \Omega_{\bar{C}}} \exp\left[\eta g(\chi, \Lambda_{C'})\right] \right\}^{\frac{1}{\eta}} , \tag{7.6}$$

where $|\Omega_{\bar{C}}|$ is the number of partitions included in $\Omega_{\bar{C}}$, and η is a positive parameter. Now, we can define a new objective function for discriminative training of Λ_C. This function takes the following form,

$$\ell(\Lambda_C) = \frac{1}{|\chi_C|} \sum_{X \in \chi_C} \frac{1}{1 + \exp\left[-\gamma\left(d\left(\chi, \Lambda_C\right) + \theta\right)\right]}, \tag{7.7}$$

where $|\chi_C|$ represents the number of recordings in χ_C, and γ and θ are control parameters. Note that for this specific case, we consider the formulation with different competing classes (as many as the number of partitions). Hence, the optimization is performed per partition. Finally, the objective function in Equation 7.7 can be optimized by applying the following generalized probabilistic descent (GPD) update rule for Λ_C:

$$\Lambda_C^{t+1} = \Lambda_C^t - \epsilon \nabla \ell(\Lambda_C)|_{\Lambda_C^t}. \tag{7.8}$$

Since all background models are GMMs, $\Lambda_C = \{w_k^C, \mu_k^C, \Sigma_k^C\}$, where w_k^C, μ_k^C and Σ_k^C are the mixture weight, mean and covariance matrix of the k-th Gaussian of the GMM for Λ_C. To obtain the update rules for these individual parameters, $\frac{\partial \ell(\Lambda_C)}{\partial w_k^C}$, $\frac{\partial \ell(\Lambda_C)}{\partial \mu_k^C}$ and $\frac{\partial \ell(\Lambda_C)}{\partial \Sigma_k^C}$ must respectively be plugged in for $\nabla \ell(\Lambda_C)$ in the update rule of Equation 7.8.

Once again, as in previous chapters we solve for the parameters. Then, we consider the following:

$$\mu = \sigma\, \tilde{\mu}, \qquad \tilde{\sigma} = \log(\sigma), \qquad w = \frac{\exp(\tilde{w})}{\sum_{k=1}^{K} \exp(\tilde{w})}.$$

For μ, a specific k, dimension d and to avoid bias, let

$$\nabla_\phi \ell(\chi, \tilde{\mu}_k) = \frac{1}{|\chi_C|} \sum_{X \in \chi_C} \gamma \ell (1 - \ell) \frac{w_k\, \mathcal{N}(\chi | \mu_k, \sigma_k)}{\sum_{k'} w_{k'}\, \mathcal{N}(\chi | \mu_{k'}, \sigma_{k'})} \left(\frac{\chi}{\sigma_k} - \tilde{\mu}_k \right). \tag{7.9}$$

For σ_k, let

$$\nabla_\phi \ell(\chi, \tilde{\sigma}_k) = \frac{1}{|\chi_C|} \sum_{X \in \chi_C} \gamma \ell (1 - \ell) \frac{w_k\, \mathcal{N}(\chi | \mu_k, \sigma_k)}{\sum_{k'} w_{k'}\, \mathcal{N}(\chi | \mu_{k'}, \sigma_{k'})} \left\{ \left(\frac{\chi - \mu_k}{\tilde{\sigma}_k} \right)^2 - 1 \right\}. \tag{7.10}$$

Finally, for the set of weights w_k,

$$\nabla_\phi \ell(\chi, \tilde{w}_k) = \frac{1}{|\chi_C|} \sum_{X \in \chi_C} \gamma \ell (1 - \ell) \frac{\mathcal{N}(\chi | \mu_k, \sigma_k)}{\sum_{k'} w_{k'}\, \mathcal{N}(\chi | \mu_{k'}, \sigma_{k'})}. \tag{7.11}$$

The procedure is employed by the MVE and JFA approaches. We obtained refined target and imposter models.

7.4.1 MVE

Once the background models Λ_C are obtained for all partitions, we can also train *partition-specific* target-speaker models, Λ_S^C by fixing the background model Λ_C and using a similar discriminative approach to train Λ_S^C. Note that we consider the usual MVE approach (the binary case), with just two classes (target and imposter), and $\eta = 1$. Hence, Equation 3.31, which is the multiclass version, can be reduced to $G(\chi; \Lambda_S^C) = \log g\left(\chi; \Lambda_{\bar{S}}^C\right)$.[2]

7.4.2 Factor Analysis

In the same way, the approach can be embedded in the JFA algorithm. The basic UBM is now a set of different imposter models that are treated separately. The system trains the loading matrices V, U and D for each partition. Moreover, it also computes the speaker factors for each Ω_C. In the test stage, the scores for a trial are obtained against each partition.[3]

7.5 Scoring for classification

Once we obtained appropriate models for the target speaker and their corresponding imposter models, the aim is to find suitable ways to score.

Given the pairs,

$$\{\Lambda_{C_1}, \Lambda_S^{C_1}\}, \{\Lambda_{C_2}, \Lambda_S^{C_2}\}, \cdots \{\Lambda_{C_P}, \Lambda_S^{C_P}\},$$

the set of background models for all P partitions and their corresponding partition-specific target speaker models for any claimed speaker S, we can compute the score $\theta^S(X)$ to be employed in the likelihood ratio test for any recording X in one of several ways. Let

$$\theta_C^S(X) = \log P(X|\Lambda_S^C) - \log P(X|\Lambda_C) \tag{7.12}$$

be the likelihood ratio computed in the log domain from the models for partition Ω_C. The options for obtaining the final score $\theta^S(X)$ are:

A) *Partition Selection*: We assign the recording to the most likely partition as

$$\hat{C}(\chi) = \arg\max_C \log P(\chi|\Lambda_C). \tag{7.13}$$

We then compute the score from the assigned partition: $\theta^S(X) = \theta_{\hat{C}(\chi)}^S$. This is a conservative score that selects the partition with signals most likely to be confused with

[2] Refer to Chapter 3.3 for details.

[3] refer to Chapter 3.5.1 for details.

the target speaker (see Figure 7.6). In this sense, the decision threshold must be trained with an accurate design that can guarantee a correct classification. We used Focal (described in [135, 129]) to calibrate the scores.

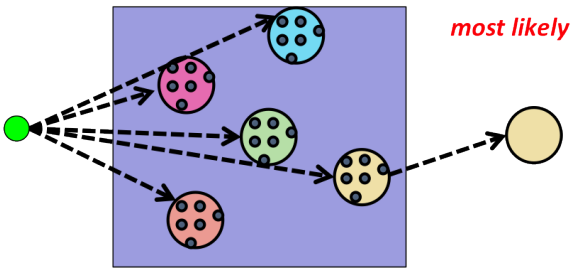


Figure 7.2: Partition Ensemble: Selection

B) *A priori*: If the correct partition Ω_C for X is known *a priori*, then we can simply set $\theta^S(X) = \theta^S_C(X)$, see Figure 7.3.

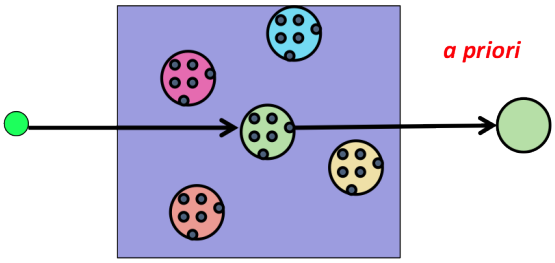


Figure 7.3: Partition Ensemble: A priori

Although we expect to obtain the best results with this approach, the labels are not always available. So we can consider the *A priori* selection as an upper bound.

C) *Best score*: We select the greatest score: $\theta^S(X) = \max_C \theta^S_C(X)$, see Figure 7.4.

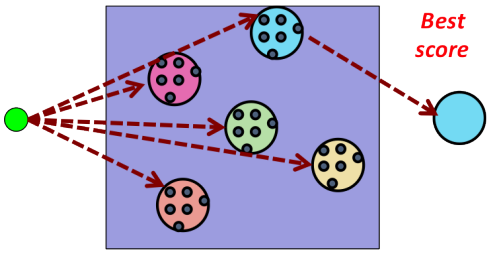


Figure 7.4: Partition Ensemble: Best score

D) *Combination*: Here we simply combine the scores from the different partitions (see Figure 7.5),

$$\theta^S(X) = \sum_C w^S_C \theta^S_C(X). \tag{7.14}$$

In our work we trained an SVM (for details on this algorithm refer to Appendix A.2) to classify the speaker; in this case, w_C^S are simply the weights assigned by the SVM.

In a first approach, we just concatenated scores from each system belonging to a target speaker, $\xi_i = [\theta_1^S, \theta_2^S, ..., \theta_P^S]$. Then, the labels and ξ are the input data for the SVM. A second approach used the normalized log likelihoods of each trial with respect to the target model within a partition, then $\xi_i = [g_1^S, g_2^S, ..., g_P^S]$. For both, the linear kernel was employed. This is equivalent to learn the weights discriminatively. It is also comparable to the fusion approach.

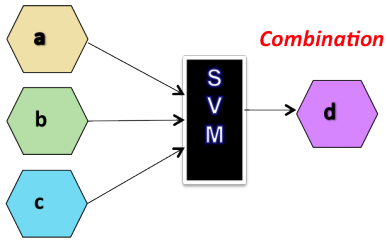


Figure 7.5: Partition Ensemble: Combination

E) *Fusion*: We obtain the scores for the target speaker with respect to the P imposter models. Then, for a trial j, target a, and N different scores, the linear fusion is given by,

$$f_j = \alpha_0 + \alpha_1 a_{1,j} + \alpha_2 a_{2,j} + ... + \alpha_N a_{N,j}. \tag{7.15}$$

To obtain the α weights, we employ the logistic regression fusion as stated in [135, 129] (see Figure 7.6). Target speaker trial scores and imposter trial scores form two matrices: A is a score matrix of $N \times K$, where N are the different classifiers (P partitions) and K are the number of target trials, and B is a score matrix of $N \times L$, where L are the number of imposter trials. Then, the objective logistic regression function based on the *cost* is

$$C = \frac{P}{K} \sum_{j=1}^{K} \log(1 + e^{-f_j - \text{logit}\, P}) + \frac{1-P}{L} \sum_{j=1}^{L} \log(1 + e^{g_j + \text{logit}\, P}), \tag{7.16}$$

where P is a prior probability usually set to 0.5. The target and imposter scores are given by,

$$f_j = \alpha_0 + \sum_{i=1}^{N} \alpha_i a_{i,j} \quad g_j = \alpha_0 + \sum_{i=1}^{N} \alpha_i b_{i,j} \tag{7.17}$$

The complete algorithm is described in Algorithm 7.1.

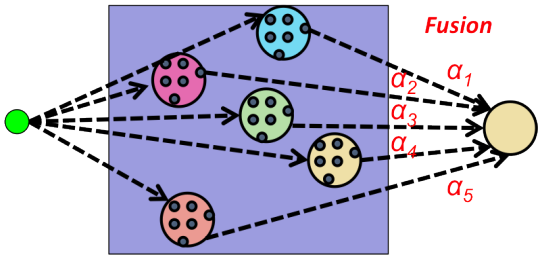


Figure 7.6: Partition Ensemble: Fusion

Algorithm 7.1. Ensemble Approach I

Initialization

1. Train the imposter model (UBM) in the usual way employing EM,

$$\bar{\Lambda} = \{\bar{W}_k^C, \bar{\mu}_k^C, \bar{\Sigma}_k^C\}. \tag{7.18}$$

Finding the partitions

Obtain the partitions Ω_P and their corresponding raw models.

1. Supervised: Assign the elements of each cluster according to some specific attribute known in advance (SNRs, similar speakers, channel). Adapt the UBM to the partition.
2. Unsupervised: Use an unsupervised clustering algorithm (like *k-means*) to find the clusters. Adapt the UBM to the partition.

Training the Ensemble Model

1. Compute the misverification distance for every partition with respect to the competing partitions and the loss function using,

$$d\left(\chi, \Lambda_C\right) = -g\left(\chi; \Lambda_C\right) + G\left(\chi, \Lambda_{\bar{C}}\right), \tag{7.19}$$

where the combined score obtained from a partitions in $\Omega_{\bar{C}}$ is,

$$G\left(\chi, \Lambda_{\bar{C}}\right) = \log\left\{\frac{1}{|\Omega_{\bar{C}}|}\sum_{C':\Omega_{C'}\in\Omega_{\bar{C}}}\exp\left[\eta g\left(\chi, \Lambda_{C'}\right)\right]\right\}^{\frac{1}{\eta}}. \tag{7.20}$$

and the loss function,

$$\ell(\Lambda_C) = \frac{1}{|\chi_C|}\sum_{X\in\chi_C}\frac{1}{1+\exp\left[-\gamma\left(d\left(\chi, \Lambda_C\right)+\theta\right)\right]}. \tag{7.21}$$

2. Obtain the gradients for the model parameters: $\{w_k^C, \mu_k^C, \Sigma_k^C\}$ and plugged them into:

$$\Lambda_{t+1} = \Lambda_t - \epsilon\nabla\ell(\chi; \Lambda), \tag{7.22}$$

$$\nabla\ell(\chi; \Lambda) = \gamma\ell(d)(1-\ell(d))\frac{\partial\, p(\chi; \Lambda)}{\partial\Lambda_k}. \tag{7.23}$$

Algorithm 7.1. Ensemble Approach, Part II

Convergence

1. Compute the loss function with the new parameters:

$$\ell_{new}(\chi;\Lambda) = \sum_n \ell_n(d(\chi;\Lambda))1(\chi \in C_n). \tag{7.24}$$

Compare ℓ_{new} with ℓ if it is higher than a threshold τ iterate from *Parameter Estimation*. Use the new P refined models as the UBM for either MVE or JFA approach.

Scoring for classification

Given the pairs,

$$\{\Lambda_{C_1}, \Lambda_S^{C_1}\}, \{\Lambda_{C_2}, \Lambda_S^{C_2}\}, \cdots \{\Lambda_{C_P}, \Lambda_S^{C_P}\}, \tag{7.25}$$

1. Compute the score for each partition,

$$\theta_C^S(X) = \log P(X|\Lambda_S^C) - \log P(X|\Lambda_C). \tag{7.26}$$

2. Choose among the following scoring options.

   a) *Partition Selection*: Assign the recording to the most likely partition as, $\hat{C}(\chi) = \arg\max_C \log P(\chi|\Lambda_C)$.

   b) *A priori*: If the correct partition Ω_C for X is known *a priori*, then we can simply set $\theta^S(X) = \theta_C^S(X)$.

   c) *Best-score*: We select the greatest score, $\theta^S(X) = \max_C \theta_C^S(X)$.

   d) *Combination*: Combine the scores from the different partitions: $\theta^S(X) = \sum_C w_C^S \theta_C^S(X)$. In this case w_C^S are simply the weights assigned by the SVM.

   e) *Fusion*: Combine the scores from the partitions in a logistic regression fusion, $f_j = \alpha_0 + \sum_{i=1}^N \alpha_i a_{i,j}$.

7.6 Ensemble example: real data

We present an example of the *ensemble* approach using clean data. The intention is to provide the reader with some insight of the procedure before considering more complex experiments. In a first stage we train the target models and the UBM as usual. Second, we partition the data space to obtain specific models. Next, the system trains the models in a discriminative manner such that the empirical error is minimized and the model specificity is enhanced. Finally, several iterations are performed to reach convergence.

The first step after computing the UBM and the target models is to obtain a set of partitions Ω_P from the data space. We employed the unsupervised partition method to show the performance in the worst case scenario where no information about the clusters is available. Figure 7.7 shows the real partition using k-means clustering for the data space, considering a clean dataset and an extraction of the real UBM. For each of those partitions the system builds a model, in this case an adaptation of the current UBM.[4]

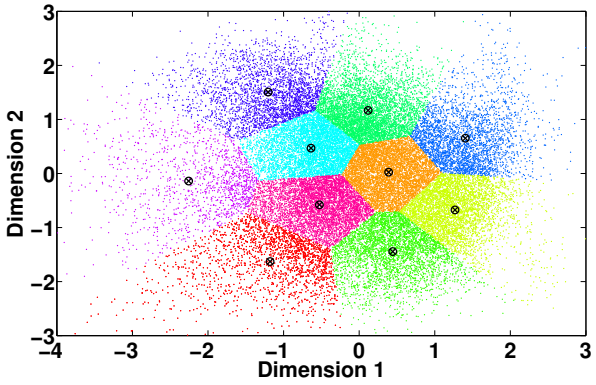


Figure 7.7: Partition Ensemble Scheme, real example

The discriminative *ensemble* approach is then applied to each of the partitions. Figure 7.8 shows the misverification measure $d(\chi, \Lambda_C)$ for some of the partitions (initial and final after 10 iterations) and a single target user. This graph clearly depicts an empirical error reduction. Note that the model parameters change along the iterations.

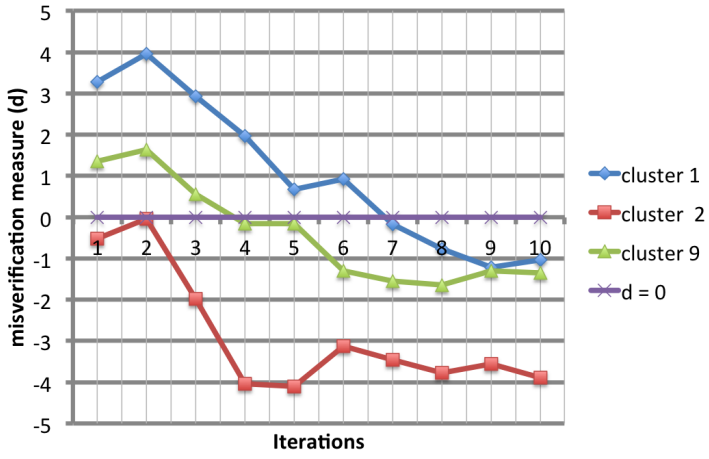


Figure 7.8: Value of d for 3 different clusters and different iterations.

The discriminant function (log-likelihood) histogram from a target and its corresponding cohort histogram are shown in Figure 7.9.[5] We observe that the histograms spread and

[4]The partition models might be an adaptation from the current UBM or an independent computation of a model. In our experience, adaptation works better for all cases.

[5]For this specific example we used all the cohort samples available from the imposter data set.

separate along the iterations, altering the misverification distance d.

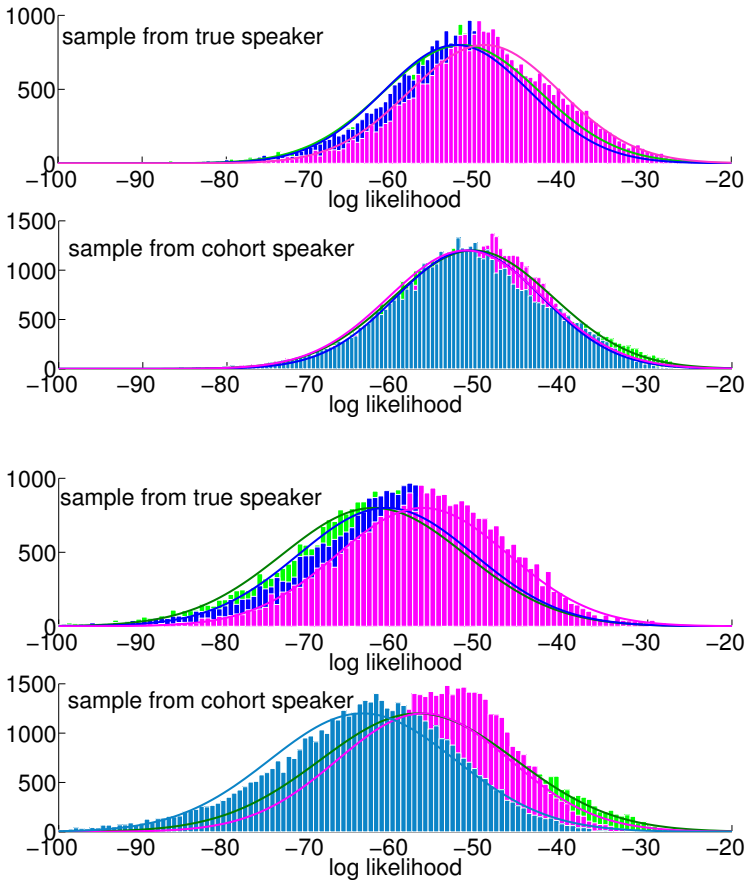


Figure 7.9: Discriminant function (log-likelihood) histograms for one target and 2 cohort samples

After reaching convergence for all the cohort models, the system performs the selected scoring method for every partition. The score results for the ten partitions and the specific target user model are shown in Table 7.1. Note that some of the imposter scores seem to be very close to the actual target speaker; for example, cluster ones and four. Recall that all of these clusters represent negative samples for the target speaker. The actual best score for a set of *positive samples* for this speaker is of 0.91216, which is clearly higher than the imposter scores presented in Table 7.1. Moreover, if the training data of the target speaker were placed in some cluster, most of the samples would be found in Cluster Five (approximately 60%). The imposter samples were randomly selected from the pool of imposter set, but the membership test cluster is number ten (approximately 35%). The clusters that provide more information about imposters that may be confused with the target are 1 and 4, according to the likelihood results. Note that these scores were obtained using unseen data, that is the reason why the actual results from *a priori* , *best score*, and *partition selection* are diferent. Several options are possible for the selection of the

most appropriate score to consider. Table 7.2 describes the results for the scoring method. According to this results, the selected scoring method is the *combination* of scores and the *fusion*.

cluster	*score imposter samples*	*score target samples*
1	0.439598	0.53196
2	-0.540602	0.88931
3	-0.600463	0.91316
4	0.364815	0.44016
5	-0.443482	0.87047
6	-0.200627	0.84925
7	-0.315752	0.85311
8	-0.228535	0.83860
9	-0.326056	0.86034
10	-0.510749	0.87213

Table 7.1: Scores for 10 clusters with respect to a target speaker model.

Scoring method	*Score imposter samples*	*Score target samples*
partition selection	-0.443482	0.88931
a priori	-0.510749	0.87047
best score	-0.600463	0.91316
combination	-0.538124	0.92154
fusion	-0.672565	0.93768

Table 7.2: Scoring selection scores with respect to a target speaker model.

This procedure is extended to all the target speakers; the results are shown in the following sections.

7.7 Experiments and Results

This section includes an experimental analysis for the *ensemble* approach. Algorithms such as MVE and JFA are used together with the *ensemble* to demonstrate its capability. We show results for both clean and noisy condition, and in terms of how the partitions are performed for each case. Moreover, we present a comparison of techniques and how they are affected by our algorithm.

7.7.1 Experimental Setup

We employed the NIST Speaker Evaluation 2004, 2005, 2010 and 2008 database [39] to complete this study. We conducted two experiments, one on clean data and the second on noise-corrupted data. For all experiments, we used the NIST2008 database as targets. For

the noisy condition experiments, babble noise extracted from the Aurora 2 database was added to the training and test files at different SNRS: 0,5,10,15 and 20 dB. The training data was randomly partitioned into equal parts, one for each noise condition; the same procedure was applied for the test set.[6]

The objective in the clean experiment was to partition the space by speaker. For the noisy data, we evaluated partitioning by noise condition. In both cases, we evaluated partitions formed from *a priori* information about the data, as well as by unsupervised k-means clustering. The five methods: *partition selection*, (PS), *a priori*, (AP), *best score*, (BS), *combination*, (CO), and *fusion*,(FU), were evaluated. We used a 256-Gaussian GMMs for background models in all cases.

In the clean experiment, we evaluated the performance obtained with speaker-based partitions with different numbers of partitions. As a baseline we also evaluated the conventional verification framework, where we trained generic gender-dependent background models (UBM) and adapted these to the target speaker using MAP [65], Joint Factor Analysis (JFA) [15, 60] and Minimum Verification Error (MVE) [83] training. The UBM did not include data from any target speaker, not even for the data partitions and their negative/positve samples.

For the noise experiment we employed 5 partitions, one corresponding to each SNR level. The proposed method only establishes a mechanism for defining background models for each partition. The method described in Section 7.4 actually uses MVE to learn partition-specific target models. Partition-specific target models were also trained via JFA or MAP. For the noise experiments these were also evaluated in addition to UBM-based baselines.

7.7.2 Results

Table 7.3 shows the results for the clean experiments with different numbers of partitions. The EER and minDFC results are shown. The ensemble model improves performance in every case. Morever, partitions based on *a priori* knowledge consistently outperform partitions obtained from unsupervised clustering. The best result is obtained when scores from the partitions are fused; this result significantly outperforms JFA, and is the best result we have ever obtained on this test set. Increasing the number of partitions beyond 16 resulted in degradation of performance probably due to the lack of data. Discriminative training of background models to make them specific also turns out to be the key. The alternative is to simply train each of them using maximum likelihood. This was consistently worse than discriminative refinement of partition-specific background models in all experiments.

Figure 7.10 presents the relative improvements for the best results (*combination* and

[6] The noise was added using FaNT software [125].

Condition	*EER*										*minDCF*				
Baseline MAP	16.9										6.9				
Baseline JFA	15.2										6.3				
Baseline MVE	16.3										6.5				
	4-clusters					8-clusters					16-clusters				
Condition	PS	AP	BS	CO	FU	PS	AP	BS	CO	FU	PS	AP	BS	CO	FU
k-means	16.2	15.5	17.1	14.9	13.8	15.2	13.8	16.0	13.2	12.7	14.8	13.2	15.4	12.5	12.0
minDCF	6.45	6.35	6.65	6.23	5.74	6.28	5.85	6.37	5.74	5.64	6.10	5.32	6.36	5.71	5.6
prior info	14.9	14.4	15.4	13.7	12.8	13.5	11.9	13.9	11.3	10.8	12.1	10.9	12.8	9.5	9.1
minDCF	6.24	5.93	6.34	5.67	5.56	5.77	5.58	5.92	5.21	4.93	5.61	5.08	5.62	4.83	4.13

Table 7.3: Ensemble modeling: EER and minDCF for different clusters on clean condition

fusion scoring methods). Although the highest relative improvement is obtained by the systems that have prior information about the attribute; the results for blind clustering provided noticeable improvements, as well.

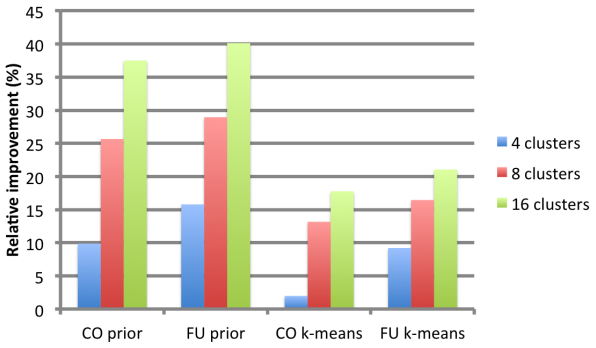


Figure 7.10: Ensemble modeling: relative improvements (clean condition)

Table 7.4 and 7.5 show results for the noisy data. Once again, the *ensemble* method results in improvements in every case. Supervised partitions obtained from *prior* knowledge of SNR result in the best performance. Interestingly, identifying the partition for a test utterance through *a priori* (AP) of its SNR did not result in the best performance, although it does outperform other methods of selecting partitions. The best results are obtained, once again, by combining scores from the partitions either by *fusion* approach or by the *combination* approach. Moreover the best results are obtained by MVE, rather than JFA. This is an inversion with respect to the usual results with generic models, where JFA consistently outperforms MVE training.

Figure 7.11 depicts the relative improvements of the proposed methods with respect to the JFA baseline. The improvements are more noticeable for the MAP and the MVE than in JFA.

	5c-MAP					5c-JFA				
Baseline	28.6					25.0				
minDCF	15.60					13.23				
Condition	PS	AP	BS	CO	FU	PS	AP	BS	CO	FU
k-means	28.3	25.8	29.5	24.2	21.2	26.0	25.1	26.8	24.5	21.5
minDCF	16.89	13.48	17.43	10.57	8.19	13.70	13.22	14.17	10.66	7.86
prior info	25.2	23.4	26.4	22.3	19.2	24.8	24.2	25.2	23.7	20.6
minDCF	13.34	8.97	13.68	8.45	7.64	9.35	9.16	13.28	9.32	7.85

Table 7.4: Ensemble modeling: EER and minDCF for different clusters (noise condition), MAP and JFA

	5c-MVE				
Baseline	28.3				
minDCF	15.32				
Condition	PS	AP	BS	CO	FU
k-means	27.0	24.1	27.7	23.1	20.13
minDCF	14.21	10.48	14.34	8.52	7.86
prior info	23.7	22.2	24.4	20.9	18.4
minDCF	9.21	8.35	10.59	7.78	7.16

Table 7.5: Ensemble modeling: EER and minDCF for different clusters (noise condition), MVE

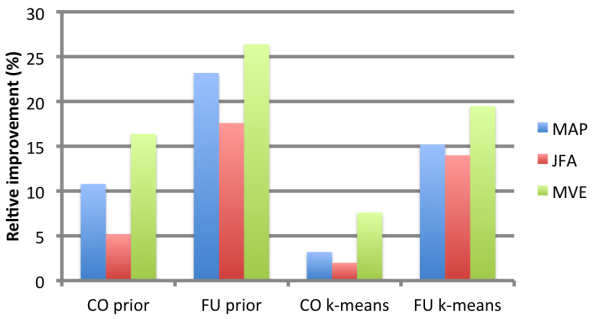


Figure 7.11: Ensemble modeling: relative improvements (noise condition)

7.8 Summary

This chapter shows the theory behind the Ensemble approach. It analyzes the different ways to perform a partition of the data space to produce suitable imposter models that can help to enhance the model specificity and reduce error. The key idea is to simplify the training modeling by discriminatively train separate models that resemble a given attribute. We give details of the relevant steps that occur in the process employing real data to have the insight of the methodology. The results show the potential of the technique specially when embedding the methodology in other discriminative approaches such as MVE. When using the technique with JFA the improvements are substantial.

Chapter 8

Integration of the AUC optimization and Ensemble Methods

> Good that you ask – you should always ask, always have doubts.
>
> - Hermann Hesse *Demian*

In this chapter, we show how to integrate the algorithms described in the two previous sections: *area under the ROC curve optimization* and the *ensemble modeling* approach. First, we present the benefits of optimizing the parameters using the AUC approach. By optimizing for every operation point along the curve, the AUC of ROC curve is maximized, and the EER and the minCDF are reduced. Moreover, we detailed the procedure to partition the data space using the *ensemble* approach. The intention of this chapter is to combine these techniques for both cases: clean and noisy speech.

A general scheme of the *ensemble-AUC* methodology is shown in Figure 8.1. We introduce a top down architecture that comprises several partition levels. Each level can be optimized by the known techniques. Let us suppose that the data space can be partitioned in distinctive clusters for any given condition, not necessarily disjunct (first level). Every cluster contains speaker or other attribute clusters from which we can compute models. The approach is to first choose a condition and build the clusters accordingly. Then, we maximize the distance between these competing regions using *ensemble modeling*. Once we have these new mapped and refined clusters, we continue to the next partition level in the same manner. As an example, the first-level partitions may correspond to any condition (channel, noise type, SNR), and the second to the actual speakers. Note that the refinement of the models, provided by the AUC of the ROC curve, can be included at every level on the methodology to ensure that we are getting optimized models.

In the next sections, we highlight the main characteristics of the *ensemble modeling* and *AUC of the ROC curve optimization* to give the reader a self-contained chapter. Afterwards, we show how both techniques merge, what are the key points to address and the improvements obtained.

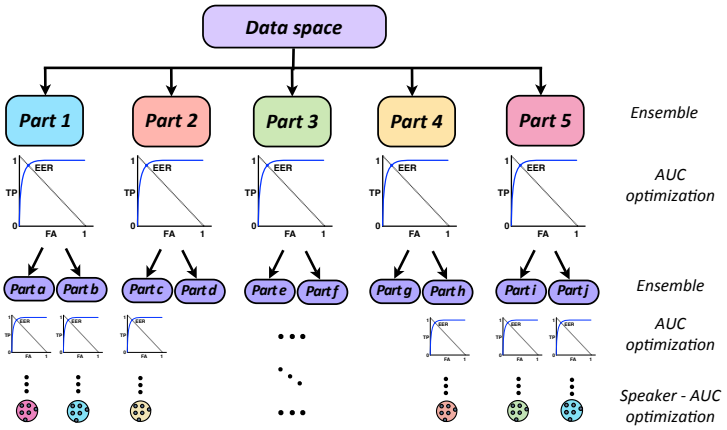


Figure 8.1: Ensemble-AUC modeling scheme

8.1 Ensemble Modeling

In previous chapters, we explored the idea behind partitioning the data space depending on a specific attribute (channel, noise or group of speakers). The data are partitioned forming an *ensemble* of clusters. This can be done in a supervised or unsupervised way. Supervised means that we have some previous knowledge of the nature of the clusters that we will obtain; for example, the type of noise, the kind of channel for each utterance, or the cluster to which each speaker belongs to. Unsupervised means that the clustering is completely blind, but with the expectation that we can partition the clusters according to a specific attribute. Once we obtain those partitions, we compute GMM models for each partition using the classic algorithms (EM for the UBM and MAP for the target speaker or for the competing models). We update the current parameters for each target model and its corresponding competing model using a discriminative approach, maximizing the misclassification distance among them. The new updated models are then the input for a training algorithm such as MVE and JFA. Each partitioned model by itself is used to train both, the target and competing models. When a new utterance is verified by the system, it will produce scores according to a certain target model and its corresponding competing models. From those scores, the final score is chosen according to a selection algorithm.

This procedure has been useful when the phrases are under noise conditions. In those cases, we can have a better definition of the attribute according to the specificity of the condition model that can be a SNR or the noise type.

8.2 Optimization AUC of the ROC curve

The AUC of the ROC curve has shown improvements when optimizing the model parameters. In essence, the methodology searches for the maximization of every point in the ROC curve, and hence the reduction of the EER and the minCDF. Although this approach can occur at any stage in the *ensemble modeling*, we discuss the simplest case: binary classification of a speaker. The first step is to obtain a set of trained models for both target and imposters. Afterwards, we use the WMW as a statistic to compute the *area under the curve* of the ROC curve of a classifier. Moreover, we modify the statistic to be a function of the current *false acceptance* FA and *miss probability*,

$$\Upsilon(\Lambda) = 1.0 - \frac{\sum_{\chi \in \mathcal{H}} \sum_{\hat{\chi} \in \mathcal{W}} 1(\theta(\chi) > \theta(\hat{\chi}))}{|\mathcal{H}||\mathcal{W}|}, \tag{8.1}$$

where $\mathcal{H}$ and $\mathcal{W}$ are two sets of data belonging respectively to class C and competing class $\bar{C}$ (*i.e.* not in C). The AUC of the ROC curve of a classifier computes a score $\theta(\chi)$ to determine if any data instance χ belongs to C. The modified statistic is, then, the new objective function to optimize. The model parameters that describe the scores (log-likelihod ratio) are given by GMMs; *i.e.*, the parameters to optimize are $\Lambda = \{W_k^C, \mu_k^C, \Sigma_k^C\}$. Once again, the optimization follows a discriminative approach based on MVE. The approach is also extended to noise conditions with promising results.

In the next sections, we can see how is it possible to combine both techniques and obtain significant improvements.

8.3 Combining Ensemble Modeling and AUC of the ROC curve optimization

The merging of both techniques results in a robust approach. On one hand, *ensemble modeling* provides specificity to the models required for each condition (for example, SNR, channel). On the other hand, *optimization of the area under the ROC curve* updates, in a discriminative way, the model parameters. Maximizing the AUC produces an impact at every operating point, not only focusing on points of interest like the EER or the minDCF. Moreover, this approach can be embedded into known algorithms such as MVE or JFA. A natural step is to extend the methodology to noise conditions.

Figure 8.2 describes the methodology followed, going from the traditional training of the target and competing models to the refinement of those models using the *ensemble modeling* and the *AUC of the ROC curve*.

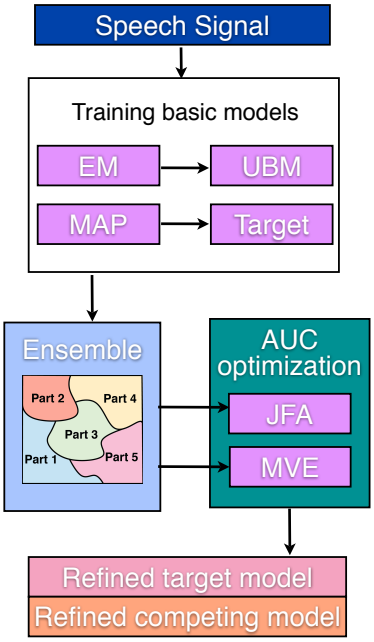


Figure 8.2: Ensemble-AUC modeling modeling in detail

8.3.1 Partitioning the data space (Ensemble)

The first step is to partition the data space, employing *ensemble modeling*. The designer decides which attribute to explore for this purpose. An intuitive way to do it is to choose an attribute for each level. From the complete data space, decide on a top partition. For our special case, we started with a channel partition. In the next layer, for example, we clustered the data according to the noise type and obtained further partitions for each possible SNR. It is possible to apply the same procedure to subsequent levels.[1]

As in the simple *ensemble* approach, at the first level the signal space is partitioned into P non-overlapping regions $\Omega_1, \Omega_2, \cdots, \Omega_P$, such that together these partitions cover the entire space. We train GMM target models for each of the partitions and a usual UBM from which we can perform the adaptations. We assume that the recordings are clustered into P groups, each corresponding to a set of signals that fall into one of these partitions.

According to [140], it is possible to build subspaces depending on the attributes of the speaker signal: channel, speaker, noisy condition and SNR. To maximize the class separation (meaning the separation among different environments or channel), we use a misclassification measure that is susceptible to changes in a certain condition. Following this work, two options were explored in this research:

- *Enviroment Clustering (EC)*: A hierarchical clustering, where the data space is the root of a tree that includes all the environments. It partitions into layers with several

[1] A common constraint is the amount of data available. For this approach, we assume that all the data is at our disposal.

branches, as described in Section 8.1. Each contiguous cluster on a same layer contains similar acoustic characteristics. For a hierarchical tree of P partitions, we denote the subspaces as: $\Omega = \{\Omega_1 \cup \Omega_2, ... \cup \Omega_u ... \cup \Omega_P\}$. Each of the these clusters is represented by a super-vector Ω_{rep^u} by a function $\mathcal{R}(\cdot)$. Hence, $\Omega_{rep^u} = \mathcal{R}(\Omega_u)$.

- *Environment Partitioning (EP)*: From the full data space we generate a GMM model, similar to the UBM. Then, we partition the GMM components according to similarity measures such as Mahalanobis, Bhattacharyya, or the divergence. Each subset $u \in \Omega_P$ of components define the new region.[2] The data space vectors establish a membership to these new clusters. Once again, $\Omega_{GMM^u} = \mathcal{R}(\Omega_u)$.

For both cases, the new objective function and misclassification measure for U utterances in P different conditions take the form,

$$\ell(\Omega_u) = \frac{1}{U} \sum_{U} \frac{1}{1 + \exp\left[-\gamma \left(d\left(\chi_u, \Omega_u, \Lambda_u\right) + \theta\right)\right]}, \tag{8.2}$$

where χ_u is an utterance in the training data for partition u, Ω_u belongs to a spanned subspace Ω, γ and θ are control parameters and Λ_u represent the GMM parameters for that specific region. Accordingly, the misverification measure,

$$d\left(\chi_u, \Omega_u, \Lambda_u\right) = -g\left(\chi_u, \Omega_u, \Lambda_u\right) + G\left(\chi_u, \Omega_u, \Lambda_{\hat{u}}\right), \tag{8.3}$$

and

$$G\left(\chi_u, \Omega_u, \Lambda_{\hat{u}}\right) = \log\left\{\frac{1}{U-1} \sum_{j, j \neq u} \exp\left[\eta \times g\left(\chi_u, \Omega_j, \Lambda_j\right)\right]\right\}^{\frac{1}{\eta}}. \tag{8.4}$$

where η is positive, $\hat{u}$ are the regions not belonging to u, and G takes into account the contribution of competing models. Equations 8.3 and 8.4 show the conditional dependence. We have $P-1$ competing models that are optimized. The above formulation can be employed for the next approaches.

The solution is given by applying the usual GPD update rule for Λ_u,

$$\Lambda_u^{t+1} = \Lambda_u^t - \epsilon \nabla \ell(\Lambda_u)|_{\Omega_u^t}. \tag{8.5}$$

Following the previous algorithm, it is possible to explore channel mismatch, noisy environment, and a combination of the two.

[2] Another possible solution is to use an acoustic-based unit to form such clusters [140]. For a feature-based approach, the super-vector is partitioned into Z sub-vectors, (for example, static, delta and double-delta coefficients).

8.3.2 Refining the Models (AUC approach)

The models obtained by the *ensemble* approach depend not only on the neighboring models, but can also be optimized by maximizing the AUC of the ROC curve, hence reducing the EER and minDCF. A certain partition u becomes a target model, the rest of the partitions become the competing models.

Let χ represent the complete set of target and competing training instances: $\chi = \mathcal{H} \cup \mathcal{W}$, where $\mathcal{H}$ contains all the target instances and $\mathcal{W}$ all the competing instances. Thus, the modified AUC function to optimize a specific partition u is,

$$\Upsilon(\chi, \Lambda_u) = 1.0 - \frac{\sum_{\chi \in \mathcal{H}} \sum_{\hat{\chi} \in \mathcal{W}} R(\theta_u(\chi), \theta_{\hat{u}}(\hat{\chi}))}{|\mathcal{H}||\mathcal{W}|}, \tag{8.6}$$

θ_u and $\theta_{\hat{u}}$ are the usual scores of χ with respect to a target model Λ_u or competing model $\Lambda_{\hat{u}}$, and R is the soft representation of the sigmoid function. The modified AUC function of Equation 8.6 can then be optimized using the GPD algorithm,

$$\Lambda_u^{t+1} = \Lambda_u^t - \epsilon \nabla \Upsilon(\chi, \Lambda_u) \tag{8.7}$$

$$\nabla \Upsilon(\chi, \Lambda_u) = -\frac{1}{|\mathcal{H}||\mathcal{W}|} \sum_{\chi \in \mathcal{H}} \sum_{\hat{\chi} \in \mathcal{W}} \gamma R(1 - R) \left[\frac{\partial \theta_u(\chi)}{\partial \Lambda_u} - \frac{\partial \theta_{\hat{u}}(\hat{\chi})}{\partial \Lambda_{\hat{u}}} \right]. \tag{8.8}$$

In the above equation, R is a shorthand notation for $R(\theta(\chi), \theta(\hat{\chi}))$, and ϵ is a learning rate parameter.

The general summary process is depicted in Figure 8.3. From a full data space, the algorithm performs a partition among a specific *attribute* (first layer). We optimize and maximize the separation of this new models; *i.e.,* the models are refined. In a second stage, the algorithm maximizes the separation within the current clusters, and refine them. In the final stage, for the special case of SV, we have to optimize the target speaker against the impostors.

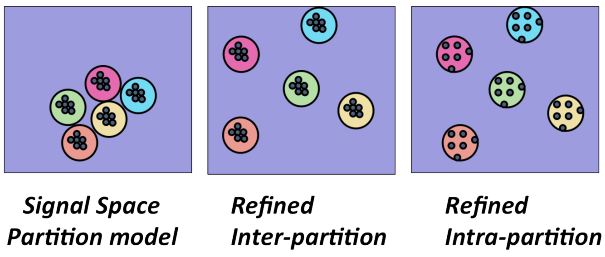


Figure 8.3: Ensemble-AUC scheme

Two clear applications arise from the above methodology: channel mismatch and noisy environments.

8.3.3 Channel Mismatch

The channels in the recordings used for training and testing might be different, causing poor system performance. To solve this issue, we introduce the channel as the intended *ensemble* attribute. We start from the premise that a gende-dependent UBM embraces the universe of all possible conditions and we will represent the Total Space as T.

Environment clustering (EC) and environment partitioning (EP) are suitable off-line options to distribute T and to prepare the channel-dependent regions. For example, the sub-partitions belonging to Ω_u can differentiate between microphone and telephone at a first level of the hierarchical structure.

Afterwards, we can refine the sub-region models by increasing coverage. Using the a priori information from the channel clustering and partitioning, we can update the models by performing inter-channel training denoted as Λ_{Q_u}, where Q_u represents the partition among a condition, followed by intra-channel training (computing $\Lambda_{\Omega_{Q_u}}$ among the competing models). The former will produce a spread among the channel sub-spaces; the latter will broaden the coverage. The final models are constructed accordingly.

For a test vector, we estimate the model from which it was produced, stochastic matching [141] is used for this purpose. Q_u i estimated according to the chosen EC or EP algorithm. For both, a cluster selection is performed based on the highest likelihood to the testing data χ_{Y}. The mapping of such utterance is given by

$$\Omega_{Q_u} = \underset{u}{\text{argmax}}\ P(\chi_{\text{Y}}|R(\Omega_{Q_u})), \tag{8.9}$$

$$Q_{Y_u} = \mathbf{G}_{\phi}(\Omega_{Q_u}), \tag{8.10}$$

where Y refers to the test data, and G_ϕ is a function that transforms the models from a training channel domain to the new test domain.

8.3.4 Noisy Environment

Once again, for this approach, we take the total data space T as our starting point. In this case, EC and EP are used to produce the sub-regions comprising Ω_e, consistent with the noise type and SNR. According to [140], after gender separation, a second layer of the hierarchical tree divides the sub-spaces in low and high SNR. We employ this methodology with a later refinement of each sub-region using the known algorithm to increase the coverage. The inter-environment algorithm maximizes the separation among the sub-regions and computes Q_e (each partition refers to a cluster with certain low or high SNR). Moreover, the coverage within each sub-region is broadened with the intra-environment technique, producing Ω_{Q_e}.

The final on-line models, Q_{Y_e} are estimated according to the chosen EC or EP, considering,

$$Q_{Y_e} = \mathbf{G}_{\phi}(\Omega_{Q_e}), \tag{8.11}$$

G_{ϕ} is a mapping function that relates the training noisy condition with the test condition.

8.3.5 Channel Mismatch and Noisy environment

This combined scheme is a summary of the previous approaches. The first step is to perform a gender separation. Afterwards, we use EC to compute Ω_u using a channel basis with the corresponding refinement. In the third layer of a hierarchical tree the noise type and SNR are included, resulting in a set of Ω_c^e sub-spaces. These final sub-spaces contain the information of channel, noise type, and SNR. The next step is to increase the separation among Ω_u^e sub-spaces, computing Q_u^e. Moreover, to increase the coverage, we can compute $\Omega_{Q_u^e}$.

The online models, $Q_{Y_u^e}$ are estimated according to

$$Q_{Y_u^e} = \mathbf{G}_{\phi}(\Omega_{Q_u^e}), \tag{8.12}$$

where G_{ϕ} is a mapping function that transforms the models from a training domain to the test domain.

8.4 Speaker modeling

For each target speaker we perform a similar procedure. After employing any of the above analyses, it is possible to refine the speaker space by maximizing the separation among competing speakers using inter-speaker training.

For example, a new objective function and misclassification measure for U utterances from a particular Speaker s in different conditions take the form,

$$\ell(\Omega_s) = \frac{1}{U}\sum_{U}\frac{1}{1+\exp\left[-\gamma\left(d\left(\chi_u, \Omega_{u,s}^e, \Lambda_u\right)+\theta\right)\right]}, \tag{8.13}$$

where χ_u is an utterance in the training data for partition u, $\Omega_{s,u}^e$ belongs to a spanned subspace Ω (that considers the channel and the SNR), γ and θ are control parameters, and $\Lambda_{u,s}^e$ represents the GMM parameters for that specific region. Accordingly,

$$d\left(\chi_u, \Omega_{u,s}^e, \Lambda_u\right) = -g\left(\chi_u, \Omega_{u,s}^e, \Lambda_u\right) + G\left(\chi_u, \Omega_{\hat{u},s}^e, \Lambda_{\hat{u}}\right), \tag{8.14}$$

where

$$g\left(\chi_u, \Omega^e_{u,s}, \Lambda_u\right) = \log\ p(\chi; \Omega^e_{u,s}, \Lambda_u),$$
$$G\left(\chi_u, \Omega^e_{u,s}, \Lambda_{\hat{u}}\right) = \log\ p(\chi; \Omega^e_{\hat{u},s}, \Lambda_{\hat{u}}). \tag{8.15}$$

Equations 8.14 and 8.15 show the conditional dependence. If we just want to finish the process by increasing coverage, we can solve by applying the (GPD) update rule for $\Lambda^e_{u,s}$,

$$\Lambda^{t+1}_{u,e,s} = \Lambda^t_{u,s,e} - \epsilon \nabla \ell(\Lambda_{u,s,e})|_{\Omega^t_{u,s,e}}. \tag{8.16}$$

For the purpose of this work, we present a formulation using an optimization of the AUC of the ROC curve. Hence, we explore two approaches as in previous Chapters, MVE and JFA.

8.4.1 MVE in an Ensemble approach

For the MVE approach we define the following objective function that represents the WMW statistics,

$$\Upsilon(\Lambda_{u,e}) = 1.0 - \frac{\sum_{\chi\in\mathcal{H}} \sum_{X\in\chi} \sum_{\hat{\chi}\in\mathcal{W}} \sum_{\hat{X}\in\hat{\chi}} R(\theta_{u,e}(X), \theta_{u,e}(\hat{X}))}{\sum_{\chi\in\mathcal{H}} L_\chi \sum_{\chi\in\mathcal{W}} L_\chi}, \tag{8.17}$$

where L_χ is the number of feature vectors in χ. Note that the formulation depends on the noise type and the channel-refined space is given by $\Omega_{u,e}$. The scores θ are computed according to this space.

Using the representation

$$\sum_{\chi\in\mathcal{H}} L_\chi = |\mathcal{H}|, \qquad \sum_{\chi\in\mathcal{W}} L_\chi = |\mathcal{W}|,$$

the GPD update rule for any parameter ϕ of the distributions is now given by $\phi_{t+1} = \phi_t - \epsilon \nabla_\phi \Upsilon(\mathbf{X}, \Lambda_{u,e})$, where

$$\nabla_\phi \Upsilon\left(\mathbf{X}, \Lambda_{u,e}\right) = -\frac{1}{|\mathcal{H}||\mathcal{W}|} \sum_{\chi\in\mathcal{H}} \sum_{X\in\chi} \sum_{\hat{\chi}\in\mathcal{W}} \sum_{\hat{X}\in\hat{\chi}} \gamma R(1-R) \nabla_\phi l(X, \hat{X}, \Lambda_{u,e}), \tag{8.18}$$

and $\nabla_\phi l(X, \hat{X}, \Lambda_{u,e})$ is a local gradient with respect to ϕ at $\chi, \hat{\chi}$ and has the form given by Equation 8.19,

$$\nabla_\phi l(X, \hat{X}, \Lambda_{u,e}) = -\frac{\partial \theta_{u,e}(X)}{\partial \phi} + \frac{\partial \theta_{u,e}(\hat{X})}{\partial \phi}, \tag{8.19}$$

$\frac{\partial \theta_{u,e}(X)}{\partial \phi}$ represents the derivative of the log-likelihood-difference given by the Gaussian

mixture models for the target speaker and the universal background model for vector X with respect to ϕ.

The final solution can be obtained in two ways: a scoring method or stochastic matching. The scoring techniques described in Section 6.5 uses a set of scores for different partitions to form a fusion scheme. Stochastic matching is also a possible solution to decide the environment for a particular phrase without needing to perform a scoring method.

8.4.2 Factor Analysis in an Ensemble Approach

The proposed *ensemble* method pursues the same goal as JFA: to extract the information of the speaker while marginalizing the contribution of any variability caused by channel or noise conditions. Going further, we include the *ensemble* approach for the i-vector and refine the current models so that the i-vectors will properly cover the space for different conditions. Recall that the utterance is defined as,

$$M = s + T\omega, \tag{8.20}$$

where M represents the combined speaker and channel-independent supervector extracted from the UBM. T is a low-rank rectangular matrix and ω is a random vector, composed of identity vectors (*i-vectors*), with normal distribution $N(0, I)$. M is normally distributed with mean m and covariance matrix TT'.[3] Hence, the new modeling is in fact a projection of an utterance onto the low-dimensional total variability space.

In an *ensemble* scenario, the space is partitioned into several regions depending on the desired channel and/or noisy condition. Hence, let $\mathbf{X}$ represent a large collection of recordings χ obtained from a large number of speakers in a certain $\Omega_{u,e}$. The starting point for the training are the models $\Lambda_{u,e}$, defined by those regions. For this formulation we also assume that the score is a likelihood ratio, so that the optimization holds.

Let $\mathcal{S}$ be the set of all speakers in $\mathbf{X}$. Let $\mathbf{X}_S$ represent the subset of $\mathbf{X}$ representing recordings from speaker S and $\mathbf{X}_{\bar{S}}$ be recordings from remaining speakers, *i.e.* $\mathbf{X} = \mathbf{X}_S \cup \mathbf{X}_{\bar{S}}$. Note that, $\mathbf{X}$ can be partitioned in this manner in as many ways as there are speakers in $\mathcal{S}$.

To learn global parameters, we define the following AUC objective for each region $\Omega_{u,e}$.

$$\Upsilon(\Lambda_{u,e}) = 1.0 - \frac{1}{|\mathbf{X}_S||\mathbf{X}_{\bar{S}}|} \sum_{S \in \mathcal{S}} \sum_{\chi \in \mathbf{X}_S} \sum_{\chi \in \mathbf{X}_{\bar{S}}} R(\theta_S^{u,e}(\chi), \theta_S^{u,e}(\hat{\chi})) \tag{8.21}$$

The GPD update rule for any global parameter $\phi^{u,e}$ is given by $\phi_{t+1}^{u,e} = \phi_t^{u,e} -$

[3]The procedure to train the matrix T is very similar to training the loading matrix V in JFA, but in eigenvoice training, all the utterances from a speaker are treated as different speakers. Refer to Appendices B.2 and B.3 for further details.

$\epsilon \nabla_{\phi^{u,e}} \Upsilon(\mathbf{X}, \Lambda_{u,e})$, where

$$\nabla_{\phi^{u,e}} \Upsilon(\mathbf{X}, \Lambda_{u,e}) = -\sum_{S \in \mathcal{S}} \frac{\sum_{\chi \in \mathbf{X}_S} \sum_{\chi \in \mathbf{X}_{\bar{S}}} \gamma R(1-R), \nabla_{\phi^{u,e}} \, l(\chi, \hat{\chi}, \Lambda_{u,e})}{|\mathbf{X}_S||\mathbf{X}_{\bar{S}}|} \tag{8.22}$$

$\nabla_{\phi^{u,e}} l(\chi, \hat{\chi}, \Lambda_{u,e})$ is a local gradient with respect to $\phi^{u,e}$ at $\chi, \hat{\chi}$, and

$$\nabla_{\phi^{u,e}} l(\chi, \hat{\chi}, \Lambda_{u,e}) = -\frac{\partial \theta_{u,e}(\chi)}{\partial \phi^{u,e}} + \frac{\partial \theta_{u,e}(\hat{\chi})}{\partial \phi^{u,e}} \tag{8.23}$$

To derive the update rules for individual parameters, it is sufficient to obtain the derivatives with respect to the corresponding loading matrix T (refer to Chapter 7 for details). Moreover, we can optimize the i-vector in a similar manner. The next step, employing the PLDA is straightforward for each of the subspaces $\Omega_{u,e}$. Consequently, we can employ a fusion scheme to decide the most appropriate scoring solution.

The global approach proposed is potentially successful since it structures the space in advance, giving the opportunity to use a priori information for better modeling. Furthermore, the technique has been presented for a SV scenario, under different real conditions. Note that this space refinement can be properly included in other optimization schemes, such as an ML-MAP approach.

This scheme has similarities with JFA and probably pursues the same goal, the marginalization of variability not belonging to the speaker. On one hand, *ensemble modeling* segments the space into regions in which the classification is localized for those specific partitions. On the other hand, the AUC of the ROC curve refines the models for every operating point. But the main purpose is to be employed in noisy conditions. In this case, the target partitions are modeled so that the localized operating points are improved.

Algorithm 8.1. Integration of the AUC optimization and Ensemble Methods I

Initialization

1. Train the general model (UBM) in the usual way employing EM,

$$\bar{\Lambda} = \{\bar{W}_k^C, \bar{\mu}_k^C, \bar{\Sigma}_k^C\}. \tag{8.24}$$

2. Decide the number of levels in the hierarchy and its type. Usually we get two levels: one for the channel and one for the SNR.

Algorithm 8.1. Integration of the AUC optimization and Ensemble Methods II

Ensemble Modeling

1. Obtain the channel partitions Ω_u and their corresponding models,

$$\Lambda_u = \{W_k^u, \mu_k^u, \Sigma_k^u\}. \tag{8.25}$$

2. Training the Ensemble Model: Choose between EC or EP.

   - *Enviroment Clustering (EC)*: For a hierarchical tree of P partitions, we can denote the subspaces as: $\Omega = \{\Omega_1 \cup \Omega_2, ...\Omega_u, ... \cup \Omega_P\}$. Each of the these clusters can be represented by a super-vector Ω_{rep^u} by a function $\mathcal{R}(\cdot)$. Hence, $\Omega_{rep^u} = \mathcal{R}(\Omega_u)$.
   - *Environment Partitioning (EP)*: Generate from the full data space a GMM model and partition the GMM components according to similarity measures. Each subset $u \in \Omega$ of components define the new region. Once again, $\Omega_{GMM^u} = \mathcal{R}(\Omega_u)$.

3. Define the loss function for U utterances in P conditions by,

$$\ell(\Omega_u) = \frac{1}{U} \sum_U \frac{1}{1 + \exp\left[-\gamma \left(d\left(\chi_u, \Omega_u, \Lambda_u\right) + \theta\right)\right]}, \tag{8.26}$$

   where χ_u is an utterance in the training data for partition u, Ω_u belongs to the spanned subspace Ω, γ and θ are control parameters and Λ_C represents the GMM parameters for that specific region.

4. Define the misclassification measure

$$d\left(\chi_u, \Omega_u, \Lambda_u\right) = -g\left(\chi_u, \Omega_u, \Lambda_u\right) + G\left(\chi_u, \Omega_u, \Lambda_{\hat{u}}\right), \tag{8.27}$$

$$G\left(\chi_u, \Omega_u, \Lambda_{\hat{u}}\right) = \log\left\{\frac{1}{U-1} \sum_{j, j \neq u} \exp\left[\eta \times g\left(\chi_u, \Omega_j, \Lambda_j\right)\right]\right\}^{\frac{1}{\eta}}. \tag{8.28}$$

   where η is positive, $\hat{u}$ are the regions not belonging to u, and G takes into account the contribution of competing models.

5. The solution is given by applying the (GPD) update rule for Λ_u,

$$\Lambda_u^{t+1} = \Lambda_u^t - \epsilon \nabla \ell(\Lambda_u)|_{\Omega_u^t}. \tag{8.29}$$

6. Check for convergence
   (a) Compute the loss function with the new parameters:

$$\ell_{new}(\Lambda_u) = \frac{1}{U} \sum_U \frac{1}{1 + \exp\left[-\gamma \left(d\left(\chi_u, \Omega_u, \Lambda_u\right) + \theta\right)\right]} \tag{8.30}$$

   (b) Compare ℓ_{new} with ℓ. If it is greater than a threshold τ, iterate.

Algorithm 8.1. Integration of the AUC optimization and Ensemble Methods III

Refining the Models (AUC approach)

Let χ represent the complete set target and competing training instances: $\chi = \mathcal{H} \cup \mathcal{W}$, where $\mathcal{H}$ contains all the target instances and $\mathcal{W}$ all the competing instances.

1. Compute the modified AUC function to optimize for a specific partition u

$$\Upsilon(\chi, \Lambda_u) = 1.0 - \frac{\sum_{\chi \in \mathcal{H}} \sum_{\hat{\chi} \in \mathcal{W}} R(\theta_u(\chi), \theta_{\hat{u}}(\hat{\chi}))}{|\mathcal{H}||\mathcal{W}|}. \tag{8.31}$$

   θ and $\hat{\theta}$ are the usual scores of χ with respect to a target model Λ_u or competing model $\Lambda_{\hat{u}}$, and R is the soft representation of the sigmoid function.

2. Optimize Equation 8.31 using the GPD algorithm using the following rule, $\Lambda_u^{t+1} = \Lambda_u^t - \epsilon \nabla \Upsilon(\chi, \Lambda_u)$. Therefore,

$$\nabla \Upsilon(\chi, \Lambda_u) = -\frac{1}{|\mathcal{H}||\mathcal{W}|} \sum_{\chi \in \mathcal{H}} \sum_{\hat{\chi} \in \mathcal{W}} \gamma R(1-R) \left[\frac{\partial \theta_u(\chi)}{\partial \Lambda_u} - \frac{\partial \theta_{\hat{u}}(\hat{\chi})}{\partial \Lambda_{\hat{u}}} \right]. \tag{8.32}$$

3. Check for convergence

   (a) Compute the loss function with the new parameters:

$$\Upsilon_{new}(\chi, \Lambda_u) = 1.0 - \frac{\sum_{\chi \in \mathcal{H}} \sum_{\hat{\chi} \in \mathcal{W}} R(\theta_u(\chi), \theta_{\hat{u}}(\hat{\chi}))}{|\mathcal{H}||\mathcal{W}|}. \tag{8.33}$$

   (b) Compare Υ_{new} with Υ if it is greater than a threshold τ, iterate.

Speaker Modeling

1. Compute the misverification function and for U utterances for a speaker s,

$$\ell(\Omega_s) = \frac{1}{U} \sum_{U} \frac{1}{1 + \exp\left[-\gamma \left(d\left(\chi_u, \Omega^e_{u,s}, \Lambda_u\right) + \theta\right)\right]}. \tag{8.34}$$

   where χ_u is an utterance in the training data for partition u, $\Omega^e_{s,u}$ belongs to the spanned subspace Ω (that considers the channel and the SNR), γ and θ are control parameters and $\Lambda^e_{u,s}$ represent the GMM parameters for that specific region.

Algorithm 8.1. Integration of the AUC optimization and Ensemble Methods IV

2. Compute the misverification measure,

$$d\left(\chi_u, \Omega^e_{u,s}, \Lambda_u\right) = -g\left(\chi_u, \Omega^e_{u,s}, \Lambda_u\right) + G\left(\chi_u, \Omega^e_{\hat{u},s}, \Lambda_{\hat{u}}\right), \tag{8.35}$$

where

$$\begin{aligned} g\left(\chi_u, \Omega^e_{u,s}, \Lambda_u\right) &= \log\ p(\chi; \Omega^e_{u,s}, \Lambda_u), \\ G\left(\chi_u, \Omega^e_{u,s}, \Lambda_{\hat{u}}\right) &= \log\ p(\chi; \Omega^e_{\hat{u},s}, \Lambda_{\hat{u}}). \end{aligned} \tag{8.36}$$

2. Solve by applying the (GPD) update rule for $\Lambda^e_{u,s}$,

$$\Lambda^{t+1}_{u,e,s} = \Lambda^t_{u,s,e} - \epsilon \nabla \ell(\Lambda_{u,s,e})|_{\Omega^t_{u,s,e}}. \tag{8.37}$$

2. Decide which method to use for optimizing: i-vector or MVE. Solve accordingly.

8.5 Experiments and Results

This section describes the *integration* of both techniques and the final results up to now. Note that in this part we include experiments not only using traditional approaches, but also with the i-vectors, considering it just as a frontend from which we can train a model. See Tables 8.1 and 8.2.

	5c-MAP					5c -MVE				
Baseline	28.6					28.3				
minDCF	15.60					15.32				
Condition	PS	AP	BS	CO	FU	PS	AP	BS	CO	FU
k-means	27.2	23.4	28.6	21.3	19.4	25.1	22.3	25.9	21.2	18.4.
minDCF	15.73	12.35	17.43	10.57	8.19	11.56	8.44	12.32	8.14	6.57
prior info	23.2	21.5	24.3	20.0	17.2	21.6	19.9	22.3	18.7	16.2
minDCF	12.88	8.03	13.2	7.84	6.56	7.91	7.53	8.54	7.23	6.68

Table 8.1: Integration approach: EER and minDCF for different clusters on noise condition, MAP and MVE.

	5c-JFA					5c i-vector				
Baseline	26.3					25.0				
minDCF	14.03					13.23				
Condition	PS	AP	BS	CO	FU	PS	AP	BS	CO	FU
k-means	24.9	23.4	25.6	23.2	20.4	24.2	23.1	24.6	22.3	19.5
minDCF	13.85	13.63	15.72	12.74	7.36	12.72	12.47	14.85	11.87	6.84
prior info	23.3	22.5	23.3	21.6	19.0	22.3	21.2	22.5	20.7	18.1
minDCF	9.77	10.13	12.78	8.11	7.51	8.71	9.16	11.90	7.31	6.05

Table 8.2: Integration approach: EER and minDCF for different clusters on noise condition, JFA and i-vector.

8.5.1 Experimental Setup

The setup for this experiments follows exactly the same procedure showed in the *ensemble modeling*. We employed the NIST Speaker Evaluation 2004, 2005, 2010 and 2008 database (see Section 7.7.1). We conducted an experiment on noise-corrupted data. For all experiments, we used the NIST2008 SRE database as targets. Babble noise, extracted from the Aurora 2 database, was added randomly to the training and test files at different SNRS: 0, 5, 10, 15 and 20 dB. Moreover, we selected the conditions and channel as attributes to explore.

First, we built a hierarchical tree, with a first layer that focuses on the channel attributes and a second layer that deals with the noise conditions. The results were obtained for five cluster partitions in the first layer, plus a refinement of those models. Two channels were considered: microphone and telephone. No further partitions were explored within these categories. The second layer accommodates the noise partition for different SNR (considering this experiments were performed just using babble noise). MAP and MVE approaches were employed as explained in Chapter 7. The JFA experiments used 50 eigenchannels for U, and 400 eigenvoices for V for every region. This experiment shows the results for a simple i-vector approach as described in Chapter 3. We extracted a set of 300 i-vectors that were followed by the reduction to 100 using LDA. Afterwards, these new i-vectors are processed by the PLDA. The five methods: *partition selection*,(PS), *a priori*, (AP), *best score*, (BS), *combination*, (CO), and *fusion*,(FU), were evaluated. Their corresponding unsupervised (k-means) and supervised partitioning were also explored.

8.5.2 Results

Tables 8.1 and 8.2 depict the results for the different optimizations and *ensemble* approaches; the AUC optimization is implicit in these results. We note that the best results were obtained for the combination(CO) and fusion (FU) in terms of EER and minDCF. The current *ensemble* approach helps the current modeling procedures such as JFA and MVE.

Surprisingly, the MVE optimization obtains the best results for this specific noise condition.

Figure 8.4 shows the results of the AUC of the ROC curve for all possible cases.

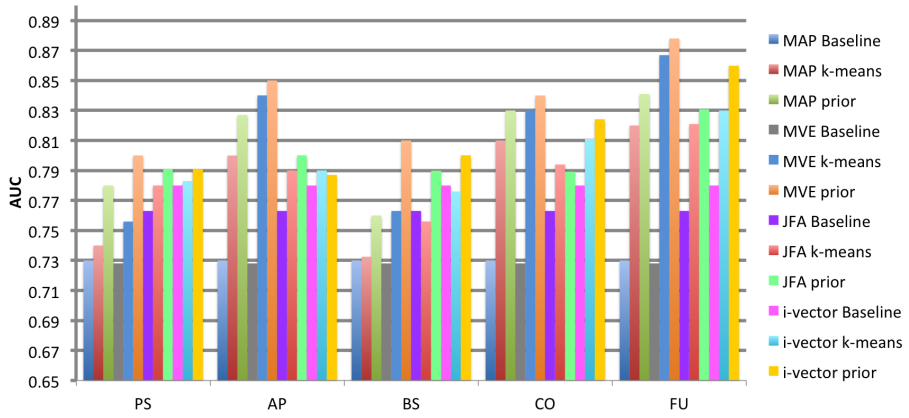


Figure 8.4: Integration approach: AUC results for babble noise condition by scoring method.

We comment on the improvements in terms of the scoring selection methods. For every set of results the greatest AUC was obtained by the MVE-related approaches, supporting the potential of this method in noise condition.

Figure 8.5 shows once again the results of the AUC, but it considers the optimization methods employed. Note that the highest results are obtained for the scoring methods: combination(CO) and fusion (FU). As expected, the results show that considering the prior information is of great benefit to any of the approaches. Moreover, for the unsupervised method, the improvements are still noticeable.

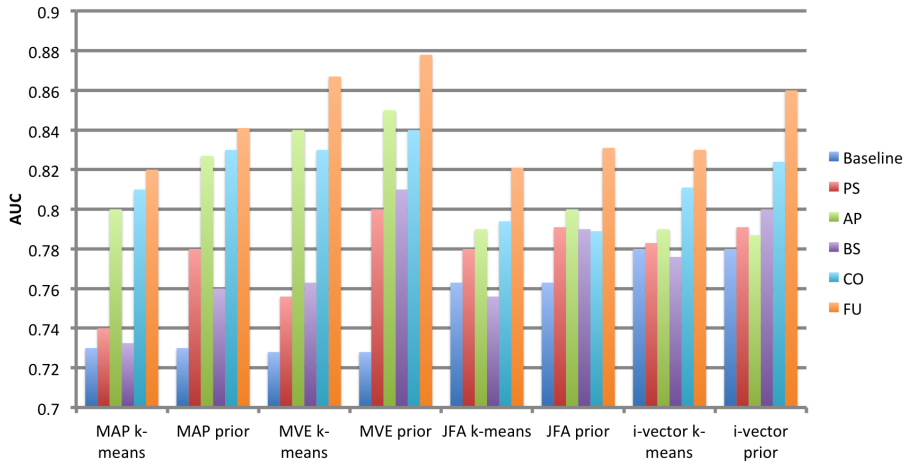


Figure 8.5: Integration approach: AUC results for babble noise condition by optimization approach.

Lastly, Figure 8.6 shows the relative improvements with respect to the most successful scoring methods: combination (CO) and fusion (FU). We observe that the MVE-based approaches outperform the current baselines such as MVE and JFA. In addition, the

ensemble modeling alone improves the results within its optimization method, specially when the prior information is known. Note also that the results considered here are a cocktail of noise samples with 0 to 15 dB SNR.

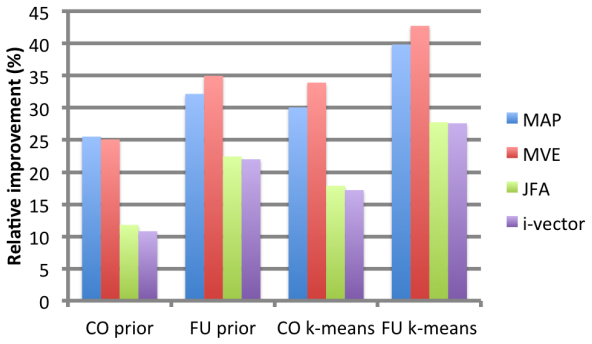


Figure 8.6: Integration approach: relative improvements (babble noise condition)

8.6 Summary

In this chapter we described the integration of both techniques, the *AUC of the ROC curve optimization* and the *ensemble modeling*. We provided some insight into how the process takes place and gave some experimental examples. We observed that performing the *integration* the two can improve the actual baselines especially under noisy conditions. We concluded that by including the *ensemble* approach, the models become more specific. Furthermore, considering the AUC optimization, the ROC curve is optimized at every point producing in improvements for every modeling scheme.

Chapter 9

Discussion

> Sin que nadie se las haya dicho, el indio sabe muchas cosas. El indio lee con sus ojos tristes lo que escriben las estrellas que pasan volando...
>
> - Antonio Mediz Bolio *La tierra del Faisán y del Venado*

In our aim to investigate the performance of our proposed methodologies in other scenarios, we extended our analysis to other databases. In this chapter, first we give a brief overview of the databases used, including the baseline results. Next, we show results for the AUC of the ROC curve, the Ensemble Modeling and the *integration* approaches. The databases employed are YOHO, MOBIO and Ahumada/Gaudi. They involve challenges such as different languages and mismatched condition.

9.1 MOBIO database

MOBIO [142] is a large bi-modal (audio/visual) database divided into two phases. For the purpose of this thesis we just included Phase I [143]. It comprises both speech and face recognition of 161 participants in six sessions (57 female, 104 male). The combined signals were recorded and saved in mp4 format from which we extracted the speech signals.

The database was recorded as follows:

A) Set responses: the users were asked questions such as: *What is your name?* (around 7 seconds)

B) Read Speech from a paper: the users read 3 fixed sentences (maximum 30 seconds)

C) Free speech: the users provide five to ten second answers to random questions.

The phrases were recorded in six sessions. Each session included 21 questions: 5 set responses, 1 read speech sample and 15 free speech recordings. The evaluation data split in: enrollment, development and testing. The training of the UBM included the set (B) of response questions, (we employed a total of 4893 audio files for male and 1764 for female). The enrollment of the target users is composed of set of responses (A). Each target speaker then has 5 training files. The testing set includes the free speech answers, set (C) for sessions 2 to 6 (we used a total of 1040 female and 1120 for male).

Other databases used in this work, particularly for the discriminative training (as in previous sections) are Switchboard-1 (SW1) and Switchboard-2 (SW2) [117]. These two databases contain telephone recordings and were used in the same manner as in the training of JFA (see Section 5.4).

9.1.1 MOBIO Results

MOBIO database represents our first attempt to extend our findings to a new database. Although, belonging to a multimodal database, we just used the speech signal to perform the experiments.

9.1.2 Experimental Setup

The traditional front-end process was employed for two setups:

- System 1: Feature vector of 33 attributes: 16 static Cepstral, 1 log Energy, and 16 delta Cepstral coefficients. We selected a single file from each target user (the average time of these utterances is 7 seconds) for training phase.
- System 2: Feature vector of 49 attributes: 16 static Cepstral, 1 log Energy, 16 delta Cepstral coefficients, and 16 double delta Cepstral coefficients. The complete set of target files were included to compute the target models (5 files).

For a first approach, a gender-dependent and target-independent 512-mixture GMM UBM was trained employing a pool of the MOBIO speech database. For the second system, a 512-mixture GMM UBM was trained using the NIST-setup, as described in Section 5.3 (NIST databases: 2004, 2005, 2006, 2010). For both approaches, the target-dependent models were then obtained employing MAP. JFA experiments used the basic training scheme, employing 90 eigenchannels for the latent variables U and 200 for the eigenvoices V.

The discriminative training for MVE, the maximization of the ROC curve, and the *ensemble* used a selection of cohort samples: 500 phrases from NIST 2004, 2005 2006, switchboard 1, switchboard 2 and 500 phrases from NIST 2010 microphone. A total of 3000 phrases were used as imposter samples. For *ensemble modeling*, we just used a five

partitions of the imposter data space, the partition was performed blindly, just the k-means was used for this experiments and the different scoring methodologies' outcomes were fused to obtain a final score. For the *integration*, we partitioned the imposter space into five different regions, the partition was performed blindly once again, and just the score fusion is presented.

The noise-conditions experiments used babble noise extracted from the Aurora 2 database at different SNRs: 0,5,10,15 and 20 dB. The *ensemble* training used five partitions; the same is applied to the *integration* (for every SNR-region the new data space is partitioned into two new regions).

9.1.3 Results

Table 9.1 presents baseline results for the proposed systems. These results are just informative. We observe how difficult it is for the system to update the parameters for very short utterances. However, when training the models with the complete data available, the verifier performance improved.

The main purpose of System 1 is to test MOBIO database in the worst scenario (baseline) following the same rules as in NIST evaluations, using one file to train the target model. For System 2, we considered all the phrases available per speaker and the improvement is evident.

	Male	*Female*	*Average*
System 1	20.55	25.23	22.89
System 2	15.45	17.41	16.43

Table 9.1: Explorative results (EER) on the Test set for the MOBIO database.

The results obtained by our approaches for MOBIO database are summarized in Table 9.2. We used System 2 as our baseline. JFA provides the greatest improvements. Although, there is a gain from the maximization of the ROC curve, the improvements are not as great as for the JFA case. The optimization performed using the *integration* presents the best results. The specificity of the trained models for every channel and the optimization of the ROC curve improved the current results.

Table 9.2 shows results for noise conditions as well. Note that although they follow similar trends as the clean results, the discriminative optimizations work better than the FA approaches in general. The best result was obtained using the *integration* method. Once again, the optimization of the AUC of ROC curve improves the MVE and FA analysis based approaches.

Figure 9.1 shows the results for the actual AUC as a metric for several cases. We

	Clean		*Noise*	
	EER	*minDCF*	*EER*	*minDCF*
Baseline MAP	16.43	9.2	21.68	14.5
MVE	14.55	8.4	15.21	10.4
JFA	11.81	7.6	18.45	12.3
MVE-AUC	12.32	8.0	14.06	9.7
JFA-AUC	10.27	7.2	13.1	8.5
Ensemble Modeling	11.33	7.3	12.41	8.3
Integration	9.96	6.8	10.97	6.6

Table 9.2: Final results (EER and minDCF) on the Test set for MOBIO database.

observe that although the result are better for clean speech, the noisy conditions did benefit from this optimization. Note that FA-based approaches performed better for the clean condition.[1] However, when the experiments were performed under noisy conditions, best results are obtained by the MVE-based approaches.

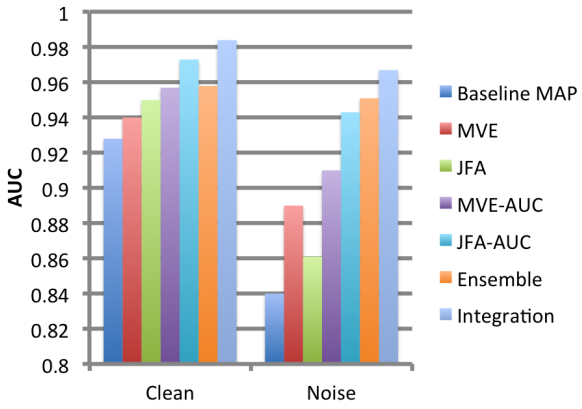


Figure 9.1: AUC of the ROC summary results for the MOBIO database.

Figure 9.2 describes the relative improvements. The relative improvement for the noisy conditions are higher than the ones observed for the on clean condition. The *integration* method for both clean and noisy conditions obtained the best results. Note that we are comparing the MVE-AUC with the MVE baseline and the rest of the approaches, JFA-AUC, *ensemble*, and *integration* with the JFA-baseline. The reason is to provide a fair comparison for the approaches that does not consider FA as an initial point.

[1]This observation was also true for the NIST database in previous Chapters.

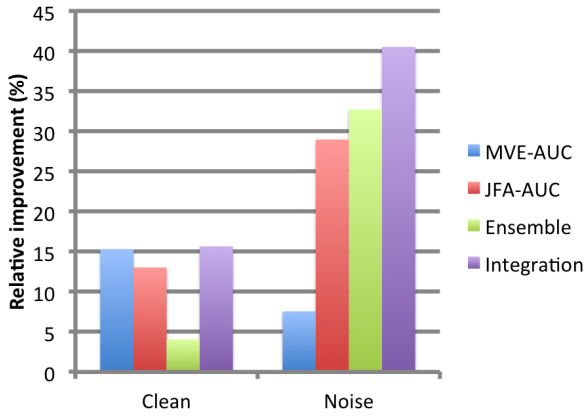


Figure 9.2: Relative improvement summary for MOBIO database.

9.2 Ahumada/Gaudi databases

The Ahumada/Gaudi data are other interesting databases in Spanish [144]. Both databases were recorded in Castilian Spanish under controlled conditions and include 200 male users for Ahumada and 200 female users for Gaudi. The signals were recorded over six sessions, three sessions for a microphone scenario (different microphones were used), and three sessions for telephone. The phrases recorded consisted of isolated digits, concatenated digits, balanced phrases, reading text, and spontaneous speech.

This database represents our first attempt to extend our knowledge beyond languages. For this database, we selected a subset of speakers to perform the experiments.

9.2.1 Experimental Setup

We followed the description in [145, 144], but we selected 100 users from each gender, from which imposters and target trials belong to matched and/or mismatched conditions. All the files were downsampled (if needed to 8 kHz). We focused mainly on channel mismatch in a text-independent context. The feature vectors include 49 components: 16 static Cepstral, 1 log Energy, 16 delta Ceptral coefficients, and 16 double delta Ceptral coefficients. A gender-dependent and target-independent 512-mixture GMM UBM model was used as a baseline. The initial GMM UBM was obtained from NIST databases: 2004, 2005, 2006, 2010 (microphone recordings). The target speakers were initially computed using MAP (all data available for the 200 users was used to compute the models). The JFA experiments employed 50 eigenchannels for the latent variable U and 100 for the eigenvoices V.

All the approaches that require a discriminative optimization (MVE, the optimization of AUC of the ROC curve, and the *ensemble*) used the usual selection of cohort samples: 500 phrases from each NIST 2004, 2005 2006, switchboard 1, switchboard 2 and 500 phrases from NIST 2010 microphone (3000 phrases were used as imposter samples). For *ensemble*

modeling, we limited to five partitions of the UBM data space. The partitioning was performed blindly, and the score outputs for the different scoring strategies were fused. For the *integration* approach, as for the *ensemble*, the region was blindly divided in to five partitions and a final fusion score was employed.

We used babble noise, extracted from the Aurora 2 database at different SNRS: 0,5,10,15 and 20 dB for the following experiments. The *ensemble* training used five partitions; the same is applied to the *integration* (for every SNR-region the new data space is partitioned into two new regions).

9.2.2 Results

Table 9.3 and 9.4 describe performance for the verification task using different approaches. Separate results for males and females were computed to analyze the current performance. For both cases we can observe that the results follow the same trend. Adaptation from a general UBM from another database caused the EERs and minDCF to be higher than usual. However, when optimizing the AUC of the ROC curve, including negative and positive examples of the current database, improved the results. Moreover, when applying *ensemble modeling* the results outperformed the MVE-AUC approach, but not JFA-AUC. Finally, the *integration* approach produced the best results. The *integration* approach includes just the fusion of scores.

Tables 9.3 and 9.4 show the same trend for the noise conditions. The discriminative optimizations improve better than the FA approaches in general. Once again, the best result was obtained using the *integration* method. The optimization of the AUC of the ROC curve improves the MVE and FA-related approaches.

	Clean		*Noise*	
	EER	*minDCF*	*EER*	*minDCF*
Baseline MAP	17.35	9.6	22.52	15.3
MVE	15.24	8.9	18.64	11.0
JFA	13.82	7.7	20.32	13.1
MVE-AUC	13.56	7.6	16.16	10.3
JFA-AUC	9.53	6.1	14.20	8.6
Ensemble Modeling	10.3	6.8	12.18	7.2
Integration	8.4	5.3	10.92	6.6

Table 9.3: Final results (EER and minDCF) for the Ahumada database.

Figure 9.3 shows the results for the AUC of the ROC curve as a metric. We observe that including the AUC optimization improves the current baselines. For clean case, the graph shows that FA-related techniques outperform approaches such as *ensemble modeling*.

However, for the noise conditions, *ensemble* and *integration modeling* improved the JFA and MVE methods.

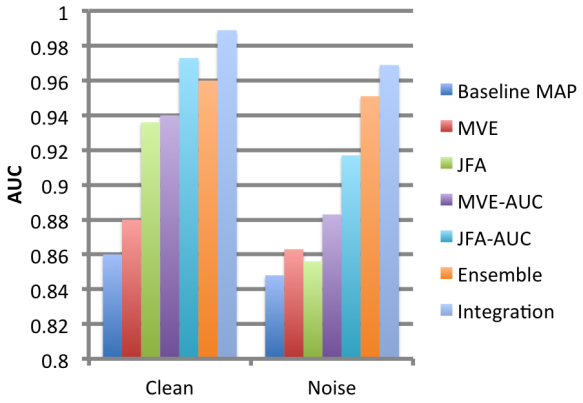


Figure 9.3: AUC of the ROC curve: summary results for the Ahumada database.

Figure 9.4 describes the relative improvements for Ahumada database. Note that MVE-AUC is compared against the MVE-baseline. Although the method improved the baseline, it did not showed any improvement against FA-related methods. Moreover, *ensemble modeling* did worse compared to JFA-AUC. *Ensemble* and *Integration* resulted in greater improvements for the noise conditions, following the same trend of previous databases.

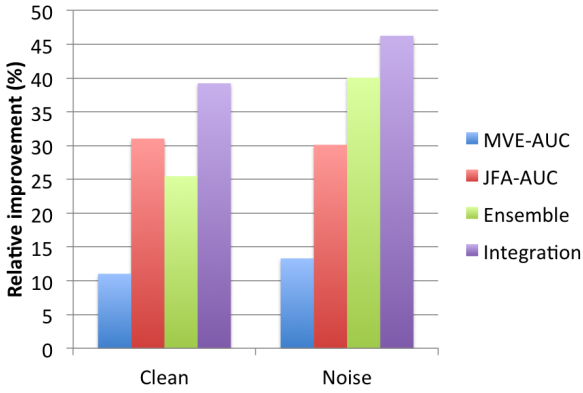


Figure 9.4: Relative improvement summary for the Ahumada database.

The results for the Gaudi (female) database are slightly better than the ones presented for Ahumada, but they follow the same tendency.

Figure 9.5 depicts the results of the AUC metric for this database. *Integration modeling* provides the greatest AUC values for both cases. For the clean case, the FA-related approaches obtained better results compared to the MVE approaches. However, under noise conditions, the MVE schemes provide better performance compared to FA-related optimizations.

Figure 9.6 shows the relative improvements for the clean and noise conditions. Once again, we compare MVE-AUC with the MVE baseline (the results if compared to the current

	Clean		Noise	
	EER	minDCF	EER	minDCF
Baseline MAP	16.5	9.3	21.26	14.7
MVE	14.4	7.9	17.45	10.6
JFA	12.4	6.6	19.1	12.4
MVE-AUC	12.6	6.4	15.12	9.1
JFA-AUC	8.3	5.2	13.31	7.2
Ensemble Modeling	9.2	6.0	11.08	6.6
Integration	7.8	4.7	9.71	5.4

Table 9.4: Final results (EER and minDCF) for the Gaudi database.

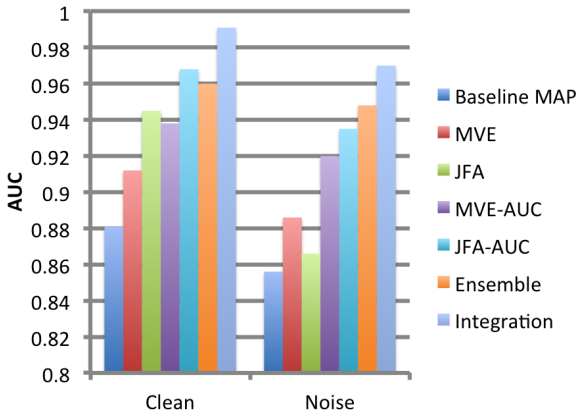


Figure 9.5: AUC of the ROC curve summary results for Gaudi database.

FA are slightly worse). For the clean condition, JFA-AUC and *integration modeling* provide the most noticeable improvement. Every proposed methodology provides a gain; moreover, we observe that the relative improvement is higher for the noise condition.

9.3 YOHO database

The YOHO database is one of the first data sets used for SV. It was recorded over telephone channels. Although the vocabulary seems quite small, (the dictionary consists of limited compound numbers), its simplicity make it easy to address issues like noise.

The database [49] has the following properties: it contains clean voice utterances of 138 speakers of different nationalities. It is a combination of lock phrases (for example, "Thirty-Two, Forty-One, Twenty-Five"), with 4 enrollment sessions per subject and 24 phrases per enrollment session; 10 verification sessions per subject and 4 phrases per verification session. Given 18768 sentences, 13248 sentences were used for training and 5520 sentences for testing.

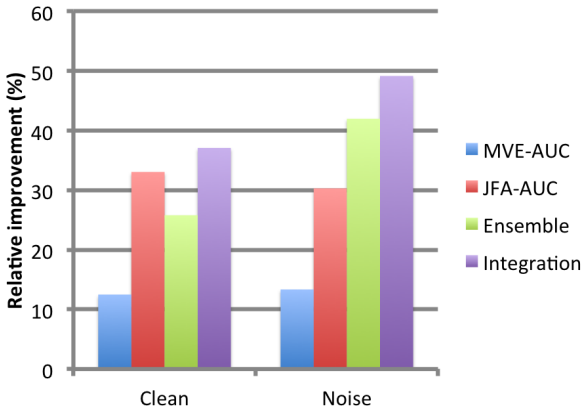


Figure 9.6: Relative improvement summary for Gaudi database.

9.3.1 Experimental Setup

We used the whole database (138 users) to obtain results based on our different approaches. We used only the current users of the database as target speakers and never developed imposter models. The basic UBM was generated using the NIST databases: 2004, 2005, 2006, 2010 (including the microphone). We tested the performance for a feature vector of 33 attributes: 16 static Cepstral coefficients, 1 log Energy, and 16 delta Cepstral coefficients (the same configuration as presented for MOBIO). Once again, a gender-dependent and target-independent 512-mixture GMM imposter model was used as a baseline. The initial GMM models were obtained from the NIST database plus the MOBIO data set used for training. For the test, we chose users from the database but also random users from other databases, such as NIST2004. JFA employed 50 eigenchannels for the latent variable U and 100 for eigenvoices V.

The discriminative approaches (MVE, the optimization of the AUC of the ROC curve, and the *ensemble*) use the usual selection of negative samples: 500 phrases from NIST 2004, 2005, 2006, switchboard 1, switchboard 2 and 500 phrases from NIST 2010 microphone (3000 imposter phrases). For the clean experiments of *ensemble modeling*, we limited the UBM space to five partitions. The partitions were blindly computed and the final outcome is the fusion of the scores. For the *integration* approach, the region was blindly divided into five partitions and a final fusion score is obtained.

For the noise-condition experiments we used babble noise extracted from the Aurora 2 database. It was added to the training and test files at different SNRS: 0, 5, 10, 15 and 20 dB. For the *ensemble* training we used a five-partitions scheme. The same is applied to the *integration*. For every SNR-dependent region the new data space is partitioned into two new regions.

9.3.2 Results

Table 9.5 shows the results for clean speech. The *integration* approach obtains the best results. Moreover, the system shows improvements for every approach. It is interesting that the *ensemble modeling* outperforms the optimized AUC for MVE, but not the optimized AUC for JFA. The JFA with the AUC optimization converges rapidly giving a good estimation of the parameters. Although the optimization of the ROC curve obtains a low misverification measure in the training stage, was not enough to compute an adequate model estimation.

Table 9.5 also shows the results for babble cocktail noise. They follow the same trend as the clean experiments. Note that even for small vocabularies, the proposed approaches lowered both the EER and the minDCF.

	Clean		*Noise*	
	EER	*minDCF*	*EER*	*minDCF*
Baseline MAP	3.29	2.1	7.24	5.8
MVE	2.93	2.1	6.53	5.2
JFA	1.54	1.1	4.39	3.9
MVE-AUC	1.64	1.2	5.58	4.8
JFA-AUC	0.81	0.5	3.23	2.1
Ensemble Modeling	1.33	0.9	3.51	2.3
Integration	0.62	0.4	2.62	1.6

Table 9.5: Final results (EER and minDCF) on the Test set for the YOHO database.

Figure 9.7 shows the AUC of the ROC curve metrics. Because of the small number of speakers and the high EER, results of the AUC are very small. We observe best results for the *integration* approach. The JFA and JFA-AUC methods performed better than MVE and MVE-AUC for the clean case (following the same trend as in previous databases). The AUC values are smaller for the noise conditions. All of them seem competitive, except that *integration modeling* clearly outperforms the rest.

In Figure 9.8 we show the relative improvement obtained using the proposed approaches. *Integration modeling* outperforms the rest of the approaches for clean and noise conditions. MVE-AUC is compared to MVE as the baseline, and JFA-AUC, *ensemble* and *integration* are compared to JFA as baseline. The *ensemble modeling* optimization shows the smallest improvements for this specific database.

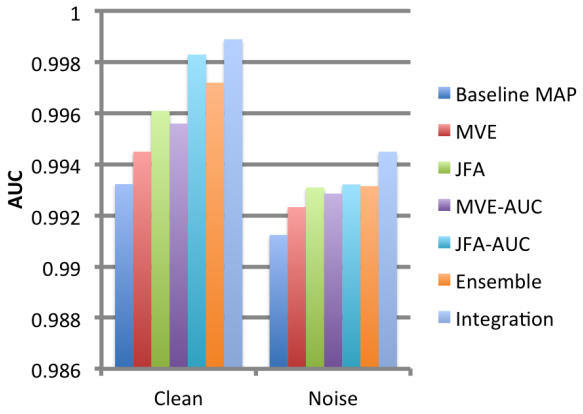


Figure 9.7: AUC of the ROC curve summary results for YOHO database.

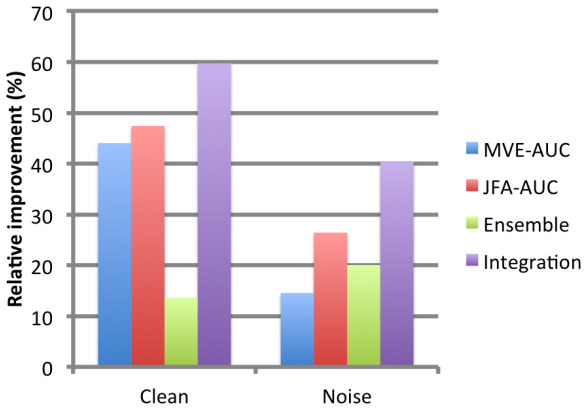


Figure 9.8: Relative improvement summary for YOHO database.

9.4 Summary

This chapter described the final set of results for the proposed optimization methods. We note that the optimization of the *area under the ROC curve* is beneficial in any case. Even for the state-of-the-art approaches, MVE and JFA, the AUC optimization reduces the EER and the minDCF. The *ensemble* approach was mainly focused to alleviate the channel mismatch problem and noise conditions. Nevertheless, it is shown that the ensemble approach is also beneficial for clean conditions, obtaining fair results. Moreover, the *integration* of all approaches shows a significant improvement, especially when fusion of the results is performed. The AUC of the ROC curve summary graphs show also significant results that support the contributions of the proposed approaches. Finally, the relative trends of improvements emphasize the potential of the techniques for new scenarios.

Chapter 10

Conclusions and Future Work

> Finally, from so little sleeping and so much reading, his brain dried up and he went completely out of his mind.
>
> - Miguel de Cervantes Saavedra *Don Quijote*

The theory and experiments in the previous chapters show the improvements obtained by optimizing the modeling of different methodologies using a *discriminative approach*. First, the AUC-based optimization is competitive with conventional training methods in the worst case (usually for a clean scenario) and can outperform them greatly under other circumstances (noise conditions). The first sign of this improvement is the decrease of the EER, minDCF results and the increase of the actual AUC of the ROC curve. Moreover, we support our initial hypothesis by providing examples where the AUC optimization actually improves the performance at all operating points. Presumably, if one were to consider the combined results at all operating points, the benefits of AUC-based learning would be even greater. Another lesson to be derived from our experiments is the direct optimization of an objective function that directly relates to the actual task being performed. This is not an epiphany – this fact has always been known; all we provide is some confirmation. Besides, if the only objective is the performance at a specific operating point (as specified by the ratio of FA and FR), then the AUC objective function could be modified to consider only that operating point. It is to be expected that the performance at that operating point will improve at the cost of performance at other operating points.

The AUC optimization with noise has provided another research branch, and the results are consistently significant. Clearly, a learning paradigm that optimizes the entire ROC curve results in better performance than that obtained with conventional maximum-likelihood of discriminative training methods. Is clearly seen not only in the ROC curve but also in the score distribution, the DET curve, and every metric. What is interesting, however, is that the improvements obtained from AUC optimization are actually

significantly greater for noisy speech than for clean speech. One reason is that AUC optimization naturally accounts for any shift in the data away from the intended operating point where performance is measured. More curiously, the performance of every method (JFA, MVE and i-vector), which function over the entire space of data, benefits significantly. It is important to point out the better performance of MVE in the noise conditions. Thus, the final improvement of AUC-optimized MVE on noisy data is superior to that obtained with similarly trained JFA. Moreover, the gains increase with increasing noise level.

We also presented an *ensemble* approach to partition the signal space into specific attributes. We found that partitioning and refining the space by speaker or by environment both improve performance over the baseline significantly. An interesting finding was the extent to which fusion and combination of the different scoring schemes outperform the baseline. Moreover, *partitioning* the space not only benefits a single methodology, but once again for both the MVE and FA approaches, the gain improves when the models become specific and refined. Even more interesting is the research under noise conditions. We found that having some prior knowledge of the noise (SNR in our case) improves the current results, and that performing a blind estimation of the attribute gives competitive results.

The *integration* of both algorithms was initially proposed for a robust system that can benefit from both approaches. However, the scheme can be applied to a broader range of possibilities, where not just noise is the most important factor. We also showed that establishing a hierarchical structure is useful, where each level acts only on a single problem improves the performance. The blind solution to obtain this structure is still a matter of research, but if some assumptions are made, like the type of noise and the amount of channels, the challenge becomes feasible. The main drawback of this approach is the amount of data needed in practice to optimize the models correctly. However, with the development of technology this is not longer a problem at the moment.

We extended our research not just to the NIST databases used to develop the system, but also to other databases that present their own challenges. We found that our proposed approaches improved the results consistently for each one of them (clean and noise condition), as in previous findings. Besides, it was interesting to consider databases in different languages, such as Ahumada and Gaudi and we obtained competitive results for them. Other small databases such as YOHO and MOBIO gave us confidence on the model refinements, but also supports the contention that the method is useful for more than just the NIST evaluations. The general models obtained from baseline databases can be adapted to new sets of data. Both of them showed improvements for every case.

The current state-of-the-art systems were outperformed by our proposed discriminative techniques. Although discriminative training is computationally expensive, we found that discarding the parts of the samples that do not provide valuable information to the models alleviates the problem.

Under controlled conditions, we showed that MVE and our discriminative optimization approach performed better than the ML approaches, specifically for systems that employ less than 512 GMM models. *FA*-based algorithms depend on the amount of data available to perform a suitable parameter estimation. Our approaches based on discriminative training mitigate this issue, when included as the last stage of the optimization procedure. The same occurs for noise conditions, where discriminative optimization given by either the *ensemble* or the AUC under the ROC curve, provide a better estimation of the noise models, specially for low SNRs.

10.1 Future Research

We presented different approaches of discriminative model optimization. Although each of them present improvements for every shown scenario, there are still challenges that may be addressed such as different noise types at different SNRs, real channel mismatches, training with just a few seconds of speech, as well as trials with just a few samples.

As noted earlier, *AUC-optimization* can also be employed for other learning and modeling techniques, *e.g.* i-vector based representations [14] and PLDA [121, 73]. Generalizations can also consider multi-class classification results such as those employed for speaker *identification*. These too are current areas of research. For the AUC optimization, it would be also interesting to select a region of interest of the curve, depending on the application, and optimize accordingly. For future research we can think of optimizing the DCF directly and calibrate according to this new scheme. As part of the future research, we will also investigate other objective functions that assigns weights to the ROC/DET curve so that we can control not only the area under the curve, but also the curve at each operating point.

For the *ensemble* approach, we found that the best results were obtained when *a priori* knowledge of the factor in the partitioning is available. We conjecture that much of this benefit may be obtained if this information is correctly *estimated* in the training stage. We will investigate this in future work. We also propose to investigate methods for identifying partitions when *multiple* factors must be considered concurrently, and particularly when they must be estimated. We will also investigate methods for formally optimizing the partitions for verification.

The *integration* approach was investigated for the particular cases of clean speech and babble noise. As for the other techniques, this approach can also be used for other type of noise types and channels. It is also interesting to find suitable techniques that can provide the appropriate hierarchical structure for each application.

Finally, our findings can also be extended and applied to other fields as speaker identification or language recognition.

Appendix A

Traditional approaches preliminaries

> Nothing is absolute. Everything changes, everything moves, everything revolves, everything flies and goes away.
>
> Frida Kahlo

This Appendix provide the readers with some details some of the classic algorithms used along the thesis.

A.1 Expectation Maximization

EM is a machine learning tool that works for point estimation. Then, for a set X_t where $t = 1...T$ and a hidden (latent) variable Z, we can estimate the parameters Λ. Let us consider define the following,

$$\begin{aligned} \ell(\Lambda) &= \log p(x|\Lambda) = \log \sum_z p(x, z|\Lambda) \\ &= \log \sum_z Q(z|x, \Lambda) \frac{p(x, z|\Lambda)}{Q(z|x, \Lambda)} \\ >= &\sum_z Q(z|x, \Lambda) \log \frac{p(x, z|\Lambda)}{Q(z|x, \Lambda)} \equiv F(Q, \Lambda), \end{aligned} \tag{A.1}$$

where $Q(z|x, \Lambda)$ is the density of z. Knowing the inequality, the task is to find the lower bound of $F(Q, \Lambda)$. Hence,

$$\mathrm{E} - \mathrm{step} \quad Q^{t+1} = \arg\max_Q p(Q, \Lambda^t) \tag{A.2}$$

$$\mathrm{M} - \mathrm{step} \quad \Lambda^{t+1} = \arg\max_\Lambda p(Q^{t+1}, \Lambda). \tag{A.3}$$

Consider $Q^{t+1} = p(z|x, \Lambda^t)$, then the distribution Q will just depend on Λ. Under this argument,

$$\ell(\Lambda) >= \sum_z Q(z|x,\Lambda) \log p(x,z|\Lambda) - \sum_z Q(z|x,\Lambda) \log Q(z|x,\Lambda) \quad (A.4)$$

$$>= Q(\Lambda|\Lambda^t) + S(Q). \quad (A.5)$$

Finally, we can obtain suitable expressions for the *expectation* and *maximization* steps:

$$\mathrm{E-step} \quad Q(\Lambda|\Lambda^t) = E(\log p(x,z|\Lambda) \quad (A.6)$$

$$\mathrm{M-step} \quad \Lambda^{t+1} = \arg\max_{\Lambda} E \log p(x,z|\Lambda). \quad (A.7)$$

A.2 Support Vector Machine

The *Support Vector Machine* (SVM) is widely used in pattern recognition. In this research, it is employed as a classifier that can transform feature data into a binary key. SVM was first developed by Vapnik and Chervonenkis [146]. Although it has been used for several applications, it has also been employed in biometrics [147].

Given the observation inputs and a function-based model, the goal of the basic SVM is to classify these inputs into one of two classes. Afterwards, the following set of pairs are defined $\{\rho_i, y_i\}$; where $\rho_i \in R^n$ are the training vectors and $y_i = \{-1, 1\}$ are the labels.

The method relies on a linear separation of the data previously mapped in a higher dimension space $\mathbb{H}$, by using $\phi : \mathbb{R}^n \to \mathbb{H}; \phi \to \phi(\rho)$. For generalization, let the margin between the separator hyperplane be,

$$\{\mathbf{h} \in \mathbb{H} | \langle \mathbf{w}, \mathbf{h} \rangle_{\mathbb{H}} + \varpi_0 = 0\}, \quad (A.8)$$

and the data $\phi(\rho)$ is maximized. Moreover, The SVM learning algorithm finds an hyperplane (w, b) such that,

$$\min_{\rho_i, b, \xi} \frac{1}{2} w^\top w + C \sum_{i=1}^{l} \xi_i \quad (A.9)$$

$$\text{subject to } y_i(w^\top \phi(\rho_i) + b) \geq 1 - \rho_i \quad (A.10)$$

$$\xi_i \geq 0,$$

where ξ_i is a slack variable and C is a positive real constant known as a tradeoff parameter between error and margin. Equations A.9 and A.10 can be transformed into a dual problem represented by the Lagrange multipliers α_i,

$$\sum_{i=1}^{l} \alpha_i - \frac{1}{2} \sum_{i=1,m=1}^{l} \alpha_i \alpha_m y_i y_m \langle \phi(\rho_i), \phi(\rho_m) \rangle \tag{A.11}$$

$$\sum_{i=1}^{l} \alpha_i y_i = 0, C \geq \alpha_i \geq 0. \tag{A.12}$$

α_i can be solved as a quadratic programming (QP) problem. The resulting values $\alpha_i \in \mathbb{R}$ have a close relation with the training points ρ_i.

To extend the linear method to a nonlinear technique, the input data is mapped into a higher dimensional space by function ϕ. However, exact specification of ϕ is not needed; instead, the expression known as kernel $K(\rho_i, \rho_j) \equiv \phi(\rho_i)^\top \phi(\rho_m)$ is defined.

There are different types of kernels: linear, polynomial, radial basis function (RBF) and sigmoid, among others. For our research, we just employed the linear kernel that is defined as,

$$K(\rho_i, \rho_j) \equiv \phi(\rho_i)^\top \phi(\rho_m), \tag{A.13}$$

which is equivalent to learn the weights discriminatively.

Appendix B

Review of Factor Analysis

> At the end of the day, we can endure much more than we think we can.
>
> Frida Kahlo

This Appendix provide the readers with some details about Factor Analysis that will be helpful to understand the state-of-the-art techniques.

B.1 Factor Analysis

Factor analysis establishes that continuous factors can control data. In other words, these factors can model the covariance structure of the high-dimensional data. So we can think of the data as being generated by first, locating a point in a subspace and then adding noise. If that is the case, we can consider the following:

$$X - \mu = LZ + \epsilon, \tag{B.1}$$

where X is a random variable of dimensions $P \times 1$ (observation), μ is the data mean vector, L refers to the loading matrix of $P \times R$ dimensions, Z is a latent random variable (known as a factor vector) of $R \times 1$ dimensions and ϵ is random noise with diagonal covariance matrix ψ.

The following assumptions are to be considered as well,

$$\mathbb{E}(X) = \mathbb{E}(Z) = \mathbb{E}(\epsilon) = 0 \tag{B.2}$$

$$\mathbb{E}(ZZ^{\top}) = \Omega = \mathbb{I} \tag{B.3}$$

$$\mathbb{E}(\epsilon\epsilon^\top) = \Psi = diag(\Psi) \tag{B.4}$$

$$\mathbb{E}(XZ^\top) = L \tag{B.5}$$

$$\Sigma = \mathbb{E}(XX^\top). \tag{B.6}$$

We consider that Z and ϵ are independent and that $X = LZ + \epsilon$ is a linear combination of common factors and Σ is full rank.

Moreover, we can show that,

$$\begin{aligned}
\mathbb{E}(XX^\top) &= \mathbb{E}[(LZ+\epsilon)(LZ+\epsilon)^\top] && \text{(B.7)}\\
&= \mathbb{E}[L\underbrace{ZZ^\top}_{\mathbb{I}}L^\top + L\underbrace{Z\epsilon^\top}_{0} + \underbrace{\epsilon Z^\top}_{0}L^T + \epsilon\epsilon^\top] && \text{(B.8)}\\
&= LL^\top + \Psi && \text{(B.9)}
\end{aligned}$$

and,

$$\begin{aligned}
\mathbb{E}(XF^\top) &= \mathbb{E}[(LZ+\epsilon)Z^\top] && \text{(B.10)}\\
&= \mathbb{E}(L\underbrace{ZZ^\top}_{\mathbb{I}} + \underbrace{\epsilon Z^\top}_{0})] && \text{(B.11)}\\
&= L && \text{(B.12)}
\end{aligned}$$

$LL^\top$ are called commonalities and ψ_i is the uniqueness.

$$\Sigma = LL^\top + \Psi = \begin{bmatrix} \ell_1\ell_1^\top + \psi_1 & \ell_1\ell_2^\top + \psi_1 & \cdots & \ell_1\ell_R^\top + \psi_1 \\ \ell_2\ell_1^\top + \psi_2 & \ell_2\ell_2^\top + \psi_2 & \cdots & \ell_2\ell_R^\top + \psi_2 \\ \vdots & \vdots & \ddots & \vdots \\ \ell_P\ell_1^\top + \psi_P & \ell_P\ell_2^\top + \psi_P & \cdots & \ell_P\ell_R^\top + \psi_P \end{bmatrix} \tag{B.13}$$

A special note: FA analysis exhibit several solutions:

$$\begin{aligned}
\mathbb{E}(LZ+\epsilon) &= \mathbb{E}[(L\overbrace{OO^\top}^{\mathbb{I}}Z+\epsilon)Z^\top] && \text{(B.14)}\\
&= \mathbb{E}[(\underbrace{LO}_{L^*}\underbrace{O^\top Z}_{Z^*}+\epsilon Z^\top] && \text{(B.15)}\\
&= \mathbb{E}(L^*Z^{*\top} + \epsilon Z^\top)] && \text{(B.16)}
\end{aligned}$$

Is not possible to distinguish between L^* and L.

B.2 Joint Factor Analysis computation in detail

The steps presented in the following sections are based on the work described in [60].

For the set feature vectors $\chi_1, \chi_2, ..., \chi_t, ...$ and speaker s, let

$$M = m + Vy + Ux + Dz. \tag{B.17}$$

1. Assuming that U and D are zero, train the loading matrix V,

$$M = m + Vy. \tag{B.18}$$

   (a) Compute the Baum-Welch Statisitcs 0-th, N, the 1-st, F, and 2-nd, $\mathfrak{S}$, order statistics of speaker s and Gaussian component c employing observations T:

$$N(s) = \sum_t \gamma_t(c) \tag{B.19}$$

$$F(s) = \sum_t \gamma_t(c)\chi_t \tag{B.20}$$

$$\mathfrak{S}(s) = \text{diag} \sum_t \gamma_t(c)\chi_t\chi_t^*, \tag{B.21}$$

   where γ is the posterior probability of component c of a GMM for a specific observation t and certain speaker s. Also note that the sum is computed over all conversation sides for speaker s.

   (b) Center the 1-st, F, and 2-nd, $\mathfrak{S}$, order statistics:

$$\tilde{F}(s) = F_c(s) - N(s)m_c \tag{B.22}$$

$$\tilde{\mathfrak{S}}(s) = \mathfrak{S}_c(s) - \text{diag}(F(s)m_c^* + m_cF(s)^* - N(s)m_cm_c^*). \tag{B.23}$$

   where m_c is the mean of a specific component in the GMM.

   (c) Expanding the matrices,

$$\mathbf{N} = \begin{bmatrix} N_1(s)\mathbf{I} & & \\ & \ddots & \\ & & N_c(s)\mathbf{I} \end{bmatrix} \tag{B.24}$$

$$\mathbf{F}(s) = \begin{bmatrix} \tilde{F}_1(s) \\ \vdots \\ \tilde{F}_c(s) \end{bmatrix} \tag{B.25}$$

$$\boldsymbol{\mathfrak{S}}(s) = \begin{bmatrix} \tilde{\mathfrak{S}}_1(s) & & \\ & \ddots & \\ & & \tilde{\mathfrak{S}}_c(s) \end{bmatrix} \tag{B.26}$$

(d) Using Baum-Welch statistics, estimate the posterior of the hidden factor y. First, let

$$l(s) = I + V^*\Sigma^{-1}\mathbf{N}(s)V, \tag{B.27}$$

where Σ is the covariance matrix of the UBM and V can use a random initialization. The posterior of $y(s)$ is given by

$$y(s) \sim \mathcal{N}(l^{-1}V\Sigma^{-1}\tilde{\mathbf{F}}(s), l_v^{-1}). \tag{B.28}$$

Hence, the expected value, $\mathbb{E}(y)$, is

$$\tilde{y}(s) = l^{-1}V\Sigma^{-1}\tilde{\mathbf{F}}(s), l_v^{-1}. \tag{B.29}$$

(e) Compute the statistics across speakers using information about the posterior distributions of the estimated y:

$$N_c = \sum_s N_c(s) \tag{B.30}$$

$$\mathcal{A} = \sum_s N_c(s) l_v^{-1}(s) \tag{B.31}$$

$$\mathcal{C} = \sum_s \mathbf{F}(s)(l^{-1}V\Sigma^{-1}\tilde{\mathbf{F}}(s))* \tag{B.32}$$

$$\mathbf{N} = \sum_s \mathbf{N}(s) \tag{B.33}$$

Note that $l_v^{-1}(s)$ is the posterior distribution of $y(s)$, and $\mathbb{E}[y^*(s)] = (l^{-1}V\Sigma^{-1}\tilde{\mathbf{F}}(s))^*$.

(f) Compute V and the update of the covariance.

$$\mathbf{V} = \begin{bmatrix} V_1 \\ \vdots \\ V_C \end{bmatrix} = \begin{bmatrix} \mathcal{A}_1\mathcal{C}_1 \\ \vdots \\ \mathcal{A}_1\mathcal{C}_C \end{bmatrix} \tag{B.34}$$

where

$$\mathbb{C} = \begin{bmatrix} \mathbb{C}_1 \\ \vdots \\ \mathbb{C}_C \end{bmatrix} \tag{B.35}$$

and

$$\Sigma = \mathbf{N}^{-1}\left(\sum_s \mathfrak{S}(s) - \mathrm{diag}(\mathcal{C}V^*)\right). \tag{B.36}$$

(g) Iterate from d to f between 15 to 25 times. In every iteration substitute the estimated V.

2. Assume D is zero, train the loading matrix U given the estimate of V.

$$M = m + Vy + Ux.$$

(a) After estimating y for each speaker, compute the Baum-Welch statistics, N_H and F_H, for each H (conversation side or channel) of each speaker S:

$$N_c(h,s) = \sum_{t \in h,s} \gamma_t(c) \tag{B.37}$$

$$F_c(h,s) = \sum_{t \in h,s} \gamma_t(c)\chi_t. \tag{B.38}$$

(b) Calculate the shift for each speaker using $\mathcal{R}(s) = m + Vy$ and compute the Gaussian posteriors. Subtract them from the first-order statistics F. The new $F_H(s)$ depends on the channel and the speaker,

$$\tilde{F}_c(h,s) = F_c(h,s) - \mathcal{R}(s)N_c(h,s). \tag{B.39}$$

(c) Expand the matrices:

$$\mathbf{N}(h,s) = \begin{bmatrix} N_1(h,s)\mathbf{I} & & \\ & \ddots & \\ & & N_c(h,s)\mathbf{I} \end{bmatrix} \tag{B.40}$$

$$\mathbf{F}(h,s) = \begin{bmatrix} \tilde{F}_1(h,s) \\ \vdots \\ \tilde{F}_c(h,s) \end{bmatrix}. \tag{B.41}$$

(d) Use the new statistics $\mathbf{N}(h,s)$ and $\mathbf{F}(h,s)$ to train U and x in the same way as the training of V and y (refer to step 4).

(e) Iterate 15 to 20 times to obtain U and x using the updated $\mathbf{N}(h,s)$ and $\mathbf{F}(h,s)$.

3. Train the residual D given the estimates of V and U,

$$M = m + Vy + Ux + Dz.$$

(a) Compute the speaker shift of $\mathcal{R}(s) = m + Vy_s$, and the channel shift, $\mathcal{W}(h,s) = Ux_{h,s}$. Compute the Gaussian posterior of the shifts and subtract them from the first order statistics, $F_{h,s}$,

$$\tilde{\mathbf{F}}_c(h,s) = F_c(h,s) - \mathcal{R}(s)N_c(s) - \sum_{h \in s} \mathcal{W}(h,s)N_c h, s. \tag{B.42}$$

(b) Expand the matrices:

$$\mathbf{N}(h,s) = \begin{bmatrix} N_1(h,s)\mathbf{I} & & \\ & \ddots & \\ & & N_c(h,s)\mathbf{I} \end{bmatrix} \tag{B.43}$$

$$\mathbf{F}(h,s) = \begin{bmatrix} \tilde{F}_1(h,s) \\ \vdots \\ \tilde{F}_c(h,s) \end{bmatrix}. \tag{B.44}$$

(c) Estimate the initial factor z and. Let D be a random,

$$l_D(s) = I + D^* \Sigma^{-1} \mathbf{N}(s) D. \tag{B.45}$$

The posterior of $z(s)$ is given by

$$z(s) \sim \mathcal{N}(l_D^{-1} D \Sigma^{-1} \mathbf{F}(s), l_D^{-1}). \tag{B.46}$$

Hence, the expected value, $\mathbb{E}(z)$, is

$$\tilde{z}(s) = l_D^{-1} D \Sigma^{-1} \mathbf{F}(s), l_D^{-1}. \tag{B.47}$$

(d) Update the statistics across the speakers:

$$N_c = \sum_s N_c(s) \tag{B.48}$$

$$\mathcal{A} = \sum_s N_c(s) l_D^{-1}(s) \tag{B.49}$$

$$\mathcal{B} = \sum_s \mathbf{F}(s)(l_D^{-1} D \Sigma^{-1} \tilde{\mathbf{F}}(s)) * \tag{B.50}$$

$$\mathbf{N} = \sum_s \mathbf{N}(s). \tag{B.51}$$

(e) Calculate D

$$\mathbf{D} = \begin{bmatrix} D_1 \\ \vdots \\ D_C \end{bmatrix} = \begin{bmatrix} \mathcal{A}_1\mathcal{B}_1 \\ \vdots \\ \mathcal{A}_1\mathcal{B}_C \end{bmatrix}, \tag{B.52}$$

where

$$\mathbb{C} = \begin{bmatrix} \mathbb{C}_1 \\ \vdots \\ \mathbb{C}_C \end{bmatrix}. \tag{B.53}$$

(f) Iterate from b to c 15 to 20 times.

Compute factors

- Compute the final factors x (channel), y (speaker) and the residual z.

Scoring

- Compute the final score by a simplified version of the log-likelihood ratio. Given a new utterance (referred here as tst), the loading matrices and the factors (referred here as tar), compute the product,

$$\theta_{s,H} = (Vy_{tar} + Dz_{tar})\Sigma^{-}1(F_{tst} - N_{tst}m - N_{tst}Ux_{tst}).$$

B.3 i-vectors

Train the Loading matrix

1. Train the loading matrix T,

$$M = m + Tw.$$

(a) As in JFA, compute the Baum-Welch statistics 0-th, N, the 1-st, F, and 2-nd, $\mathfrak{S}$, order statistics of speaker s and Gaussian component c. Collect the phrases belonging to specific channels and speaker, treat each channel-set collection as belonging to different speakers:

$$N(s) = \sum_t \gamma_t(c) \tag{B.54}$$

$$F(s) = \sum_t \gamma_t(c)\chi_t \tag{B.55}$$

$$\mathfrak{S}(s) = \text{diag} \sum_t \gamma_t(c)\chi_t\chi_t^*. \tag{B.56}$$

(b) Using Baum-Welch statistics, estimate the posterior of the hidden factor w.

(c) Compute statistics across speakers using the information of the posterior distributions of the estimated w.

(d) Compute T and the update of its covariance.

(e) Iterate from b to d between 15 to 25 times. In every iteration substitute the estimated T.

Channel compensation

Two approaches have provided successful results:

1. Compute the linear discriminant analysis (LDA) to reduce the dimensionality of the i-vectors.

   Afterwards, use Probabilistic Linear Discriminant Analysis (PLDA) as a the target trainer

2. Compute LDA of the i-vectors followed by with-In-class covariance normalization (WCCN).

Scoring

- Compute cosine distance scoring (CDS) on the updated i-vectors (after channel compensation).

Appendix C

Notes about MVE

> Most people believe the mind to be a mirror, more or less accurately reflecting the world outside them, not realizing on the contrary that the mind is itself the principal element of creation.
>
> Rabindranath Tagore

This Appendix provides the readers with some details about *Minimum Verification Approach* and its extensions.

C.1 Minimum Verification Error

To update the model parameters for a speaker S a GMM defined as, $\Lambda = \{w_k^C, \mu_k^C, \Sigma_k^C; \forall k\}$,where C is the class (target, S or impostor speaker, $\bar{S}$) and k is the GMM component.

Let us consider a dimension optimization, where $\mu_k^C = [\mu_{kz}^C]_{z=1}^D$, $\sigma_k^C = [\sigma_{kz}^C]_{z=1}^D$ and $\sum_{k=1}^M w_k^C = 1$.

Then, for μ_{kz}^C we perform the following,

$$\tilde{\mu}_{kz}^C = \frac{\mu_{kz}^C}{\sigma_{kz}^C} \tag{C.1}$$

For simplicity, we dropped the subindex z considering the optimization of just one dimension (keep in mind that the optimization is performed dimension by dimension),

$$\tilde{\mu}_k^C(t+1) = \tilde{\mu}_k^C(t) - \epsilon \frac{\partial \ell_C(\chi;\Lambda)}{\partial \tilde{\mu}_k^C}|_{\Lambda=\Lambda_t}, \tag{C.2}$$

where

$$\frac{\partial \ell_C(\chi;\Lambda)}{\partial \tilde{\mu}_k^C} = \frac{\partial \ell_C(\chi;\Lambda)}{\partial d_C}\frac{\partial d_C}{\partial \tilde{\mu}_k^C}, \tag{C.3}$$

$$\frac{\partial \ell_C(\chi;\Lambda)}{\partial d_C} = \gamma \ell_C(d_C)(1-\ell_C(d_C)), \tag{C.4}$$

$$\frac{\partial d_C}{\partial \tilde{\mu}_k^C} = -\frac{\partial g_n(\chi;\Lambda)}{\partial \tilde{\mu}_k^C} + \frac{\partial G_n(\chi;\Lambda)}{\partial \tilde{\mu}_k^C} \;\; \chi \in C_n \tag{C.5}$$

$$\frac{\partial g_C(\chi;\Lambda)}{\partial \mu_k^C} = \frac{1}{p(\chi;\Lambda_C)}\frac{\partial\, p(\chi;\Lambda_C)}{\partial \mu_k^C}, \tag{C.6}$$

$$\frac{\partial G_C(\chi;\Lambda)}{\partial \mu_k^C} = \frac{1}{p(\chi;\bar{\Lambda}_C)}\frac{\partial\, p(\chi;\bar{\Lambda}_C)}{\partial \mu_k^C}. \tag{C.7}$$

Consider for a specific k, dimension d, the speaker C and to avoid bias, let $\mu_k^C = \sigma\,\tilde{\mu_k^C}$. To compute the mean,

$$\frac{\partial p(\chi;\Lambda)}{\partial \tilde{\mu}_k} = \frac{w_k\,\mathcal{N}(\chi|\tilde{\mu}_k,\sigma_k)}{\sum_{k'} w_{k'}\,\mathcal{N}(\chi|\tilde{\mu}_{k'},\sigma_{k'})}\left(\frac{\chi}{\sigma_k} - \tilde{\mu}_k\right). \tag{C.8}$$

The computation for Σ_k^C and w_k^C adopts the same procedure.

Let $\tilde{\sigma}_k^C = \log(\sigma_k^C)$,

$$\tilde{\sigma}_k^C(t+1) = \tilde{\sigma}_k^C(t) - \epsilon\frac{\partial \ell_C(\chi;\Lambda)}{\partial \tilde{\sigma}_k^C}|_{\Lambda=\Lambda_t}, \tag{C.9}$$

where

$$\frac{\partial \ell_C(\chi;\Lambda)}{\partial \tilde{\sigma}_k^C} = \frac{\partial \ell_C(\chi;\Lambda)}{\partial d_C}\frac{\partial d_C}{\partial \tilde{\sigma}_k^C}, \tag{C.10}$$

$$\frac{\partial \ell_C(\chi;\Lambda)}{\partial d_C} = \gamma \ell_C(d_C)(1-\ell_C(d_C)), \tag{C.11}$$

$$\frac{\partial d_C}{\partial \tilde{\sigma}_k^C} = -\frac{\partial g_n(\chi;\Lambda)}{\partial \tilde{\sigma}_k^C} + \frac{\partial G_n(\chi;\Lambda)}{\partial \tilde{\sigma}_k^C} \;\; \chi \in C_n \tag{C.12}$$

$$\frac{\partial g_C(\chi;\Lambda)}{\partial \tilde{\sigma}_k^C} = \frac{1}{p(\chi;\Lambda_C)}\frac{\partial\, p(\chi;\Lambda_C)}{\partial \tilde{\sigma}_k^C}, \tag{C.13}$$

$$\frac{\partial G_C(\chi;\Lambda)}{\partial \tilde{\sigma}_k^C} = \frac{1}{p(\chi;\bar{\Lambda}_C)}\frac{\partial\, p(\chi;\bar{\Lambda}_C)}{\partial \tilde{\sigma}_k^C}. \tag{C.14}$$

The final solution to compute σ is

$$\frac{\partial p(\chi;\Lambda)}{\partial \sigma_k} = \frac{w_k\,\mathcal{N}(\chi|\mu_k,\sigma_k)}{\sum_{k'} w_{k'}\,\mathcal{N}(\chi|\mu_{k'},\sigma_{k'})}\left(\frac{\chi-\mu_k}{\tilde{\sigma}_k} - 1\right). \tag{C.15}$$

For the weights, w, we proceed as follows,

$$\tilde{w}_k^C(t+1) = \tilde{w}_k^C(t) - \epsilon \frac{\partial \ell_C(\chi;\Lambda)}{\partial \tilde{w}_k^C}|_{\Lambda=\Lambda_t}, \tag{C.16}$$

where

$$\frac{\partial \ell_C(\chi;\Lambda)}{\partial \tilde{w}_k^C} = \frac{\partial \ell_C(\chi;\Lambda)}{\partial d_C}\frac{\partial d_C}{\partial \tilde{w}_k^C}, \tag{C.17}$$

$$\frac{\partial \ell_C(\chi;\Lambda)}{\partial d_C} = \gamma \ell_C(d_C)(1-\ell_C(d_C)), \tag{C.18}$$

$$\frac{\partial d_C}{\partial \tilde{w}_k^C} = -\frac{\partial g_n(\chi;\Lambda)}{\partial \tilde{w}_k^C} + \frac{\partial G_n(\chi;\Lambda)}{\partial \tilde{w}_k^C} \;\; \chi \in C_n \tag{C.19}$$

$$\frac{\partial g_C(\chi;\Lambda)}{\partial \tilde{w}_k^C} = \frac{1}{p(\chi;\Lambda_C)}\frac{\partial\, p(\chi;\Lambda_C)}{\partial \tilde{w}_k^C}, \tag{C.20}$$

$$\frac{\partial G_C(\chi;\Lambda)}{\partial \tilde{w}_k^C} = \frac{1}{p(\chi;\bar{\Lambda}_C)}\frac{\partial\, p(\chi;\bar{\Lambda}_C)}{\partial \tilde{w}_k^C}. \tag{C.21}$$

Finally, for the set of weights w_k and to ensure that $\sum_{k=1}^K w_k = 1$, let, $w = \frac{\exp(\tilde{w})}{\sum_{k=1}^K \exp(\tilde{w})}$.

$$\frac{\partial p(\chi;\Lambda)}{\partial \tilde{w}_k} = \frac{\mathcal{N}(\chi|\mu_k,\sigma_k)}{\sum_{k'} w_{k'}\,\mathcal{N}(\chi|\mu_{k'},\sigma_{k'})}. \tag{C.22}$$

C.2 AUC optimization using FA

The optimization of the parameters for FA has two stages (as in the usual JFA) to decouple the estimation of V, U, Ψ.

1. Start by V maintaining U fixed, considering that $Q = VV^\top + \Psi$, where $\Psi = DD^\top$ and is diagonal. The computation of μ, for a specific k, dimension d and to avoid bias, is straightforward using, $\mu_k = q_k\,\tilde{\mu}_k$; hence,

$$\nabla_\phi \Upsilon(\chi,\tilde{\mu}_k) = -\sum_{S\in\mathcal{S}} \frac{\sum_{\chi\in\mathbf{X}_S}\sum_{\chi\in\mathbf{X}_{\bar{S}}}\gamma R(1-R)}{|\mathbf{X}_S||\mathbf{X}_{\bar{S}}|}\frac{w_k\,\mathcal{N}(\chi|\mu_k,q_k)}{\sum_{k'} w_{k'}\,\mathcal{N}(\chi|\mu_{k'},q_{k'})}\left(\frac{\chi}{q_k}-\tilde{\mu}_k\right). \tag{C.23}$$

2. To estimate Q, we formulate first for V and then Ψ.

   We use the following identities to solve for Q. Let,

$$\frac{\partial(\log|q_k|)}{\partial q_k^{i,j}} = (q_{i.j}^k)^{-1} \tag{C.24}$$

$$\frac{\partial (q_{o.p}^k)^{-1}}{\partial q_k^{i,j}} = (q_{o.i}^k)^{-1}(q_{p.j}^k)^{-1} \tag{C.25}$$

We will also make the matrix dimensions explicit, assuming that Ψ is diagonal. Then, q_k, let $\tilde{q} = \log(q)$.

Then, for V,

$$\nabla_\phi \Upsilon(\chi, \tilde{v}_{ij}) = -\sum_{S \in \mathcal{S}} \frac{\sum_{\chi \in \mathbf{X}_S} \sum_{\chi \in \mathbf{X}_{\bar{S}}} \gamma R(1-R)}{|\mathbf{X}_S||\mathbf{X}_{\bar{S}}|} \tag{C.26}$$

$$\frac{w_k \mathcal{N}(\chi|\mu_k, q_k)}{\sum_{k'} w_{k'} \mathcal{N}(\chi|\mu_{k'}, q_{k'})} \left\{ [Q^{-1}(x-u)]_i [(x-u)^\top Q^{-1} V]_j - [Q^{-1}V]_{ij} \right\}. \tag{C.27}$$

3. for ψ,

$$\nabla_\phi \Upsilon(\chi, \tilde{\psi}_{i,j}) = -\sum_{S \in \mathcal{S}} \frac{\sum_{\chi \in \mathbf{X}_S} \sum_{\chi \in \mathbf{X}_{\bar{S}}} \gamma R(1-R)}{|\mathbf{X}_S||\mathbf{X}_{\bar{S}}|} \frac{w_k \mathcal{N}(\chi|\mu_k, q_k)}{\sum_{k'} w_{k'} \mathcal{N}(\chi|\mu_{k'}, q_{k'})} \left\{ \frac{1}{2} \left(Q_{ii}^{-1} - [Q^{-1}x - \mu]_i^2 \right) \right\}. \tag{C.28}$$

Now, for U, we fix the current Q and estimate accordingly.

$$\nabla_\phi \Upsilon(\chi, \tilde{U}_k) = -\sum_{S \in \mathcal{S}} \frac{\sum_{\chi \in \mathbf{X}_S} \sum_{\chi \in \mathbf{X}_{\bar{S}}} \gamma R(1-R)}{|\mathbf{X}_S||\mathbf{X}_{\bar{S}}|} \frac{w_k \mathcal{N}(\chi|\mu_k, q_k)}{\sum_{k'} w_{k'} \mathcal{N}(\chi|\mu_{k'}, q_{k'})} \left\{ \left(\frac{\chi - \mu_k}{\tilde{U}_k} \right)^2 - 1 \right\}. \tag{C.29}$$

Finally, for the set of weights w_k and to ensure that $\sum_{k=1}^K w_k = 1$, let $w = \frac{\exp(\tilde{w})}{\sum_{k=1}^K \exp(\tilde{w})}$,

$$\nabla_\phi \Upsilon(\chi, \tilde{w}_k) = -\sum_{S \in \mathcal{S}} \frac{\sum_{\chi \in \mathbf{X}_S} \sum_{\chi \in \mathbf{X}_{\bar{S}}} \gamma R(1-R)}{|\mathbf{X}_S||\mathbf{X}_{\bar{S}}|} \frac{\tilde{w}_k \mathcal{N}(\chi|\mu_k, q_k)}{\sum_{k'} w_{k'} \mathcal{N}(\chi|\mu_{k'}, q_{k'})}. \tag{C.30}$$

Bibliography

[1] Y. Lei, L. Burget, and N. Scheffer, "A noise robust i-vector extractor using vector taylor series for speaker recognition," in *Acoustics, Speech and Signal Processing (ICASSP), 2013 IEEE International Conference on.* IEEE, 2013, pp. 6788–6791.

[2] T. Hasan and J. Hansen, "Integrated feature normalization and enhancement for robust speaker recognition using acoustic factor analysis," in *Proc. Interspeech*, 2012, pp. 1568–1571.

[3] C. Hanilçi, T. Kinnunen, R. Saeidi, J. Pohjalainen, P. Alku, F. Ertas, J. Sandberg, and M. Hansson-Sandsten, "Comparing spectrum estimators in speaker verification under additive noise degradation," in *Acoustics, Speech and Signal Processing (ICASSP), 2012 IEEE International Conference on.* IEEE, 2012, pp. 4769–4772.

[4] T. Weigold, T. Kramp, and M. Baentsch, "Remote Client Authentication," *IEEE Security & Privacy*, vol. 6, no. 4, pp. 36–43, 2008.

[5] L. O'Gorman, "Comparing passwords, tokens, and biometrics for user authentication," *Proceedings of the IEEE*, vol. 91, no. 12, pp. 2021–2040, 2003.

[6] A. Jain, A. Ross, and S. Pankanti, "Biometrics: a tool for information security," *IEEE transactions on information forensics and security*, vol. 1, no. 2, pp. 125–143, 2006.

[7] J. P. Campbell, W. Shen, W. M. Campbell, R. Schwartz, J.-F. Bonastre, and D. Matrouf, "Forensic speaker recognition," *Signal processing magazine, IEEE*, vol. 26, no. 2, pp. 95–103, 2009.

[8] P. J. Barger and S. Sridharan, "On the performance and use of speaker recognition systems for surveillance," in *Video and Signal Based Surveillance, 2006. AVSS'06. IEEE International Conference on.* IEEE, 2006, pp. 109–109.

[9] T. Herbig, F. Gerl, and W. Minker, "Simultaneous speech recognition and speaker identification," in *Spoken Language Technology Workshop (SLT), 2010 IEEE.* IEEE, 2010, pp. 218–222.

[10] Q. Li, B. Juang, C. Lee, Q. Zhou, and F. Soong, "Recent advancements in automatic speaker authentication," *IEEE Robotics & Automation Magazine*, vol. 6, no. 1, pp. 24–34, 1999.

[11] J. Campbell *et al.*, "Speaker recognition: A tutorial," *Proceedings of the IEEE*, vol. 85, no. 9, pp. 1437–1462, 1997.

[12] D. Petrovska-Delacrétaz, A. El Hannani, and G. Chollet, "Text-independent speaker verification: state of the art and challenges," *Progress in nonlinear speech processing*, pp. 135–169, 2007.

[13] F. Bimbot, J.-F. Bonastre, C. Fredouille, G. Gravier, I. Magrin-Chagnolleau, S. Meignier, T. Merlin, J. Ortega-García, D. Petrovska-Delacrétaz, and D. A. Reynolds, "A tutorial on text-independent speaker verification," *EURASIP journal on applied signal processing*, vol. 2004, pp. 430–451, 2004.

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

[15] P. Kenny, G. Boulianne, P. Ouellet, and P. Dumouchel, "Joint factor analysis versus eigenchannels in speaker recognition," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 15, no. 4, pp. 1435–1447, 2007.

[16] W. Chou, "Discriminant-function-based minimum recognition error rate pattern-recognition approach to speech recognition," *Proceedings of the IEEE*, vol. 88, no. 8, pp. 1201–1223, 1992.

[17] B.-H. Juang and S. Katagiri, "Discriminative learning for minimum error classification," *Signal Processing, IEEE Transactions on*, vol. 40, no. 12, pp. 3043–3054, 1992.

[18] D. Povey, P. Woodland, and M. Gales, "Discriminative MAP for acoustic model adaptation," in *IEEE Intl. Conf. on Acoustics, Speech, and Sig. Proc. (ICASSP)*, vol. 1, 2003, pp. I–312.

[19] D. Zhu, H. Li, B. Ma, and C. Lee, "Discriminative learning for optimizing detection performance in spoken language recognition," in *Acoustics, Speech and Signal Processing, 2008. ICASSP 2008. IEEE International Conference on.* IEEE, 2008, pp. 4161–4164.

[20] C. Ma and E. Chang, "Comparison of discriminative training methods for speaker verification," in *IEEE Intl. Conference on Acoustics, Speech, and Sig. Proc.(ICASSP)*, vol. 1, 2003, pp. 192–195.

[21] M. G. Rahim, C. H. Lee, B. H. Juang, and W. Chou, "Discriminative utterance verification using minimum string verification error (MSVE) training," *Proc. ICASSP*, vol. 6, pp. 3585–3588, 1996.

[22] L. Burget, O. Plchot, S. Cumani, O. Glembek, P. Matejka, and N. Brummer, "Discriminatively trained probabilistic linear discriminant analysis for speaker verification," in *IEEE Intl. Conf. on Acoustics, Speech and Sig. Proc. (ICASSP)*, 2011, pp. 4832–4835.

[23] S. Cumani, N. Brummer, L. Burget, and P. Laface, "Fast discriminative speaker verification in the i-vector space," in *Acoustics, Speech and Signal Processing (ICASSP), 2011 IEEE International Conference on.* IEEE, 2011, pp. 4852–4855.

[24] B.-H. Juang, W. Chou, and C.-H. Lee, "Minimum classification error rate methods for speech recognition," *IEEE Trans. on Speech and Audio Processing*, vol. 5, pp. 257–265, May 1997.

[25] H. B. Mann and D. R. Whitney, "On a test of whether one of two random variables is stochastically larger than the other," *Annals of Mathematical Statistics*, vol. 18:1, pp. 50–60, 1947.

[26] C. Cortes and M. Mohri, "AUC optimization vs. error rate minimization," *Advances in neural information processing systems*, vol. 16, no. 16, pp. 313–320, 2004.

[27] L. Burget, M. Fapso, V. Hubeika, O. Glembek, M. Karafiát, M. Kockmann, P. Matejka, P. Schwarz, and J. Cernockỳ, "BUT system for NIST 2008 speaker recognition evaluation." in *INTERSPEECH*, 2009, pp. 2335–2338.

[28] M. A. H. Huijbregts, "Segmentation, diarization and speech transcription: surprise data unraveled," Ph.D. dissertation, Centre for Telematics and Information Technology University of Twente, 2009.

[29] S. 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, p. 357, 1980.

[30] B. Bogert, M. Healy, and J. Tukey, "The quefrency alanysis of time series for echoes: Cepstrum, pseudo-autocovariance, cross-cepstrum and saphe cracking," in *Proc. Symp. on Time Series Analysis*, 1963, pp. 209–243.

[31] M. Kockmann, L. Ferrer, L. Burget, E. Shriberg, and J. Cernocky, "Recent progress in prosodic speaker verification," in *Acoustics, Speech and Signal Processing (ICASSP), 2011 IEEE International Conference on.* IEEE, 2011, pp. 4556–4559.

[32] L. Ferrer, N. Scheffer, and E. Shriberg, "A comparison of approaches for modeling prosodic features in speaker recognition," in *Acoustics Speech and Signal Processing (ICASSP), 2010 IEEE International Conference on.* IEEE, 2010, pp. 4414–4417.

[33] W. Campbell, J. Campbell, D. Reynolds, D. Jones, and T. Leek, "High-level speaker verification with support vector machines," in *Acoustics, Speech, and Signal Processing, 2004. Proceedings.(ICASSP'04). IEEE International Conference on*, vol. 1. IEEE, 2004, pp. I–73.

[34] T. Kinnunen, "Spectral features for automatic text-independent speaker recognition," Licenciate's Thesis, University of Joensuu, 2003.

[35] L. R. Rabiner and R. W. Schafer, *Digital processing of speech signals.* IET, 1979, vol. 19.

[36] D. A. Reynolds, "Experimental evaluation of features for robust speaker identification," *Speech and Audio Processing, IEEE Transactions on*, vol. 2, no. 4, pp. 639–643, 1994.

[37] J. I. Makhoul, "Linear prediction: A tutorial review," *Proceedings of the IEEE*, vol. 63, pp. 561–580, Apr. 1975.

[38] H. Hermansky, "Perceptual linear predictive (PLP) analysis of speech," *Journal of the Acoustical Society of America*, vol. 87, no. 4, pp. 1738–1752, Apr. 1990.

[39] A. F. Martin and C. S. Greenberg, "NIST 2008 speaker recognition evaluation: performance across telephone and room microphone channels." in *INTERSPEECH*, 2009, pp. 2579–2582.

[40] S. Furui, "Cepstral analysis techniques for automatic speaker verification," *IEEE Trans. on Acoustics, Speech, and Signal Processing*, vol. 27, pp. 254–272, Apr. 1979.

[41] O. Viikki and K. Laurila, "Cepstral domain segmental feature vector normalization for noise robust speech recognition," *Speech Communication*, vol. 25, no. 1-3, pp. 133–147, 1998.

[42] H. Hermansky, N. Morgan, A. Bayya, and P. Kohn, "RASTA-PLP speech analysis technique," in *Proc. ICASSP*, San Francisco, CA, Mar. 1992, pp. I:121–124.

[43] J. Pelecanos and S. Sridharan, "Feature warping for robust speaker verification," in *2001: A Speaker Odyssey-The Speaker Recognition Workshop*, 2001, pp. 213–218.

[44] S. Chen and R. Gopinath, "Gaussianization," *Advances in neural information processing systems*, pp. 423–429, 2001.

[45] C. Myers, L. Rabiner, and A. Rosenberg, "Performance tradeoffs in dynamic time warping algorithms for isolated word recognition," *Acoustics, Speech and Signal Processing, IEEE Transactions on*, vol. 28, no. 6, pp. 623–635, 1980.

[46] F. Soong, A. Rosenberg, L. Rabiner, and B. Juang, "A vector quantization approach to speaker recognition," in *Acoustics, Speech, and Signal Processing, IEEE International Conference on ICASSP'85.*, vol. 10. IEEE, 1985, pp. 387–390.

[47] J.-M. Marin, K. Mengersen, and C. P. Robert, "Bayesian modelling and inference on mixtures of distributions," *Handbook of statistics*, vol. 25, pp. 459–507, 2005.

[48] D. Reynolds, "A Gaussian mixture modeling approach to text-independent speaker identification," Ph.D. dissertation, Georgia Institute of Technology, 1992.

[49] J. P. Campbell Jr, "Testing with the YOHO cd-rom voice verification corpus," in *Acoustics, Speech, and Signal Processing, 1995. ICASSP-95., 1995 International Conference on*, vol. 1. IEEE, 1995, pp. 341–344.

[50] R. O. Duda, P. E. Hart *et al.*, *Pattern classification and scene analysis.* Wiley New York, 1973, vol. 3.

[51] A. Wald, *Sequential analysis.* New York, NY: John Wiley and Sons, 1947.

[52] E. Lehmann and J. Romano, *Testing statistical hypotheses.* Springer Verlag, 2005.

[53] C. M. Bishop and N. M. Nasrabadi, *Pattern recognition and machine learning.* Springer, New York, 2006, vol. 1.

[54] L. Wasserman, *All of statistics: a concise course in statistical inference.* Springer, 2004.

[55] Y. Zigel and A. Cohen, "On cohort selection for speaker verification." in *INTERSPEECH*, 2003, pp. 2977–2980.

[56] T. Kinnunen, E. Karpov, and P. Fränti, "Efficient online cohort selection method for speaker verification." in *INTERSPEECH*, 2004, pp. 2401–2404.

[57] L. Burget, O. Plchot, S. Cumani, O. Glembek, P. Matejka, and N. Brummer, "Discriminatively trained probabilistic linear discriminant analysis for speaker verification," in *Acoustics, Speech and Signal Processing (ICASSP), 2011 IEEE International Conference on.* IEEE, 2011, pp. 4832–4835.

[58] W.-Q. Zhang, Y. Shan, and J. Liu, "Multiple background models for speaker verification," in *Proc. of Odyssey speaker and language recognition workshop*, 2010, pp. 47–51.

[59] A. Sarkar and S. Umesh, "Multiple background models for speaker verification using the concept of vocal tract length and MLLR super-vector," *International Journal of Speech Technology*, vol. 15, no. 3, pp. 351–364, 2012.

[60] P. Kenny, P. Oueleet, N. Dehak, V. Gupta, and P. Dumouchel, "A study of inter-speaker variability in speaker verification," *IEEE Trans. ASLP*, vol. 16, pp. 980–988, 2008.

[61] T. K. Moon, "The expectation-maximization algorithm," *Signal processing magazine, IEEE*, vol. 13, no. 6, pp. 47–60, 1996.

[62] D. A. Reynolds, T. F. Quatieri, and R. B. Dunn, "Speaker verification using adapted Gaussian mixture models," *Digital Signal Processing*, vol. 10, pp. 19–41, 2000.

[63] D. A. Reynolds, "Speaker identification and verification using Gaussian mixture speaker models," *Speech Communication*, vol. 17, no. 1-2, pp. 91–108, 1995.

[64] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the EM algorithm," *Annals of the Royal Statistical Society*, vol. 39, pp. 1–38, Dec. 1977.

[65] J.-L. Gauvain and C.-H. Lee, "Maximum a posteriori estimation for multivariate Gaussian mixture observations of markov chains," *IEEE Trans. on Speech and Audio Processing*, vol. 2, pp. 291–299, Apr. 1994.

[66] Y. Normandin and S. D. Morgera, "An improved MMIE training algorithm for speaker-independent, small vocabulary, continuous speech recognition," in *Acoustics, Speech, and Signal Processing, 1991. ICASSP-91., 1991 International Conference on.* IEEE, 1991, pp. 537–540.

[67] Y. Normandin, "Maximum mutual information estimation of hidden markov models," in *Automatic Speech and Speaker Recognition.* Springer, 1996, pp. 57–81.

[68] V. Vapnik, *The nature of statistical learning theory.* Springer Verlag, 2000.

[69] B. Schölkopf and A. Smola, *Learning with kernels: Support vector machines, regularization, optimization, and beyond*. The MIT Press, 2002.

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

[71] J. Mariéthoz and S. Bengio, "A unified framework for score normalization techniques applied to text-independent speaker verification," *IEEE Signal Processing Letters*, vol. 12, no. 7, pp. 532–535, 2005.

[72] J. Navratil and G. Ramaswamy, "The awe and mystery of t-norm," in *Eighth European Conference on Speech Communication and Technology*, 2003, pp. 2009–2012.

[73] P. Kenny, "Bayesian speaker verification with heavy-tailed priors," *keynote presentation, Odyssey Speaker and Language Recognition Workshop Brno, Czech Republic*, 2010.

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

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

[76] D. A. van Leeuwen and N. Brümmer, "An introduction to application-independent evaluation of speaker recognition systems," in *Speaker Classification I*. Springer, 2007, pp. 330–353.

[77] J. G. Fiscus and G. R. Doddington, "Topic detection and tracking evaluation overview," in *Topic detection and tracking*. Springer, 2002, pp. 17–31.

[78] L. Ferrer, "Statistical modeling of heterogeneous features for speech processing tasks," Ph.D. dissertation, Stanford University, 2009.

[79] G. McLachlan and T. Krishnan, *The EM algorithm and extensions*. John Wiley & Sons, 2007, vol. 382.

[80] R. M. Neal and G. E. Hinton, "A view of the EM algorithm that justifies incremental, sparse, and other variants," in *Learning in graphical models*. Springer, 1998, pp. 355–368.

[81] R. Lippmann, "An introduction to computing with neural nets," *ASSP Magazine, IEEE*, vol. 4, no. 2, pp. 4–22, 1987.

[82] T. Kohonen, G. Barna, and R. Chrisley, "Statistical pattern recognition with neural networks: Benchmarking studies," in *Neural Networks, 1988., IEEE International Conference on*. IEEE, 1988, pp. 61–68.

[83] A. E. Rosenberg, O. Siohan, and S. Parthasarathy, "Speaker verification using minimum verification error training," *Proc. ICASSP*, pp. 105–108, 1998.

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

[85] L. R. Rabiner and B. H. Juang, *Fundamentals of Speech Recognition*. Englewood Cliffs, NJ: Prentice-Hall, 1993.

[86] C. Lee, "A unified statistical hypothesis testing approach to speaker verification and verbal information verification," in *Proc. COST,Workshop on Speech Technology in the Public Telephone Network: Where are we today?*, vol. 250, Greece, September 1997, pp. 63–72.

[87] Y.-H. Chao, W.-H. Tsai, H.-M. Wang, and R.-C. Chang, "Improving the characterization of the alternative hypothesis via minimum verification error training with applications to speaker verification," *Pattern Recogn.*, vol. 42, pp. 1351–1360, July 2009.

[88] A. E. Rosenberg, J. DeLong, C.-H. Lee, B.-H. Juang, and F. K. Soong, "The use of cohort normalized scores for speaker verification," in *Second international conference on spoken language processing*, 1992, pp. 599–602.

[89] A. E. Rosenberg, O. Siohan, and S. Parathasarathy, "Speaker verification using minimum verification error training," in *Acoustics, Speech and Signal Processing, 1998. Proceedings of the 1998 IEEE International Conference on*, vol. 1. IEEE, 1998, pp. 105–108.

[90] J. Markel, B. Oshika, and A. Gray Jr, "Long-term feature averaging for speaker recognition," *Acoustics, Speech and Signal Processing, IEEE Transactions on*, vol. 25, no. 4, pp. 330–337, 1977.

[91] T. Kinnunen, V. Hautamäki, and P. Fränti, "On the use of long-term average spectrum in automatic speaker recognition," in *5th Internat. Symposium on Chinese Spoken Language Processing (ISCSLP'06), Singapore*, 2006, pp. 559–567.

[92] K.-A. Lee, C. You, H. Li, T. Kinnunen, and D. Zhu, "Characterizing speech utterances for speaker verification with sequence kernel svm." in *INTERSPEECH*, 2008, pp. 1397–1400.

[93] A. Solomonoff, W. M. Campbell, and I. Boardman, "Advances in channel compensation for SVM speaker recognition," in *Proc. ICASSP*, vol. 1, 2005, pp. 629–632.

[94] O. Glembek, L. Burget, N. Dehak, N. Brummer, and P. Kenny, "Comparison of scoring methods used in speaker recognition with joint factor analysis," in *Acoustics, Speech and Signal Processing, 2009. ICASSP 2009. IEEE International Conference on.* IEEE, 2009, pp. 4057–4060.

[95] S. J. Prince and J. H. Elder, "Probabilistic linear discriminant analysis for inferences about identity," in *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on.* IEEE, 2007, pp. 1–8.

[96] M. A. Przybocki, A. F. Martin, and A. N. Le, "NIST speaker recognition evaluation chronicles-part 2," in *Speaker and Language Recognition Workshop, 2006. IEEE Odyssey 2006: The.* IEEE, 2006, pp. 1–6.

[97] C. S. Greenberg and A. F. Martin, "NIST speaker recognition evaluations 1996-2008," in *SPIE Defense, Security, and Sensing.* International Society for Optics and Photonics, 2009, pp. 732 411–732 411.

[98] A. F. Martin and C. S. Greenberg, "The NIST 2010 speaker recognition evaluation." in *INTERSPEECH*, 2010, pp. 2726–2729.

[99] N. Scheffer, L. Ferrer, M. Graciarena, S. Kajarekar, E. Shriberg, and A. Stolcke, "The sri NIST 2010 speaker recognition evaluation system," in *Acoustics, Speech and Signal Processing (ICASSP), 2011 IEEE International Conference on.* IEEE, 2011, pp. 5292–5295.

[100] R. Saeidi, K. Lee, T. Kinnunen, T. Hasan, B. Fauve, P. Bousquet, E. Khoury, P. S. Martinez, J. Kua, C. You *et al.*, "I4u submission to NIST sre 2012: A largescale collaborative effort for noise-robust speaker verification," in *Proc. Interspeech*, 2013.

[101] S. Boll, "Suppression of acoustic noise in speech using spectral subtraction," *IEEE Trans. on Acoustics, Speech, and Signal Processing*, vol. 27, pp. 113–120, 1979.

[102] L. Ferrer, H. Bratt, L. Burget, H. Cernocky, O. Glembek, M. Graciarena, A. Lawson, Y. Lei, P. Matejka, O. Plchot *et al.*, "Promoting robustness for speaker modeling in the community: the PRISM evaluation set," in *Proceedings of SRE11 Analysis Workshop*, 2011.

[103] M.-W. Mak and H.-B. Yu, "A study of voice activity detection techniques for NIST speaker recognition evaluations," *Computer Speech & Language*, vol. 28, no. 1, pp. 295–313, 2014.

[104] L. Buera, "Normalización y adaptación a entornos acústicos para la robustez en sistemas de reconocimiento automático del habla." Ph.D. dissertation, Universidad de Zaragoza, Zaragoza, Spain, 2007.

[105] R. Teunen, B. Shahshahani, and L. P. Heck, "A model-based transformational approach to robust speaker recognition." in *INTERSPEECH*, 2000, pp. 495–498.

[106] M. W. Mason, R. J. Vogt, B. J. Baker, and S. Sridharan, "Data-driven clustering for blind feature mapping in speaker verification," in *Proceedings Eurospeeech*, 2005, pp. 3109–3112.

[107] A. Acero, *Acoustical and environmental robustness in automatic speech recognition*. Springer, 1993.

[108] C. Leggetter and P. Woodland, "Maximum likelihood linear regression for speaker adaptation of continuous density hidden Markov models," *Computer speech and language*, vol. 9, no. 2, p. 171, 1995.

[109] M. Gales and S. Young, "Robust continuous speech recognition using parallel model combination," *IEEE Trans. on Speech and Audio Processing*, 1996.

[110] M. Gales, "Transformation smoothing for speaker and environmental adaptation." in *EUROSPEECH*, 1997.

[111] D. Kim, C. Kwan Un, and N. Kim, "Speech recognition in noisy environments using first-order vector Taylor series," *Speech Communication*, vol. 24, no. 1, pp. 39–49, 1998.

[112] A. Acero, L. Deng, T. Kristjansson, and J. Zhang, "HMM adaptation using vector Taylor series for noisy speech recognition," in *Sixth International Conference on Spoken Language Processing*, 2000, pp. 869–872.

[113] L. Deng, J. Droppo, and A. Acero, "Recursive estimation of nonstationary noise using iterative stochastic approximation for robust speech recognition," *Speech and Audio Processing, IEEE Transactions on*, vol. 11, no. 6, pp. 568–580, 2003.

[114] J. Dropo, L. Deng, and A. Acero, "Evaluation of the SPLICE algorithm on the Aurora2 database," in *Eurospeech*, vol. 2, Aalborg, Denmark, 2001, pp. 217–220.

[115] L. Buera, E. Lleida, J. Rosas, J. Villalba, A. Miguel, A. Ortega, and O. Saz, "Speaker verification and identification using phoneme dependent multi-environment models based linear normalization in adverse and dynamic acoustic environments," *Summer School for Advanced Studies on Biometrics for Secure Authentication: Multimodality and System Integration*, 2005.

[116] "NIST speaker recognition evaluation," http://www.nist.gov/itl/iad/mig/sre.cfm.

[117] J. J. Godfrey, E. C. Holliman, and J. McDaniel, "Switchboard: Telephone speech corpus for research and development," in *Acoustics, Speech, and Signal Processing, 1992. ICASSP-92., 1992 IEEE International Conference on*, vol. 1. IEEE, 1992, pp. 517–520.

[118] C. Cieri, D. Miller, and K. Walker, "The fisher corpus: a resource for the next generations of speech-to-text," in *Proceedings of LREC*, 2004, pp. 69–71.

[119] Y. Lei, L. Burget, L. Ferrer, M. Graciarena, and N. Scheffer, "Towards noise-robust speaker recognition using probabilistic linear discriminant analysis," in *Acoustics, Speech and Signal Processing (ICASSP), 2012 IEEE International Conference on*. IEEE, 2012, pp. 4253–4256.

[120] D. Martinez, L. Burget, T. Stafylakis, Y. Lei, P. Kenny, and E. Lleida, "Unscented transform for ivector-based noisy speaker recognition," sumitted to *Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on*, IEEE, 2014.

[121] P. Matejka, O. Glembek, F. Castaldo, M. Alam, O. Plchot, P. Kenny, L. Burget, and J. Cernocky, "Full-covariance UBM and heavy-tailed PLDA in i-vector speaker verification," in *IEEE Intl. Conf. on Acoustics, Speech and Sig. Proc. (ICASSP)*, 2011, pp. 4828–4831.

[122] M. E. Tipping and C. M. Bishop, "Mixtures of probabilistic principal component analyzers," *Neural computation*, vol. 11, no. 2, pp. 443–482, 1999.

[123] L. A. Ekman, W. B. Kleijn, and M. N. Murthi, "Regularized linear prediction of speech," *Audio, Speech, and Language Processing, IEEE Transactions on*, vol. 16, no. 1, pp. 65–73, 2008.

[124] H.-G. Hirsch and D. Pearce, "The Aurora experimental framework for the performance evaluation of speech recognition systems under noisy conditions," in *ASR2000-Automatic Speech Recognition: Challenges for the new Millenium ISCA Tutorial and Research Workshop (ITRW)*, 2000.

[125] H.-G. Hirsch, "FaNT-filtering and noise adding tool," http://dnt.kr.hsnr.de/download.html, 2005.

[126] D. Escobar-Valdivieso, "Improvement of joint factor analysis parameter estimation for a speaker verification task," MSc. dissertation, Tecnológico de Monterrey, Campus Monterrey, 2011.

[127] T. Fawcett, "An introduction to ROC analysis," *Pattern recognition letters*, vol. 27, no. 8, pp. 861–874, 2006.

[128] L. Yan, R. Dodier, M. C. Mozer, and R. Wolniewicz, "Optimizing classifier performance via the Wilcoxon-Mann-Whitney statistics," in *Proceedings of the 20th international conference on machine learning*, 2003, pp. 848–855.

[129] N. Brummer, "Measuring, refining and calibrating speaker and language information extracted from speech," Ph.D. dissertation, Stellenbosch: University of Stellenbosch, 2010.

[130] F. Provost and T. Fawcett, "Robust classification for imprecise environments," *Machine Learning*, vol. 42, no. 3, pp. 203–231, 2001.

[131] F. Sha and L. Saul, "Large margin Gaussian mixture modeling for phonetic classification and recognition," in *IEEE Intl. Conf. on Acoustics, Speech and Sig. Proc. (ICASSP)*, 2006, pp. 265–268.

[132] L. Saul and M. Rahim, "Maximum likelihood and minimum classification error factor analysis for automatic speech recognition," *Speech and Audio Processing, IEEE Transactions on*, vol. 8, no. 2, pp. 115–125, 2000.

[133] N. Brümmer and J. du Preez, "Application-independent evaluation of speaker detection," *Computer Speech & Language*, vol. 20, no. 2, pp. 230–275, 2006.

[134] N. Brummer and D. A. van Leeuwen, "On calibration of language recognition scores," in *Speaker and Language Recognition Workshop, 2006. IEEE Odyssey 2006.* IEEE, 2006, pp. 1–8.

[135] N. Brümmer and J. du Preez, "Application-independent evaluation of speaker detection," *Computer Speech & Language*, vol. 20, no. 2, pp. 230–275, 2006.

[136] D. Zhu, H. Li, B. Ma, and C.-H. Lee, "Optimizing the performance of spoken language recognition with discriminative training," *Audio, Speech, and Language Processing, IEEE Transactions on*, vol. 16, no. 8, pp. 1642–1653, 2008.

[137] D. Ramos-Castro, J. Fierrez-Aguilar, J. Gonzalez-Rodriguez, and J. Ortega-Garcia, "Speaker verification using speaker-and test-dependent fast score normalization," *Pattern recognition letters*, vol. 28, no. 1, pp. 90–98, 2007.

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

[139] T. Hasan and J. H. Hansen, "A study on universal background model training in speaker verification," *Audio, Speech, and Language Processing, IEEE Transactions on*, vol. 19, no. 7, pp. 1890–1899, 2011.

[140] Y. Tsao and C.-H. Lee, "An ensemble speaker and speaking environment modeling approach to robust speech recognition," *IEEE transactions on Speech*, vol. 43, pp. 781–785, August 1994.

[141] A. Sankar and C.-H. Lee, "A maximum-likelihood approach to stochastic matching for robust speech recognition," *Speech and Audio Processing, IEEE Transactions on*, vol. 4, no. 3, pp. 190–202, 1996.

[142] C. McCool, S. Marcel, A. Hadid, M. Pietikainen, P. Matejka, N. Poh, J. Kittler, A. Larcher, C. Levy, D. Matrouf *et al.*, "Bi-modal person recognition on a mobile phone: using mobile phone data," in *Multimedia and Expo Workshops (ICMEW), 2012 IEEE International Conference on*. IEEE, 2012, pp. 635–640.

[143] S. Marcel, C. McCool, P. Matějka, T. Ahonen, J. Černockỳ, S. Chakraborty, V. Balasubramanian, S. Panchanathan, C. H. Chan, J. Kittler *et al.*, "On the results of the first mobile biometry (MOBIO) face and speaker verification evaluation," in *Recognizing Patterns in Signals, Speech, Images and Videos*. Springer, 2010, pp. 210–225.

[144] J. Ortega-Garcia, J. Gonzalez-Rodriguez, and V. Marrero-Aguiar, "Ahumada: A large speech corpus in spanish for speaker characterization and identification," *Speech communication*, vol. 31, no. 2, pp. 255–264, 2000.

[145] J. Ortega-Garcia, S. Cruz-Llanas, and J. Gonzalez-Rodriguez, "Facing severe channel variability in forensic speaker verification conditions." in *EUROSPEECH*, 1999, pp. 783–786.

[146] C. Cortes and V. Vapnik, "Support-vector networks," *Machine Learning*, vol. 20, no. 3, pp. 273–297, 1995.

[147] E. Osuna, R. Freund, and F. Girosi, "Support vector machines: Training and applications," Massachusetts Institute of Technology Artificial Intelligence Laboratory, Technical Report AIM-1602, 1996.